



Guia do Desenvolvedor

AWS SDK para SAP ABAP



AWS SDKpara SAP ABAP: Guia do Desenvolvedor

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

As marcas comerciais e imagens comerciais 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 SDK da AWS para SAP ABAPé	1
Características do SDK da AWS para SAP ABAP	1
Manutenção	1
APIreferência	2
Definição de preço	2
Recursos	2
Conceitos básicos	3
Etapa 1: Prepare sua AWS conta	3
IAMfunção para SAP usuários	3
Autenticação	4
Etapa 2: instalar a SDK	6
Etapa 3: Configurar o SDK	6
Etapa 4: configuração funcional	8
Etapa 5: Autorizar usuários SAP	10
Etapa 6: escrever o código	12
Etapa 7: executar o aplicativo	15
Configuração	17
Pré-requisitos da SAP	17
SDK para SAP ABAP	17
SDK para SAP ABAP - edição BTP	21
Instalando SDK da AWS para SAP ABAP	22
Fazer download do SDK	22
Verifique o arquivo	22
AWS Transportes do SDK	23
Instalando SDK - BTP edição	27
Instalar SDK para SAP ABAP - BTP edição	27
Modules	27
Correção de patches SDK para SAP ABAP - edição BTP	28
Configuração	29
Configurações globais	30
Configurações técnicas	31
Configurar cenários	31
Configuração da aplicação	32
Perfil do SDK	32

Resolvedor lógico de recursos	33
Exemplo	34
Configurações de runtime	34
Registrar e rastrear	35
OPT-IN: telemetria aprimorada	35
Cenário ativo	35
Cenários avançados de conectividade	35
Conexão por meio de um servidor proxy	36
Conexão por meio de um pacote de inspeção do firewall	36
Endpoints de gateway	37
Endpoints de interface personalizada	37
Acesso de endpoints em várias regiões	38
Configurações do provedor de serviços	39
Atualização, rastreamento e telemetria	40
Atualização do sistema SAP	40
Rastreamento	41
Telemetria	42
Uso da SDK	43
Representação de dados	43
Tipos de dados	44
Tipos de dados do AWS	46
Exemplo de programa	47
Pré-requisitos	47
Código	48
Seções de código	48
Conceitos	51
Classes de API	51
Objetos adicionais	52
Classes de estrutura	52
Matrizes	54
Mapas	55
Funções de nível superior	55
Atributos	1
Configuração programática	56
Waiters	57
Paginadores	58

Comportamento de repetição	59
Criar produtos	60
Configurar um ID de produto	60
Limitações	61
Exemplos de código	62
Ações e cenários	62
Amazon Bedrock Runtime	63
CloudWatch	68
DynamoDB	76
Amazon EC2	90
Kinesis	106
Lambda	116
Amazon S3	131
SageMaker	139
Amazon SNS	158
Amazon SQS	167
Amazon Textract	175
Amazon Translate	185
Segurança	195
Autenticação do sistema	195
Autenticação de metadados	196
Autenticação da chave de acesso secreta	196
Autenticação baseada em certificado usando Roles Anywhere IAM	197
Próxima etapa	197
Práticas recomendadas de segurança do IAM	198
Melhores práticas para perfil de instância do Amazon EC2	198
Perfil do IAM para usuários do SAP	199
Autorizações da SAP	202
Autorizações para configuração	202
Autorizações SAP para usuários finais	203
Operações seguras	204
Criptografia de dados em repouso	204
Criptografia de dados em trânsito	205
Uso da API	2
Usar certificados	205
Pré-requisitos	205

Procedimento	206
Armazenamento de credenciais	21
Etapas de configuração	209
Usando o SAP Credential Store com o SDK	211
Solução de problemas	215
Falha de importação	215
Restrição de localização não especificada	215
Erro SSL	216
Configuração do perfil	217
Autorização do IAM	218
Autorização para ações	218
Cenário ativo	35
Caracteres especiais	219
Conectividade	219
Tópicos adicionais	220
Versões	220
Estratégia de lançamento	220
Práticas recomendadas	198
Patches de SDK para SAP ABAP	221
Instalação de um módulo adicional	221
Desinstalando o SDK para SAP ABAP	221
Licenciamento SAP	222
Histórico do documento	224
.....	CCXXV

O que SDK da AWS para SAP ABAP é

SDK da AWS para SAP ABAP fornece uma interface para os serviços oferecidos pelo AWS ABAP idioma. Usando o SDK, você pode implementar relatórios ABAPBADIs, transações, OData serviços e outros ABAP artefatos Serviços da AWS, como o Amazon Simple Storage Service (Amazon S3), Amazon DynamoDB e muito mais. Amazon Translate Você também pode desenvolver sistemas ABAP baseados em For, a partir da SAP NetWeaver versão 7.4 e em um ambiente de SAP Business Technology Platform. Para obter mais informações, consulte [Instalando AWS SDK para SAP ABAP - BTP edição](#).

Tópicos

- [Características do SDK da AWS para SAP ABAP](#)
- [Manutenção e suporte para as versões SDK principais](#)
- [API referência](#)
- [Definição de preço](#)
- [Recursos adicionais](#)

Características do SDK da AWS para SAP ABAP

SDK da AWS para SAP ABAP foi projetado para parecer familiar e natural para SAP os desenvolvedores. Por exemplo, embora todos Serviços da AWS usem as `false` strings `true` e para representar dados booleanos em JSON estruturas XML e dados, SDK for os SAP ABAP converte em valores ABAP -nativos 'X' e de um único caractere. ' ' SDK for SAP ABAP usa ABAP construções nativas tanto quanto possível, inclusive em tipos de dados e formatos de carimbo de data/hora. Como resultado, o ABAP programador não precisa se preocupar com a base JSON e a XML serialização ou com o formato eletrônico do API protocolo.

Manutenção e suporte para as versões SDK principais

Para obter informações sobre manutenção e suporte para as versões SDK principais e suas dependências subjacentes, consulte o seguinte no [Guia de referência de ferramentas AWS SDKs e ferramentas](#):

- [AWS SDK e política de manutenção de ferramentas](#)
- [AWS SDK e ferramentas e matriz de suporte de versões](#)

API referência

Para ver uma lista completa de SDK da AWS para SAP ABAP APIs, consulte [SDK da AWS para SAP ABAP - Guia API de referência](#).

Para ver uma lista completa de módulos SDK da AWS para SAP ABAP TLAs, consulte [SDK da AWS para SAP ABAP - Lista de módulos](#).

Para ver uma lista completa de módulos da versão prévia SDK para SAP ABAP desenvolvedores da BTP edição for TLAs, consulte [AWS SDK for SAP ABAP - BTP edition - Module List](#).

Definição de preço

SDK da AWS para SAP ABAP está disponível para você sem custo adicional. Você paga apenas pelos AWS recursos e serviços que consome com SDK o.

Recursos adicionais

Além deste guia, os seguintes recursos on-line estão disponíveis SDK para SAP ABAP.

- [SAP na AWS documentação](#)
- [Blog de desenvolvedores da AWS](#)
- [AWS fóruns de desenvolvedores](#)
- [AWS SDK Biblioteca de exemplos de código](#)
- [@awsdevelopers](#) (Twitter)

Começando com SDK da AWS para SAP ABAP

Esta seção descreve como começar a usar SDK o. Ele inclui informações sobre como instalar o SDK, realizar a configuração básica e criar um exemplo de código Hello World que traduz uma frase de um idioma para outro. Se você é novato AWS SDK, recomendamos realizar essas etapas em um ambiente sandbox.

Etapas

- [Etapa 1: Prepare sua AWS conta](#)
- [Etapa 2: instalar a SDK](#)
- [Etapa 3: Configurar o SDK](#)
- [Etapa 4: configuração funcional](#)
- [Etapa 5: Autorizar usuários SAP](#)
- [Etapa 6: escrever o código](#)
- [Etapa 7: executar o aplicativo](#)

Etapa 1: Prepare sua AWS conta

Para começar a usar SDK o for SAPABAP, você deve ter um ativo Conta da AWS . Você precisa de um, Conta da AWS mesmo que seu SAP sistema esteja hospedado localmente, na SAP Business Technology Platform (BTP) ou com outro provedor de nuvem.

Se o seu SAP sistema estiver sendo executado na AWS nuvem, você fará chamadas para AWS serviços no seu Conta da AWS.

Tópicos

- [IAM função para SAP usuários](#)
- [Autenticação](#)

IAM função para SAP usuários

- Crie uma IAM função com as instruções fornecidas no Guia do AWS Identity and Access Management usuário. Para obter mais informações, consulte [Criação de uma função para delegar](#)

[permissões a um AWS serviço](#). Anote o Amazon Resource Name (ARN) da IAM função para uso posterior.

- Selecione Amazon EC2 como caso de uso.
- Use SapDemoTranslate como nome do perfil.
- Anexe o perfil TranslateReadOnly ao perfil.
- A função deve ter as seguintes entidades para permitir que o SAP sistema assuma a função. Substitua **"111122223333"** pelo número da sua conta da AWS .

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "sts:AssumeRole"
      ],
      "Principal": { "AWS": "111122223333" }
    }
  ]
}
```

Este exemplo mostra que qualquer diretor da Conta da AWS **"111122223333"** pode assumir a função. Essa é uma permissão ampla que é adequada para proof-of-concept. Você pode usar uma entidade principal mais restrito para produção, como nos exemplos a seguir.

- Um usuário específico — quando o SAP sistema está usando um dos seguintes:
 - SSF-credenciais criptografadas de um sistema local SAP
 - Credenciais do serviço SAP Credential Store em SAPBTP, ambiente ABAP
- Uma função específica — quando o SAP sistema está na Amazon EC2 e há um perfil de instância.
- Amazon EC2 — quando o SAP sistema está na Amazon EC2 e não há perfil de instância.

Para obter mais informações, consulte [Melhores práticas de IAM segurança](#).

Autenticação

A autenticação depende de onde seu SAP sistema está hospedado.

Locais

- [Na AWS nuvem](#)
- [No local SAP BTP ou em outra nuvem](#)

Na AWS nuvem

Certifique-se de que a EC2 instância na qual seu SAP sistema está sendo executado tenha um perfil de instância com as seguintes permissões.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "VisualEditor0",
      "Effect": "Allow",
      "Action": "sts:AssumeRole",
      "Resource": "arn:aws:iam::111122223333:role/SapDemoTranslate"
    }
  ]
}
```

Adicione o ARN que você anotou na etapa anterior.

Essa permissão permite que seu SAP sistema assuma a SapDemoTranslate função em nome do ABAP usuário.

No local SAP BTP ou em outra nuvem

Se seu SAP sistema estiver localizado no local, em SAP BTP ou em outra nuvem, use as etapas a seguir para estabelecer uma conexão para autenticação usando a chave de acesso secreta.

1. Criar um usuário do IAM. Para obter mais informações, consulte [Criação de IAM usuários \(console\)](#).
2. Use SapDemoSID como nome do IAM usuário. SID é o ID do sistema do seu SAP sistema.
3. Atribua um perfil SapDemoTranslate a esse usuário.

Guarde a `access_key` e `secret_access_key`. Você deve configurar essas credenciais em seu SAP sistema.

Note

Se seu SAP sistema estiver localizado no local, em SAP BTP ou em outra nuvem, você poderá se autenticar usando uma das opções a seguir.

- [Autenticação de chave de acesso secreta](#) usando SSF no nosso SAP Credential Store
- [Usando certificados com IAM Roles Anywhere](#)

Etapa 2: instalar a SDK

Consulte as guias a seguir para obter instruções de instalação.

SDK for SAP ABAP

Importe SDK para SAP ABAP transportes em seu SAP sistema. Você pode importar os transportes para qualquer cliente. Para obter mais informações, consulte [Instalando SDK para SAP ABAP](#).

SDK for SAP ABAP - BTP edition

Instale SDK para SAP ABAP BTP edição usando o aplicativo Deploy Product. Para obter mais informações, consulte [Instalando SDK para SAP ABAP - BTP edição](#).

Etapa 3: Configurar o SDK

Antes de configurar o SDK, verifique se você tem as autorizações necessárias. Para obter mais informações, consulte [SAP autorizações](#).

Consulte as guias a seguir para obter instruções de configuração.

SDK for SAP ABAP

Execute a /AWS1/IMG transação para abrir o Guia de Implementação do SDK for SAP ABAP. Para executar essa transação, insira /n/AWS1/IMG na barra de comando do seu SAP sistema e escolha Enter.

Conclua as configurações a seguir.

- Acesse Pré-requisitos técnicos.

- Analise os [parâmetros](#) e a [HTTPSconectividade](#) recomendados.
- Vá para Configurações globais → Configurar cenários.
 - Altere as configurações, de acordo com as recomendações em [Configurações globais](#).
- Vá para Configurações globais → Configurações técnicas.
 - Altere as configurações, de acordo com as recomendações em [Configurações globais](#).
- Vá para Configurações de runtime → Registrar e rastrear.
 - Selecione Novas entradas.
 - Nível de rastreamento: sem rastreamento.
 - Linhas de despejo máximas: 100.
 - OPT-IN: enh telemetria: mantenha isso em branco.
 - Selecione Save (Salvar).
- Vá para Configurações de runtime → Cenário ativo.
 - Em Novo cenário, selecione DEFAULT.
 - Selecione Confirmar alteração de cenário.
 - Aceite a solicitação.

Pré-requisitos para sistemas on-premises

Se seu SAP sistema estiver sendo executado localmente ou em outra nuvem, as credenciais deverão ser armazenadas no seu SAP banco de dados. As credenciais são criptografadas usando SAP SSF e exigem uma biblioteca criptográfica configurada, como SAP a de. CommonCryptoLib

As etapas de configuração do SSF SDK for SAP ABAP estão descritas na /AWS1/IMG transação.

Note

O pré-requisito anterior não se aplica se seu SAP sistema estiver sendo executado na Amazon. EC2 SAPos sistemas em execução na Amazon EC2 recuperam credenciais de curta duração e rotatividade automática dos metadados da instância da Amazon. EC2

SDK for SAP ABAP - BTP edition

Abra seu ABAP ambiente em um navegador da Web e navegue até o aplicativo Custom Business Configurations.

Conclua as configurações a seguir.

- Vá para Configurar cenários.
 - Altere as configurações, de acordo com as recomendações em [Configurações globais](#).
- Vá para Configurações técnicas.
 - Altere as configurações, de acordo com as recomendações em [Configurações globais](#).

Etapa 4: configuração funcional

Consulte as guias a seguir para obter instruções de configuração.

SDK for SAP ABAP

Execute a transação /AWS1/IMG (insira /n/AWS1/IMG na barra de comando e escolha Enter) para abrir o guia de implementação do AWS SDK.

- Vá para Configuração do aplicativo → SDKPerfil.
 - Selecione Novas entradas.
 - Perfil: DEMO.
 - Descrição: Demo profile.
 - Selecione Save (Salvar).
 - Destaque a entrada que você criou e clique na ramificação de árvore Autenticação e Configurações.
 - Selecione Novas entradas.
 - SID: o ID do SAP sistema em que você está atualmente.
 - Cliente: O cliente do SAP sistema em que você está atualmente.
 - ID do cenário: a lista suspensa onde você encontrará o DEFAULT cenário criado pelo administrador do Basis.
 - AWS Região: insira a AWS região para a qual você deseja fazer chamadas. Se o SAP sistema estiver sendo executado AWS, insira a AWS região em que ele está sendo executado.

- Método de autenticação:
 - Selecione a função da instância por meio de metadados se seu SAP sistema estiver em execução na AmazonEC2.
 - Selecione Credenciais do SSF Armazenamento se seu SAP sistema estiver sendo executado no local ou em outra nuvem.
 - Selecione Definir credenciais.
 - Insira a ID da chave de acesso e a chave de acesso secreta que você criou na etapa anterior.
- Mantenha a opção Desativar IAM funções em branco.
- Selecione Save (Salvar).
- Clique na ramificação da árvore do IAMRole Mapping.
 - Selecione Novas entradas.
 - Insira o número de sequência: 010.
 - Insira a IAMfunção lógica:TESTUSER.
 - Insira a IAMfunção ARN: insira o arn:aws: da IAM função que contém a TranslateReadOnly política criada na etapa anterior.

SDK for SAP ABAP - BTP edition

Configure a autenticação usando o SAP Credential Store. Para obter mais informações, consulte [Usando o SAP Credential Store](#).

Abra seu ABAP ambiente em um navegador da Web e navegue até o aplicativo Custom Business Configurations.

- Vá para SDKPerfil.
 - Selecione Editar para criar um novo perfil.
 - Perfil: DEMO.
 - Descrição: Demo profile.
- Selecione a tecla de seta para a direita ao lado da entrada criada para navegar até a guia Autenticação e Configurações.

Selecione Novas entradas.

- Cliente: O cliente do SAP sistema em que você está atualmente.
- ID do cenário: a lista suspensa onde você encontrará o DEFAULT cenário criado pelo administrador do Basis.
- AWS Região: insira a AWS região para a qual você deseja fazer chamadas. Se o SAP sistema estiver sendo executado AWS, insira a AWS região em que ele está sendo executado.
- Método de autenticação: selecione Credenciais no SAP Credential Store.
- Insira o namespace e o nome da chave das credenciais armazenadas no SAP Credentials Store.
- Insira o nome do Acordo de Comunicação criado para estabelecer a comunicação entre SDK for SAP ABAP - BTP edition e o SAP Credential Store.
- Mantenha a opção Desativar IAM funções em branco.
- Clique com o botão direito do mouse na tecla de seta direita ao lado da entrada criada para navegar até a guia Mapeamento de IAM funções.

Selecione Novas entradas.

- Insira o número de sequência: 010.
- Insira a IAM função lógica: TESTUSER.
- Insira a IAM função ARN: insira o arn:aws: da IAM função que contém a `TranslateReadOnly` política criada na etapa anterior.

Etapa 5: Autorizar usuários SAP

SAPos usuários não estão autorizados a usar a AWS funcionalidade por padrão. Os usuários devem ser explicitamente autorizados usando SAP autorizações. Consulte as guias a seguir para obter mais detalhes.

SDK for SAP ABAP

Crie uma PFCG função

- Vá para a transação PFCG
- Insira o nome do perfil `ZAWS_SDK_DEMO_TESTUSER` e selecione Criar perfil único.
 - Descrição: Role for demo AWS SDK functionality
 - Vá até a guia Autorizações.

- Selecione Alterar dados de autorização e aceite o pop-up informativo.
- No pop-up Escolher modelo, selecione Não selecionar modelos.
- Selecione Adicionar manualmente na barra de ferramentas.
- Adicione os seguintes objetos de autorização:
 - /AWS1/LROL
 - /AWS1/SESS
- Na árvore de autorização, insira:
 - Perfil para acessar AWS APIs: DEMO
 - IAM Papel lógico: TESTUSER
- Selecione Save (Salvar).
- Selecione Gerar.
- Selecione Voltar.
- Selecione Salvar para salvar o perfil.

Atribuir a PFCG função aos SAP usuários

Qualquer usuário que tenha a ZAWS_SDK_DEMO_TESTUSER função atribuída será autorizado a usar AWS SDK funções com as configurações definidas no DEMO SDK perfil. O usuário autorizado também assumirá a IAM função mapeada para a IAM função TESTUSER lógica nesse perfil.

- Execute a transação SU01.
 - Insira o ID de usuário de um SAP usuário que testará a AWS SDK funcionalidade.
 - Selecione Alterar.
 - Vá até a guia Perfis e atribua o perfil ZAWS_SDK_DEMO_TESTUSER ao usuário.
 - Selecione Save (Salvar).

SDK for SAP ABAP - BTP edition

Crie uma função comercial

- Abra seu ABAP ambiente em um navegador da Web e navegue até o aplicativo Maintain Business Roles.
- Selecione Criar a partir do modelo e insira os detalhes a seguir.

- Modelo — Escolha **AWS1/RT_BTP_ENDUSER**.
- Nova ID da função comercial — insira uma ID.
- Descrição da nova função comercial — insira uma descrição.
- Selecione OK para ver a página da função comercial.
- Na guia Detalhes gerais da função, acesse Categorias de acesso e defina o campo Ajuda para gravação, leitura e valor como Restrito.
- Selecione Manter restrições e expanda Tipos de restrição atribuídos no painel de navegação esquerdo. Atualize o campo a seguir na seção Restrições e valores.
 - Em Escolher SDK sessão, selecione o ícone de lápis ao lado de SDKPerfil e navegue até a guia Intervalos. Digite **DEMO** e selecione Adicionar.
 - Em Escolher IAM função lógica, selecione o ícone de lápis ao lado de IAMFunção lógica e navegue até a guia Intervalos. Digite **TESTUSER** e selecione Adicionar.

Selecione o ícone de lápis ao lado SDKde Perfil e navegue até a guia Intervalos. Digite **DEMO** e selecione Adicionar

- Navegue de volta até o modelo de Função Comercial e abra a guia Usuários Comerciais. Selecione Adicionar para atribuir a função comercial recém-criada a um usuário SAP comercial que testará a SDK funcionalidade. Selecione Save (Salvar).

Qualquer usuário comercial atribuído à função comercial criada será autorizado a usar AWS SDK funções com as configurações definidas no DEMO SDK perfil. O usuário autorizado também assumirá a IAM função mapeada para a IAM função TESTUSER lógica nesse perfil.

Etapa 6: escrever o código

Consulte as guias a seguir para obter mais detalhes.

SDK for SAP ABAP

1. Encerrar transação SE38.
 - Insira ZDEMO_TRANSLATE_HELLO_WORLD como nome do programa.
 - Selecione Create.
 - Insira AWS SDK Hello World In Any Language como título.
 - Tipo: escolha Programa executável.

- Status: escolha Programa de teste.
- Selecione Save (Salvar).
- Salve o programa como um objeto local.

Adicione o seguinte código.

```
*&-----*
*& Report  ZAWS1_DEMO_XL8_SIMPLE
*&
*&-----*
*& A simple demo of language translation with AWS Translate
*&
*&-----*
REPORT zaws1_demo_xl8_simple.

START-OF-SELECTION.
  PARAMETERS pv_text TYPE /aws1/xl8boundedlengthstring DEFAULT 'Hello, World'
  OBLIGATORY.

  PARAMETERS pv_lang1 TYPE languageiso DEFAULT 'EN' OBLIGATORY.
  PARAMETERS pv_lang2 TYPE languageiso DEFAULT 'ES' OBLIGATORY.

  TRY.
    DATA(go_session) = /aws1/cl_rt_session_aws=>create( 'DEM0' ).
    DATA(go_xl8)      = /aws1/cl_xl8_factory=>create( go_session ).
    DATA(lo_output) = go_xl8->translatetext(
      iv_text          = pv_text
      iv_sourcelanguagecode = CONV /aws1/xl8languagecodestring( pv_lang1 )
      iv_targetlanguagecode = CONV /aws1/xl8languagecodestring( pv_lang2 )
    ).

    WRITE: / 'Source Phrase: ', pv_text.
    WRITE: / 'Target Phrase: ', lo_output->get_translatedtext( ).
  CATCH /aws1/cx_xl8unsuppedlanguage00 INTO DATA(lo_lang).
    WRITE: / 'ERROR' COLOR COL_NEGATIVE,
      'Cannot translate from',
      lo_lang->sourcelanguagecode,
      'to',
      lo_lang->targetlanguagecode.
  CATCH cx_root INTO DATA(lo_root).
    WRITE: / 'ERROR' COLOR COL_NEGATIVE, lo_root->get_text( ).
```

```
ENDTRY.
```

SDK for SAP ABAP - BTP edition

1. Clique com o botão direito do mouse no pacote em que a ABAP classe precisa ser criada e selecione Novo > ABAPclasse.
2. Insira **ZCL_DEMO_XL8_SIMPLE** o nome da classe e adicione uma descrição da classe. Escolha Próximo.
3. Crie ou escolha uma solicitação de transporte. Selecione Concluir.

Adicione o seguinte código.

```
CLASS zcl_demo_xl8_simple DEFINITION
  PUBLIC
  FINAL
  CREATE PUBLIC .

  PUBLIC SECTION.
    INTERFACES if_oo_adt_classrun.
  PROTECTED SECTION.
  PRIVATE SECTION.
ENDCLASS.

CLASS zcl_demo_xl8_simple IMPLEMENTATION.
  METHOD if_oo_adt_classrun~main.

    TRY.
      " input parameters
      DATA(pv_text) = |Hello, World|.
      DATA(pv_lang1) = |EN|.
      DATA(pv_lang2) = |ES|.

      DATA(go_session) = /aws1/cl_rt_session_aws=>create( 'DEMO' ).
      DATA(go_xl8)      = /aws1/cl_xl8_factory=>create( go_session ).
      DATA(lo_output) = go_xl8->translatetext(
        iv_text          = pv_text
        iv_sourcelanguagecode = pv_lang1
        iv_targetlanguagecode = pv_lang2
      ).
    CATCH cx_root.
  ENDTRY.
ENDMETHOD.
ENDCLASS.
```

```
        out->write( |Source Phrase: { pv_text }| ).
        out->write( |Target Phrase: { lo_output->get_translatedtext( ) }| ).
    CATCH /aws1/cx_xl8unsuppdedlanguage00 INTO DATA(lo_lang).
        out->write( |ERROR - Cannot translate from { lo_lang->sourcelanguagecode }
to { lo_lang->targetlanguagecode }| ).
    CATCH cx_root INTO DATA(lo_root).
        out->write( |ERROR - { lo_root->get_text( ) }| ).
    ENDTRY.
    ENDMETHOD.
ENDCLASS.
```

Para obter detalhes sobre como escrever ABAP código que usa o SDK, consulte [Usando SDK da AWS para SAP ABAP](#).

Etapa 7: executar o aplicativo

Consulte as guias a seguir para obter mais detalhes.

SDK for SAP ABAP

Execute o aplicativo no SE38. Se for bem-sucedido, o seguinte será o resultado.

```
Source Phrase: Hello, World
Target Phrase: Hola, mundo
```

Se você não tiver autorizações, configurações ou pré-requisitos básicos, poderá receber uma mensagem de erro. Veja o exemplo a seguir.

```
ERROR Could not find configuration under profile DEMO with
scenario DEFAULT for SBX:001
```

Se sua SAP função autorizar você a usar um SDK perfil e mapeá-lo para uma IAM função lógica enquanto suas IAM permissões não estiverem configuradas para que o SAP sistema assuma a IAM função, o seguinte será seu resultado.

```
ERROR Could not assume role arn:aws:iam::111122223333:role/SapDemoTranslate
```

Nesse caso, revise suas IAM permissões e configuração de confiança nas IAM funções, nos usuários ou em ambos definidos em [the section called “Etapa 1: Prepare sua AWS conta”](#).

SDK for SAP ABAP - BTP edition

Execute o aplicativo no Eclipse > Executar como > ABAPAplicativo (Console). Se for bem-sucedido, o seguinte será o resultado.

```
Source Phrase: Hello, World  
Target Phrase: Hola, mundo
```

Se você não tiver autorizações, configurações ou pré-requisitos básicos, poderá receber uma mensagem de erro. Veja o exemplo a seguir.

```
ERROR Could not find configuration under profile DEMO with  
scenario DEFAULT for SBX:001
```

Se sua SAP função autorizar você a usar um SDK perfil e mapeá-lo para uma IAM função lógica enquanto suas IAM permissões não estiverem configuradas para que o SAP sistema assuma a IAM função, o seguinte será seu resultado.

```
ERROR Could not assume role arn:aws:iam::111122223333:role/SapDemoTranslate
```

Nesse caso, revise suas IAM permissões e configuração de confiança nas IAM funções, nos usuários ou em ambos definidos em [the section called “Etapa 1: Prepare sua AWS conta”](#).

Configuração

Esta seção fornece informações sobre como configurar seu ambiente de desenvolvimento para uso do SDK da AWS para SAP ABAP.

Tópicos

- [Pré-requisitos da SAP](#)
- [Instalando SDK da AWS para SAP ABAP](#)
- [Instalando AWS SDK para SAP ABAP - BTP edição](#)

Pré-requisitos da SAP

Os pré-requisitos a seguir para instalar o SDK são aplicáveis quando seus sistemas SAP estão hospedados no. AWS

Tópicos

- [Pré-requisitos do AWS SDK for SAP ABAP](#)
- [Pré-requisitos do AWS SDK for SAP ABAP - edição BTP](#)

Pré-requisitos do AWS SDK for SAP ABAP

A seguir estão os pré-requisitos do AWS SDK for SAP ABAP.

Tópicos


- [Lançamento básico](#)
- [Lançamento do kernel](#)
- [Parâmetros](#)
- [Observações](#)
- [Conectividade de saída](#)
- [Conectividade HTTPS](#)
- [Acesso aos metadados da instância do Amazon EC2](#)

Lançamento básico

O SDK para SAP ABAP é compatível com o SAP NetWeaver 7.4 e superior. O SDK para SAP ABAP não toca em nenhuma tabela de aplicativos da SAP. É totalmente independente dos aplicativos, como o SAP Enterprise Resource Planning e o SAP Landscape Transformation Replication Server.

O nível mínimo de SP suportado para SAP_BASIS 740 é SP 0008. Para obter mais informações, consulte [SAP Note 1856171 - Supporting form fields of the same name in CL_HTTP_ENTITY](#) (requer acesso ao portal SAP). Com base nas suas necessidades de negócios, você pode escolher um nível de SP mais alto, conforme mostrado na imagem a seguir.

Installed Software Component Versions Installed Product Versions



Component	Release	SP-Level	Support Package	Short Description of Component
SAP_BASIS	740	0026	SAPKB74026	SAP Basis Component
SAP_ABA	740	0026	SAPKA74026	Cross-Application Component
SAP_GWFND	740	0027	SAPK-74027INSAPGWFND	SAP Gateway Foundation
SAP_UI	754	0008	SAPK-75408INSAPUI	User Interface Technology
PL_BASIS	740	0008	SAPK-74026INSAPBASIS	Basis Plug-In

Não há requisitos mínimos de nível de SP para versões do SAP_BASIS 750 e superiores.

Lançamento do kernel

O SDK para SAP ABAP e as ferramentas que usam o Internet Communication Manager (ICM) para conectividade HTTP dependem do kernel SAP para seus recursos criptográficos, HTTP, XML e JSON. Recomendamos usar a versão mais recente do kernel compatível com sua NetWeaver plataforma SAP. O requisito mínimo é a versão 741 do kernel. Para obter mais informações, consulte [SAP Note 2083594 - Versões do SAP Kernel e níveis de patch do SAP Kernel](#) (requer acesso ao portal SAP).

Se você estiver usando a versão 741 ou 742 do kernel, os seguintes níveis de patch são necessários:

- 741, atualização 212
- 742, atualização 111

Parâmetros

Seu sistema SAP deve oferecer suporte à Indicação de Nome de Servidor (SNI) conforme descrito nas seguintes notas SAP (requer acesso ao portal SAP).

- [SAP Note 2124480 - ICM/Web Dispatcher: Indicação de nome de servidor de extensão TLS \(SNI\) como cliente](#)
- [SAP Note 2582368 - Atualização do SAPSSL para envio da extensão TLS SNI pelo lado do cliente por saphttp, sapkprotp, sldreg](#)

Configure o seguinte parâmetro no arquivo DEFAULT.PFL.

```
icm/HTTPS/client_sni_enabled = TRUE
```

Observações

Aplique a seguinte nota SAP ao seu sistema.

- <https://launchpad.support.sap.com/#/notes/0001856171>
- <https://launchpad.support.sap.com/#/notes/0002619546>

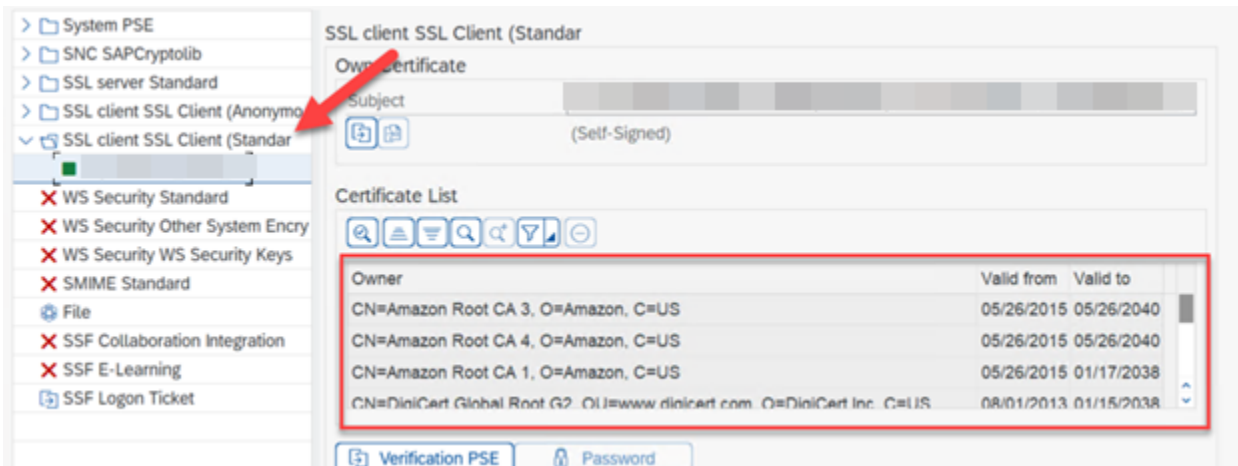
Conectividade de saída

O SDK para SAP ABAP é um cliente HTTPS. O sistema da SAP envia mensagens HTTPS de saída. Não há exigência de conectividade de entrada.

Conectividade HTTPS

Todas as chamadas de AWS API são feitas com canais HTTPS criptografados. O sistema SAP deve ser configurado para confiar em AWS certificados para estabelecer uma conexão HTTPS de saída.

1. Acesse <https://www.amazontrust.com/repository/>.
2. Em Root CAs, baixe todos os certificados usando o link PEM.
3. Importe esses certificados STRUST de seu SSL Client (Standard) PSE em cada um dos seus sistemas SAP, conforme mostrado na imagem a seguir.



Acesso aos metadados da instância do Amazon EC2

O sistema ABAP faz conexões HTTP não criptografadas com o localhost (<http://169.254.169.254>) para habilitar os metadados da instância Amazon EC2. O canal HTTP é usado somente para recuperar AWS credenciais do servidor local. O tráfego HTTP permanece dentro do host.

Os metadados permitem que um sistema SAP AWS se autentique com segurança sem armazenar uma chave secreta no SAP Secure Store. Só há um atributo aplicável somente aos sistemas SAP hospedados no Amazon EC2.

Configure o arquivo DEFAULT.PFL com o parâmetro a seguir para permitir que seu sistema SAP faça uma conexão HTTP de saída não criptografada.

```
icm/server_port_<xx> = PROT=HTTP,PORT=8000,TIMEOUT=60,PROCTIMEOUT=600
```

Use o parâmetro a seguir para habilitar a conexão HTTP de saída sem habilitar a conexão de entrada.

```
icm/server_port_<xx> = PROT=HTTP,PORT=0,TIMEOUT=60,PROCTIMEOUT=600
```

Verifique se seu sistema SAP está configurado para conexões HTTP de saída com as seguintes etapas:

1. Execute a transação SMICM.
2. Vá para Serviços ativos.
3. Verifique se você vê uma marca de seleção verde na linha HTTP, na coluna Ativo, conforme mostrado na imagem a seguir.

Active Services						
No.	Protocol	Service Name/Port	Host Name	Keep Alive	Proc.Timeo	Actv E:
1	HTTPS	50001		60	600	✓
2	HTTP	0		60	600	✓

Pré-requisitos do AWS SDK for SAP ABAP - edição BTP

A seguir estão os únicos pré-requisitos para o AWS SDK for SAP ABAP - edição BTP.

Tópicos

- [SAP Landscape Portal — edição BTP](#)
- [SAP Credential Store — edição BTP](#)

SAP Landscape Portal — edição BTP

Esse pré-requisito é aplicável somente ao AWS SDK for SAP ABAP - edição BTP.

O SAP Landscape Portal é o único mecanismo compatível para instalar complementos em um ambiente SAP BTP. Certifique-se de estar inscrito para usar este serviço. Para obter mais informações, consulte [Landscape Portal](#).

SAP Credential Store — edição BTP

Esse pré-requisito é aplicável somente ao AWS SDK for SAP ABAP - edição BTP.

Na versão prévia para desenvolvedores, a autenticação por chave de acesso secreta é o único mecanismo compatível para autenticar o AWS SDK para a edição SAP ABAP - BTP. O SDK lê as credenciais do Credential Store e armazena a chave de acesso secreta com segurança.

Você deve atender aos seguintes pré-requisitos.

- Assinatura da Credential Store.
- Armazenamento de credenciais atribuído como um direito à sua subconta BTP. Consulte [Configuração inicial](#) para obter mais detalhes.
- Uma instância de serviço com plano padrão para o Credential Store. Consulte [Criar uma instância de serviço](#) para obter mais detalhes.

Para obter mais informações, consulte [Usando o SAP Credential Store](#).

O serviço SAP Credential Store é executado no SAP BTP fora do sistema ABAP BTP. Consulte o [SAP Credential Store](#) para obter mais detalhes.

Instalando SDK da AWS para SAP ABAP

Tópicos

- [Fazer download do SDK para SAP ABAP](#)
- [Verifique o SDK para o arquivo SAP ABAP, opcional](#)
- [AWS Transportes do SDK](#)

Fazer download do SDK para SAP ABAP

```
curl "https://sdk-for-sapabap.aws.amazon.com/awsSdkSapabapV1/release/abapsdk-LATEST.zip" -o "abapsdk-LATEST.zip"
```

Quando o download estiver concluído, recomendamos que você descompacte o arquivo baixado em um diretório, como `/tmp/awssdk`.

Verifique o SDK para o arquivo SAP ABAP, opcional

Essa etapa opcional de validar a assinatura do seu arquivo SDK ajuda você a confirmar que seu SDK não foi adulterado. Use as etapas a seguir para verificar seu arquivo SDK.

1. Faça download do arquivo SDK SIGNATURE com o seguinte comando:

```
curl "https://sdk-for-sapabap.aws.amazon.com/awsSdkSapabapV1/release/abapsdk-LATEST.sig" -o "abapsdk-LATEST.sig"
```

2. Copie a chave pública a seguir e salve-a em um arquivo chamado `abapsdk-signing-key.pem`.

```
-----BEGIN PUBLIC KEY-----  
MIICIjANBgkqhkiG9w0BAQEFAAACAg8AMIICCgKCAgEAmS3oN3wKBh4HJ0Ga0tye  
15RR5909nuw0Jx0vEDCT709wUrXS3mjgEw6b6hvr2dLdoFr+eH4ewT5bV16U3gDv  
051sTdEJJpfLEWJJZZNK3v9fGWKyXgYe+ifmsPmf4lhNd2auzpvIy2UrlSYijCRB  
BWZFW+Ux00kILz+8vCFSXMZ6Z0qtLI1ZFbGrn6A5adbwwzf0qkg9BUEZK0wB6TAi
```

```
ZTnkMdBZGCBM9K2MRKKMxtirxUn+TFcAYyh5pM9tUAb2q4XE5m7092UnZG7ur/QY1
1FSZwAhQmk8hUPgUaq00QRC6z3TRzIGK0A/DI0cUPJMzFR4LCxEJkgh4rkRaU9V2
07DthUpj8b7QcQai0pnMpBf3zWLgbjNmX0hB0Eprg8/nVRHspf3zuisCJ21MPkz0
cH0R31MNsMLzm+d/gVklT31R/JwAcFCkXTWvR8/V0WNGZZXdVUbefrfI/k7fP60B
bzUrI1N4poq16rc4Tk5Derg+wQ7r0WjXkXop2kiCMjbYo0o10kS/At64PLjPz8dH
Zg25o79U9EJln+lpqZ297Ks+Hoct0v2GPbeeh0s7+N0fRTy0r81EZIUURLPKLVQUw
otVRzNDgLOA7eA667NrmegZfHCmqEwK9tXakZUHAcMzRPyhALc/HtmovxdStN9h1
JC4ex0GqstAv1fX5QaTbMSECAwEAAQ==
-----END PUBLIC KEY-----
```

3. Verifique o download do ZIP do SDK por meio do comando a seguir. O comando exige `openssl` que faz parte de muitas distribuições Linux.

```
openssl dgst -sha256 -verify abapsdk-signing-key.pem -keyform PEM -signature
abapsdk-LATEST.sig abapsdk-LATEST.zip
```

4. Verifique se a saída do comando anterior é `Verified OK`.
5. Se a saída for `Verification Failure`, repita as etapas anteriores. Se você continuar recebendo uma saída com falha, não instale o SDK e entre em contato com o AWS Support.

AWS Transportes do SDK

Tópicos

- [Conteúdo](#)
- [Importação](#)
- [Namespace](#)

Conteúdo

A instalação do SDK para SAP ABAP é concluída por meio do ABAP Transports. Você deve importar esses transportes para seu ambiente de desenvolvimento ou sandbox.

Cada versão do SDK para SAP ABAP substitui completamente a anterior. Não há necessidade de aplicar transportes incrementais. Os transportes são empacotados em um arquivo ZIP. A seguir está a estrutura do arquivo ZIP.

```
transports/
transports/core/
transports/core/Knnnnnn.AWS
```

```
transports/core/Rnnnnnn.AWS
transports/tla1/
transports/tla1/Knnnnnn.AWS
transports/tla1/Rnnnnnn.AWS
transports/tla2/
transports/tla2/Knnnnnn.AWS
transports/tla2/Rnnnnnn.AWS
.
.
.
```

A pasta `transports` contém uma subpasta `core`. A subpasta `core` contém os principais transportes de runtime e uma subpasta para cada módulo, nomeada pela abreviatura de três letras do módulo. Para obter uma lista completa dos módulos dos TLAs, consulte [SDK da AWS para SAP ABAP - Lista de módulos](#).

AWS Os transportes do SDK são solicitações de bancada de trabalho. Dependendo da configuração de suas rotas TMS, o SDK pode não ser encaminhado automaticamente para suas filas de garantia de qualidade e produção após a importação para o sistema anterior. Você deve adicioná-los manualmente à fila de cada sistema.

Quando seu projeto estiver pronto para a próxima fase, o AWS SDK poderá ser importado junto com transportes separados contendo seu próprio Z código com funcionalidade comercial. Se você estiver usando um sistema de controle de alterações, como o SAP Change Request Management (ChaRM), consulte o administrador do ChaRM para saber como lidar corretamente com transportes de terceiros.

Importação

Tópicos

- [Ponteiros principais](#)
- [Hora da importação](#)

AWS Os transportes do SDK são independentes do cliente. O transporte principal é obrigatório e contém o código de tempo de execução do SDK, a API e a API do Amazon Simple Storage Service. AWS Security Token Service Cada um dos módulos SDK restantes é entregue em um transporte separado. Para manter o tamanho do SDK pequeno em seu sistema, cada módulo do SDK é opcional. Você pode instalar módulos adicionais posteriormente, se necessário para sua lógica de negócios.

Por exemplo, se você quiser usar as APIs para o Amazon S3 Amazon Translate e importar core os transportes (contendo o tempo de execução principal, o Amazon S3 e os módulos) AWS STS e x18 os transportes (contendo o módulo para). Amazon Translate

Para ver uma lista completa das APIs do SDK para SAP ABAP, consulte [SDK para SAP ABAP - Guia de referência da API](#).

A seguir estão os principais indicadores ao importar transportes do AWS SDK.

- Cada transporte é entregue como Knnnnnn.AWS e Rnnnnnn.AWS
 - Knnnnnn.AWS deve ser copiado para /usr/sap/trans/cofiles
 - Rnnnnnn.AWS deve ser copiado para /usr/sap/trans/data.
- Ao importar transportes, você deve selecionar a opção Ignorar versão inválida do componente encontrada em Solicitação de transporte de importação > Opções > Opções de importação.
- Todos os transportes desejados podem ser importados simultaneamente.
- Se importar os transportes separadamente, o transporte core deve ser importado primeiro.
- O nível de liberação de todos os transportes deve ser idêntico.

Hora de importar

AWS Os transportes do SDK podem levar muitos minutos para serem importados. Os transportes são bem-sucedidos se o STMS mostrar uma luz verde (RC=0) ou amarela (RC=4).

- Uma luz vermelha (RC=8) indica que a importação teve um erro de sintaxe.
 - Selecione Solicitar → Exibir → Logs para examinar o erro de importação.
 - Durante a importação, se ocorrer um erro devido à falta de uma interface IF_SYSTEM_UUID_RFC4122_STATIC, verifique se o SAP Note 2619546 está aplicado ao sistema. Para obter mais informações, consulte [Notas](#).
 - Se a causa do erro for desconhecida, entre em contato AWS Support.
- Um raio vermelho (RC=12) indica que os arquivos de transporte não foram carregados corretamente para o /usr/sap/trans ou não têm as permissões necessárias.

Ponteiros principais

A seguir estão os principais indicadores ao importar transportes do AWS SDK.

- Cada transporte é entregue como Knnnnnn.AWS e Rnnnnnn.AWS

- Knnnnnn.AWS deve ser copiado para `/usr/sap/trans/cofiles`
- Rnnnnnn.AWS deve ser copiado para `/usr/sap/trans/data`.
- Ao importar transportes, você deve selecionar a opção Ignorar versão inválida do componente encontrada em Solicitação de transporte de importação > Opções > Opções de importação.
- Todos os transportes desejados podem ser importados simultaneamente.
- Se importar os transportes separadamente, o transporte core deve ser importado primeiro.
- O nível de liberação de todos os transportes deve ser idêntico.

Hora da importação

AWS Os transportes do SDK podem levar muitos minutos para serem importados. Os transportes são bem-sucedidos se o STMS mostrar uma luz verde (RC=0) ou amarela (RC=4).

- Uma luz vermelha (RC=8) indica que a importação teve um erro de sintaxe.
 - Selecione Solicitar → Exibir → Logs para examinar o erro de importação.
 - Durante a importação, se ocorrer um erro devido à falta de uma interface `IF_SYSTEM_UUID_RFC4122_STATIC`, verifique se o SAP Note 2619546 foi aplicado ao sistema. Para obter mais informações, consulte [Notas](#).
 - Se a causa do erro for desconhecida, entre em contato AWS Support.
- Um raio vermelho (RC=12) indica que os arquivos de transporte não foram carregados corretamente `/usr/sap/trans` ou não têm as permissões necessárias.

Namespace

O SDK para SAP ABAP usa o namespace `/AWS1/` e não modifica objetos SAP ou quaisquer outros objetos em seu sistema, com a seguinte exceção.

- AWS authos objetos estão em uma classe de objeto Auth. As classes de objetos de autenticação são limitadas a quatro caracteres e não oferecem suporte a namespaces. O SDK para SAP ABAP que usa classe de objeto de autenticação é YAW1. Se você já tiver uma classe de objeto de autenticação YAW1 na transaçãoSU21, entre em contato AWS Support antes da instalação.

Instalando AWS SDK para SAP ABAP - BTP edição

A BTP edição está na versão prévia para desenvolvedores e pode ser instalada juntando-se à versão prévia. Para instalar o SDK, preencha o formulário de participação na versão [prévia AWS SDK para SAP ABAP desenvolvedores da BTP edição](#).

Antes de instalar SDK para SAP ABAP - BTP edição, verifique se você está atendendo aos pré-requisitos exigidos. Para obter mais informações, consulte [SAPLandscape Portal](#) e [SAPCredential Store](#).

Tópicos

- [Instalar SDK para SAP ABAP - BTP edição](#)
- [Modules](#)
- [Correção de patches SDK para SAP ABAP - edição BTP](#)

Instalar SDK para SAP ABAP - BTP edição

1. Acesse sua instância do SAP Landscape Portal e inicie o aplicativo Deploy Product fiori.
2. Em Produtos, selecione **/AWS1/SDK_OMNI** em Produtos de parceiros.

Entre em contato AWS Support se você não ver **/AWS1/SDK_OMNI** depois de ser aceito na prévia para desenvolvedores.

3. Em Target Version, escolha a versão da SDK for SAP ABAP - BTP edition que você deseja instalar em seu sistema.
4. Em Sistemas Disponíveis, marque as caixas de seleção de todos os SIDs nos quais você deseja instalar o SDK
5. Selecione Implantar, insira os detalhes do agendamento e selecione Agendar. Você pode monitorar o progresso no Status de implantação da versão do produto.

A instalação pode levar de 30 a 45 minutos e inclui o tempo de inatividade do sistema. Para obter mais detalhes, consulte [Implantar produto](#).

Modules

Os módulos a seguir estão incluídos na prévia AWS SDK para SAP ABAP desenvolvedores da BTP edição for.

- [Amazon API Gateway \[agw\]](#)
- [Amazon Athena \[ath\]](#)
- [Amazon Bedrock Runtime \[bdr\]](#)
- [Amazon Comprehend \[cpd\]](#)
- [Amazon EventBridge \[evb\]](#)
- [Previsão da Amazon \[fcs\]](#)
- [Amazon Kinesis \[kns\]](#)
- [Amazon Data Firehose \[frh\]](#)
- [Amazon SageMaker \[sgm\]](#)
- [Serviço de notificação simples da Amazon \[...sns\]](#)
- [Amazon Simple Queue Service \[sqs\]](#)
- [Serviço Amazon Simple Storage \[s3\]](#)
- [AWS Systems Manager \[ssm\]](#)
- [Amazon Textract \[tex\]](#)
- [Amazon Transcribe \[tnb\]](#)
- [Amazon Translate \[x18\]](#)
- [AWS CloudTrail \[tr1\]](#)
- [AWS IoT \[iot\]](#)
- [AWS KMS \[kms\]](#)
- [AWS Lambda \[lmd\]](#)
- [AWS Secrets Manager \[smr\]](#)
- [AWS Security Token Service \[sts\]](#)
- [AWS Transfer Family \[trn\]](#)
- [IAM Funções em qualquer lugar \[...r1a\]](#)
- [Dados do Amazon Redshift \[API rsd\]](#)

Correção de patches SDK para SAP ABAP - edição BTP

O processo de correção SDK para a BTP edição for SAP ABAP - é semelhante ao processo de instalação. Se você instalar o SDK em um sistema que já tenha uma versão mais antiga instalada, ele será corrigido para a nova versão de sua escolha. SDK

Configurando SDK da AWS para SAP ABAP

Antes de usar SDK da AWS para SAP ABAP, você deve configurar o SDK com as configurações técnicas e funcionais necessárias para as SDK operações. Algumas configurações são transportáveis e outras são configurações de runtime. Muitas das configurações são diretamente análogas às configurações definidas nos .INI arquivos para outros SDKs.

SDKAs configurações, exceto as configurações de tempo de execução, devem ser concluídas em seu ambiente de desenvolvimento. Você pode transportar as configurações para o controle de qualidade e produção seguindo as regras usuais de transporte e controle de alterações. A configuração transportável não é recomendada para ambientes de produção.

Se você não tiver permissões para configurar AWS SDK, consulte [SAPautorizações](#).

Configurando SDK da AWS para SAP ABAP

Para executar a transação de configuração, insira /n/AWS1/IMG na barra de SAPGUI comando.

Configurando AWS SDK para SAP ABAP - edição BTP

Use as etapas a seguir para configurar SDK para SAP ABAP - BTP edição.

1. Abra seu ABAP ambiente em um navegador da web.
2. Navegue até o aplicativo Custom Business Configurations.

Para criar uma solicitação de personalização usando o aplicativo Export Customizing Transports, consulte [Trabalhando no aplicativo Export Customizing Transports](#) - Create Request.

No aplicativo Configuração Empresarial Personalizada, você pode agrupar configurações com base no tipo de SDK configuração. Use as etapas a seguir para agrupar as configurações.

1. Abra seu ABAP ambiente em um navegador da Web e navegue até o aplicativo Custom Business Configurations.
2. Selecione Configurações > Grupo e escolha Grupo de configuração na lista suspensa. Selecione OK.
3. As configurações agora estão disponíveis em uma estrutura hierárquica, conforme exibido na imagem. Para salvar a exibição, consulte [Visualizações \(gerenciamento de variantes\) - Componentes](#).

Custom Business Configurations (4)

Name	Description	
Application Configuration		
SDK Profile	Maintain AWS SDK Profile	>
Logical Resource Resolver	Maintain Logical Resource Resolution	>
Global Settings		
Technical Settings	Maintain Technical Settings	>
Configure Scenarios	Configure Scenarios	>

Esta seção abrange os seguintes tópicos.

Tópicos

- [Configurações globais](#)
- [Configuração da aplicação](#)
- [Configurações de runtime](#)
- [Cenários avançados de conectividade](#)
- [Configurações do provedor de serviços](#)
- [Tópicos de atualização, rastreamento e telemetria para SDK da AWS para SAP ABAP](#)

Configurações globais

Use a transação `/n/AWS1/IMG IMG` para AWS SDK para SAP ABAP e o aplicativo Custom Business Configuration para AWS SDK for SAP ABAP - edição BTP para definir as configurações globais. Este tópico usa IMG e Configuração Empresarial Personalizada de forma intercambiável.

Esta seção abrange os seguintes tópicos.

Tópicos

- [Configurações técnicas](#)

- [Configurar cenários](#)

Configurações técnicas

As configurações globais da transação /AWS1/IMG afetam o comportamento de todo o SDK. Essas configurações geralmente são definidas por um administrador do Basis. Você pode definir esses valores para as seguintes configurações recomendadas.

- Selecione Novas entradas.
 - Regionalização S3: [acesse us-east-1 buckets usando s3.amazonaws.com.](#)
 - Regionalização do STS: acesse o STS usando o endpoint global.
 - Desativar metadados do EC2: mantenha esse campo em branco. Esse campo é somente para leitura na edição BTP e está definido como 'Sim' por padrão.
 - Modo de extremidade de metadados: use um endpoint de metadados IPv4. Esse campo é somente para leitura na edição BTP e é atualizado automaticamente.
 - URL de envio de metadados: mantenha esse campo em branco. Esse campo é somente para leitura na edição BTP.
- Selecione Save (Salvar).

Configurar cenários

Os cenários permitem que o AWS SDK alterne as configurações com mais eficiência durante um cenário de teste de desastres em várias regiões ou de recuperação de desastres. Talvez você não precise desse atributo e, em vez disso, precise apenas configurar o seguinte cenário DEFAULT.

- Selecione Novas entradas.
 - ID do cenário: DEFAULT
 - Descrição do cenário: Cenário padrão
- Selecione Save (Salvar).

Se você tiver uma configuração de recuperação de desastres em várias regiões ou outros casos exclusivos que exijam uma alteração rápida das configurações, poderá configurar vários cenários.

- DEFAULT - Operação padrão.

- DR - Configuração especial se um desastre exigir a transferência de todo o sistema para outra região.
- DR_TEST - Configuração especial para simular um desastre, por exemplo, em um clone temporário da produção.

Configuração da aplicação

A configuração do SDK para SAP ABAP é semelhante à configuração de outros aplicativos baseados em ABAP. Ele é organizado em diferentes perfis para agrupar as configurações de vários cenários. Um perfil ABAP SDK define as configurações necessárias para um cenário de aplicativo específico. Por exemplo, se as transações ZVA01, ZVA02, e ZVA03 forem transações relacionadas a faturas aprimoradas e executadas Serviços da AWS, como Amazon S3, e AWS Lambda Amazon SageMaker, então um perfil de SDK chamado pode ser criado. ZINVOICE Esse perfil pode agrupar as configurações técnicas, as autorizações SAP e os mapeamentos de perfis do IAM para a funcionalidade relacionada à fatura.

Use a /n/AWS1/IMG transação para AWS SDK para SAP ABAP e o aplicativo Custom Business Configuration para AWS SDK for SAP ABAP - edição BTP para definir as configurações globais. Este tópico usa IMG e Configuração Empresarial Personalizada de forma intercambiável.

Tópicos

- [Perfil do SDK](#)
- [Resolvedor lógico de recursos](#)
- [Exemplo](#)

Perfil do SDK

Um perfil ABAP SDK define o seguinte para cada SID e cliente.

Note

O cliente está sempre 100% no ambiente SAP BTP, ABAP.

- A AWS região padrão para todas as chamadas de API. Por exemplo, se seus sistemas SAP estão sendo executados na us-east-1 região, é provável que seus outros AWS recursos também

estejam na mesma região, e essa deve ser sua região padrão. Seu código ABAP pode substituir a região padrão.

- Método de autenticação
 - Para sistemas SAP executados no Amazon EC2, é altamente recomendável escolher metadados de perfil de instância para se beneficiar das credenciais de curta duração e rotação automática.
 - Para sistemas SAP executados on-premises ou em outra nuvem, você deve escolher as credenciais do armazenamento SSF.
 - Para sistemas ABAP executados no SAP BTP, você deve escolher as credenciais do SAP Credential Store. Para obter mais informações, consulte [Usando o SAP Credential Store para autenticação](#).
- Um mapeamento dos perfis lógicos do IAM para os perfis do IAM.
 - Esse mapeamento é classificado em ordem decrescente de prioridade.
 - Um perfil do IAM de maior prioridade para a qual um usuário está autorizado em um perfil PFCG será selecionada automaticamente para o usuário.

Note

As funções do PFCG são chamadas de funções de negócios no ambiente SAP, BTP e ABAP.

Quando um programa ABAP quiser se conectar a um Serviço da AWS, ele especificará um perfil ABAP SDK que extrai as configurações necessárias. Uma AUTHORIZATION-CHECK será realizada para confirmar que o usuário tem permissões para acessar o perfil do SDK. Seu administrador de segurança SAP pode definir um perfil de PFCG concedendo acesso aos usuários apropriados.

Resolvedor lógico de recursos

O resolvedor lógico de recursos permite que você tenha um local padrão para armazenar nomes de recursos. Ele vem com o SDK para SAP ABAP. Sua ação é semelhante à forma como a transação de FILE mapeia nomes de arquivos lógicos para nomes de arquivos físicos.

Um recurso lógico define o conceito de um AWS recurso, como o bucket Amazon S3 que contém nossas faturas. Esse recurso lógico, por exemplo, pode ser nomeado ZINVOICES_OUTBOUND

e mapeado para um nome de bucket físico diferente, dependendo se o sistema SAP é de desenvolvimento, controle de qualidade ou produção.

O SDK para SAP ABAP é configurado de forma que um sistema de controle de qualidade resolva recursos lógicos para os recursos físicos de controle de qualidade, mesmo após uma atualização do sistema da produção. Os mapeamentos de recursos para TODOS os sistemas são definidos em seu sistema SAP de desenvolvimento e transportados adiante. Essa abordagem é diferente da configuração usual em sistemas SAP, em que o mapeamento é tratado como dados mestre e definido em cada sistema. A vantagem do resolvedor lógico de recursos oferecido pelo SDK para SAP ABAP é que as chances de um transporte incorreto após as atualizações do sistema são quase nulas.

Exemplo

Há quatro buckets separados do Amazon S3: um para desenvolvimento, produção e controle de qualidade, além de um segundo bucket de controle de qualidade para testes de regressão.

Quando o SDK resolve um recurso lógico, como ZINVOICE_OUTBOUND para um recurso físico, ele verifica SY-SYSID e SY-MANDT para perguntar em qual SID e cliente estou executando e seleciona automaticamente o recurso físico correto.

Se o mapeamento de um recurso em produção precisar mudar, você deverá alterar o mapeamento no sistema de desenvolvimento IMG e transportá-lo para frente. Isso garante que a reatribuição de AWS recursos a um sistema SAP esteja sujeita ao controle de alterações, como acontece com qualquer outro transporte.

Note

Como a configuração do SDK depende do cliente, a reatribuição de recursos é transportada em uma solicitação de personalização e o transporte deve ser importado para cada cliente.

Configurações de runtime

Esta seção abrange os seguintes tópicos.

Note

Essas configurações não são transportáveis e são locais para cada sistema SAP.

Tópicos

- [Registrar e rastrear](#)
- [OPT-IN: telemetria aprimorada](#)
- [Cenário ativo](#)

Registrar e rastrear

Você pode ativar um rastreamento para fins de depuração. É recomendável manter o nível de rastreamento em Sem rastreamento, a menos que seja diagnosticado um problema técnico. Para obter mais informações, consulte [operação segura](#).

Essas configurações não são aplicáveis ao SDK para SAP ABAP - edição BTP.

OPT-IN: telemetria aprimorada

Todos os SDKs enviam informações de telemetria AWS para fins de suporte. Você pode optar por aceitar a telemetria aprimorada. Isso é particularmente útil quando você entra em contato AWS Support para identificar a origem de uma chamada de API específica. Para obter mais informações, consulte [Rastreamento](#) e [Telemetria](#).

Essas configurações não são aplicáveis ao SDK para SAP ABAP - edição BTP.

Cenário ativo

Ative seu cenário DEFAULT nesta transação. Essa ativação é necessária apenas uma vez para cada sistema e não deve ser alterada, a menos que o sistema esteja passando por uma recuperação de desastres em várias regiões. Em uma configuração multirregional, você pode usar essa configuração para mudar seu sistema SAP para um ambiente de recuperação de desastres ou cenários de teste de recuperação de desastres.

Cenários avançados de conectividade

O SDK da AWS para SAP ABAP consome os Serviços da AWS fazendo chamadas HTTPS para endpoints da AWS. Em geral, os endpoints da AWS são acessados pela Internet. Um sistema SAP deve ser capaz de acessar a Internet para estabelecer essas conexões de saída. O SDK para SAP ABAP nunca requer uma conexão de entrada da Internet para o sistema SAP.

Os cenários a seguir oferecem maneiras diferentes de estabelecer a conexão de saída.

Cenários

- [Conexão por meio de um servidor proxy](#)
- [Conexão por meio de um pacote de inspeção do firewall](#)
- [Endpoints de gateway](#)
- [Endpoints de interface personalizada](#)
- [Acesso de endpoints em várias regiões](#)

Conexão por meio de um servidor proxy

Para estabelecer uma conexão por meio de um servidor proxy, use as etapas a seguir.

1. No SDK, acesse Transação **SICF**.
2. Escolha Executar.
3. No menu, escolha Cliente > Servidor proxy.
4. Defina a Configuração de proxy como Ativa.
5. No campo Sem proxy para os seguintes endereços, liste todas as exceções separadas por ponto e vírgula.
6. Nos campos Protocolo HTTP e Protocolo HTTPS, especifique os detalhes da conexão do seu servidor proxy.

O SDK não conhece o servidor proxy e não exige nenhuma configuração para usar a configuração do servidor proxy do sistema SAP.

Note

Se você usa a [autenticação de metadados de instância do Amazon EC2](#), o sistema SAP não pode usar o servidor proxy para acessar os metadados da instância local em `http://169.254.169.254`. Você deve incluir `169.254.169.254` no campo Sem proxy para os seguintes endereços.

Conexão por meio de um pacote de inspeção do firewall

Você pode configurar um pacote de inspeção do firewall para conexão de saída. Esses firewalls descriptografam o tráfego SSL e, em seguida, o criptografam novamente antes de passá-lo para o

endpoint. Essa configuração geralmente exige que o firewall emita seus próprios certificados para o sistema SAP que está consumindo um Serviço da AWS. Você deve instalar o certificado CA do seu firewall em STRUST. Para obter mais informações, consulte [Conectividade HTTPS](#).

Endpoints de gateway

Alguns Serviços da AWS oferecem endpoints de gateway para fornecer a uma VPC acesso de alto desempenho sem Internet. Esses endpoints são transparentes para o SDK para SAP ABAP e não exigem nenhuma configuração.

Para obter mais informações, consulte [Endpoints de gateway](#).

Endpoints de interface personalizada

Se precisar substituir a resolução padrão do endpoint por um endpoint personalizado, use um endpoint de interface para fornecer à sua VPC acesso de alto desempenho sem Internet. Para obter mais informações, consulte [Configurar um endpoint de interface](#).

Quando não estão usando DNS privado, esses endpoints têm seus próprios endereços DNS, e um programa ABAP deve substituir explicitamente a lógica usual de resolução de endpoints. Para obter mais informações, consulte [AWS re:Post Por que não consigo resolver nomes de domínio de serviço para um endpoint da VPC da interface?](#)

No exemplo a seguir, um endpoint de interface é criado para AWS STS e Amazon Translate. O sistema SAP não está usando DNS privado e chama os serviços com um endpoint personalizado. Os recursos lógicos definidos em `/AWS1/IMG` representam os endereços dos endpoints da interface física, como `vpce-0123456789abcdef-hd52vxz.translate.us-west-2.vpce.amazonaws.com`. Isso evita a codificação rígida do DNS no código.

No código a seguir, os recursos lógicos no `/AWS1/IMG` são primeiro resolvidos em nomes de endpoints físicos. Em seguida, eles são fornecidos aos métodos de fábrica da classe de sessão AWS (que usa AWS STS para assumir um perfil do IAM) e da classe de API de tradução.

```
" This example assumes we have defined our logical endpoints in /AWS1/IMG
" as logical resources so that we don't hardcode our endpoints in code.
" The endpoints may be different in Dev, QA and Prod environments.
DATA(lo_config) = /aws1/cl_rt_config=>create( 'DEMO' ).
DATA(lo_resolver) = /aws1/cl_rt_lresource_resolver=>create( lo_config ).

" logical resource STS_ENDPOINT should resolve to the interface endpoint
" for example vpce-0123456789-abcdefg.sts.us-west-2.vpce.amazonaws.com
```

```

DATA(lv_sts_endpoint) = lo_resolver->resolve_lresource( 'STS_ENDPOINT' ).

" logical resource XL8_ENDPOINT should resolve to the interface endpoint
" e.g. vpce-0123456789abcdefg-12345567.translate.us-west-2.vpce.amazonaws.com
DATA(lv_xl8_endpoint) = lo_resolver->resolve_lresource( 'XL8_ENDPOINT' ).

" the session itself uses the sts service to assume a role, so the
" session creation process requires a custom endpoint, specified here
DATA(lo_session) = /aws1/cl_rt_session_aws=>create(
  iv_profile_id = 'DEMO'
  iv_custom_sts_endpoint = |https://{ lv_sts_endpoint }|
).

" now we create an API object, and override the default endpoint with
" the custom endpoint
DATA(lo_xl8)      = /aws1/cl_xl8_factory=>create(
  io_session = lo_session
  iv_custom_endpoint = |https://{ lv_xl8_endpoint }| " provide custom endpoint
).
" now calls to lo_xl8 go to custom endpoint...

```

Conforme mostrado no exemplo, qualquer chamada de método no `go_xl8` vai para o endpoint `https://vpce-0123456789abcdefg-12345567.translate.us-west-2.vpce.amazonaws.com`.

Acesso de endpoints em várias regiões

O endpoint da AWS é determinado automaticamente a partir da Região da AWS padrão definida no perfil do SDK. Você também pode especificar uma região programaticamente, substituindo a região padrão. Isso pode ser substituído no método `CREATE()` de fábrica ou posteriormente com o objeto de configuração do SDK. Para obter mais informações, consulte [Configuração programática](#).

No exemplo a seguir, o método de fábrica `CREATE()` é usado para definir a região e listar as filas do Amazon SQS nas regiões `us-east-1` e `us-west-2`.

```

REPORT zdemo_sqs_queue_list.
parameters: profile type /AWS1/RT_PROFILE_ID OBLIGATORY.

START-OF-SELECTION.
DATA(go_session) = /aws1/cl_rt_session_aws=>create( profile ).
data(lt_region) = VALUE stringtab(
  ( |us-east-1| )

```

```
( |us-west-2| )
).

LOOP AT lt_region INTO DATA(lv_region).
  DATA(go_sqs) = /aws1/cl_sqs_factory=>create(
    io_session = go_session
    iv_region = conv /AWS1/RT_REGION_ID( lv_region )
  ).
  WRITE: / lv_region COLOR COL_HEADING.
  LOOP AT go_sqs->listqueues( )->get_queueurls( ) INTO DATA(lo_url).
    WRITE: / lo_url->get_value( ).
  ENDLLOOP.
ENDLOOP.
```

Configurações do provedor de serviços

Às vezes, os administradores do Basis precisam controlar certos recursos de SDK todo o sistema, a partir do cliente000. Esse é um cenário comum para provedores de hospedagem e serviços que operam sistemas em suas próprias AWS contas em nome de seus clientes. AWS SDK for SAP ABAP suporta as configurações do provedor de serviços. Essas configurações são definidas no cliente 000 e afetam SDK todos os clientes. As configurações do provedor de serviços não são suportadas na BTP edição SDK for SAP ABAP.

As configurações do provedor de serviços são definidas na transação /AWS1/IMG e devem ser definidas no cliente000. As configurações do provedor de serviços em outros clientes são ignoradas. As configurações no cliente entram em 000 vigor em todos os clientes e substituem outras IMG configurações em caso de conflito.

Use as etapas a seguir para definir as configurações do provedor de serviços no cliente000.

1. Expanda a ramificação Configurações do provedor de serviços na transação/AWS1/IMG.
2. Escolha as grades de proteção do provedor de serviços
3. Selecione Novas entradas e ajuste as configurações com base nos requisitos da sua empresa.
 - Desativar EC2 metadados — SDK impede o acesso aos metadados da EC2 instância em todos os clientes, mesmo que um SDK perfil esteja configurado para autenticação usando EC2 metadados da instância. Isso SDK gera uma exceção se um ABAP programa tentar acessar os metadados da instância usando o. SDK
4. Selecione Save (Salvar).

Tópicos de atualização, rastreamento e telemetria para SDK da AWS para SAP ABAP

Esta seção abrange os seguintes tópicos.

Tópicos

- [Atualização do sistema SAP](#)
- [Rastreamento](#)
- [Telemetria](#)

Atualização do sistema SAP

Depois de uma atualização do sistema, o principal desafio para um administrador do Basis é garantir que os sistemas separados não estejam acessando os recursos uns dos outros. Por exemplo, talvez você queira garantir que seu sistema SAP de controle de qualidade não esteja acessando os recursos, como um bucket S3, do seu cenário de produção.

O SDK para SAP ABAP fornece uma abordagem segura dos recursos lógicos para esse desafio. Um analista de negócios pode realizar as seguintes etapas:

1. Definir um recurso lógico, como ZINVOICE_OUTBOUND.
2. Mapear todos os sistemas e clientes no sistema de desenvolvimento.
3. Transportar a configuração de TODOS os sistemas até o cenário de produção.

Etapas básicas após uma atualização

1. Verifique a autenticação
 - Se o sistema estiver usando a autenticação de chave de acesso secreta, as credenciais criptografadas por SSF serão inválidas porque estão armazenadas nos dados principais. As credenciais devem ser inseridas novamente, o que pode exigir a regeneração de uma nova chave de acesso secreta em <https://console.aws.amazon.com/iam/>.
 - Se o sistema estiver se autenticando com metadados da instância EC2, nenhuma etapa será necessária.

Verifique as configurações de rastreamento

- No /AWS1/IMG, verifique se as configurações de rastreamento são as desejadas. Essas configurações não são transportáveis.

Rastreamento

A saída do rastreamento é controlada nas configurações de runtime do IMG.

Os níveis de rastreamento que você pode usar são:

- Sem rastreamento
- Chamadas de API de rastreamento
- Chamadas de API de rastreamento e carga útil

Essa opção contém informações de carga não criptografadas.

- Chamadas de API de rastreamento, carga útil e transformação interna de XML

Essa opção contém informações de carga não criptografadas.

Se o rastreamento da API for ativado, os rastreamentos serão gravados no DIR_WORK no arquivo `aws1_trace-YYYY-MM-DD.log`.

Se o rastreamento de carga útil também for ativado, arquivos adicionais com o título `aws1_payload_*` serão criados para cada chamada e componente de carga útil. O comprimento do rastreamento da carga útil pode ser limitado pelo comprimento aplicado a cada falha individual do rastreamento da carga útil.

Os rastreamentos de carga útil têm como objetivo principal coletar informações a serem fornecidas AWS Support no caso de um erro de serialização. Recomendamos que você escolha Sem rastreamento, a menos que esteja tentando diagnosticar um erro do SDK.

Note

Os rastreamentos de carga podem conter informações comerciais não criptografadas. Recomendamos ativar esses rastreamentos somente para uma solicitação do AWS Support para ajudá-lo a solucionar problemas. Você pode desativar esses rastreamentos após a

resolução. Os rastreamentos não são excluídos automaticamente e precisam ser removidos pelo administrador do sistema quando não são mais necessários.

Essas configurações não são aplicáveis ao SDK para SAP ABAP - edição BTP.

Telemetria

Os SDKs enviam informações de telemetria para o AWS Support. O SDK para SAP ABAP coleta as seguintes informações:

- Versão do sistema operacional e nível de patch
- Nível de lançamento e patch do SAP_BASIS
- Versão do SAP Kernel e nível de patch

Você pode optar por enviar as seguintes informações para AWS Support.

- SAP SID e nome da instância (host_sid_nn)
- Cliente SAP (SY-MANDT)
- Código da transação (SY-TCODE) e relatório (SY-REPID)

As informações adicionais AWS Support permitem ajudá-lo melhor. AWS Support pode detectar por que uma determinada chamada de API foi feita e pode encontrar ainda mais a transação relevante em um sistema SAP.

A telemetria é limitada às versões SDK e API do SDK for SAP ABAP - edição BTP.

Usando SDK da AWS para SAP ABAP

O SDK para SAP ABAP tem dois componentes principais.

- SDK Runtime (pacote /AWS1/RT): um conjunto de objetos que sustentam a segurança, a autenticação, o rastreamento, a configuração, a conversão de dados e outras funções entre APIs. Os módulos de API para Amazon S3 AWS STS, IAM Roles Anywhere e Secrets Manager são obrigatórios.
- APIs (pacote /AWS1/API e seus subpacotes): um subpacote para cada API em que os objetos de cada API são completamente independentes uns dos outros, garantindo que uma alteração em uma API não interrompa outra API. Para ver uma lista completa de APIs do SDK da AWS para SAP ABAP, consulte [SDK da AWS para SAP ABAP - Guia de referência de APIs](#).

Esta seção abrange os seguintes tópicos.

Tópicos

- [Representação de dados no ABAP](#)
- [Exemplo de programa do Amazon S3](#)
- [Conceitos do SDK para SAP ABAP](#)
- [SDK da AWS para SAP ABAP features](#)
- [Criar produtos com SDK](#)
- [Limitações](#)

Representação de dados no ABAP

Esta seção abrange os seguintes tópicos.

Tópicos

- [Tipos de dados](#)
- [Tipos de dados do AWS](#)

Tipos de dados

Os Serviços da AWS têm um conjunto padrão de tipos de dados que devem ser mapeados para tipos de dados ABAP. Para obter mais detalhes, consulte a tabela a seguir.

Tipo de dados do AWS	Tipo de dados do ABAP	Comentários
booleano	C	Caractere único "X" e " "
Segmento	STRING	
Byte	INT2	INT2 tem um intervalo maior do que 0-255. A maioria dos Serviços da AWS truncará os estouros, mas esse comportamento não está definido formalmente.
Short	INT2	
Inteiro	INT4	
Longo	19 DE DEZEMBRO	O INT8 não está disponível até o ABAP 750. O DEC19 é usado para compatibilidade e consistência em todas as plataformas ABAP suportadas.
Blob	XSTRING	Representa dados binários
Float	STRING	Embora o ABAP suporte DecFloats, ele não pode representar valores como NaN, Infinity e -Infinity. AWS SDK os representa internamente como STRINGS e os converte em DECFLOAT16 em runtime. Se NaN, Infinity ou +Infinity estiverem
Double	STRING	

Tipo de dados do AWS	Tipo de dados do ABAP	Comentários
		representados, o desenvolvedor poderá processá-los em resposta a um conjunto especial de exceções ou mapeamentos.
bigInteger	STRING	Esses valores representam números de comprimento infinito que não podem ser representados em ABAP, e STRINGS são usadas em vez de bigInteger.
bigDecimal	STRING	
Timestamp	TZNTSTMP	TZNTSTMP permite o processamento com funções nativas de carimbo de data/hora ABAP.

Serviços da AWS também retornam os seguintes tipos de dados agregados.

Tipo de dados do AWS	Tipo de dados do ABAP	Comentários
Estrutura	Classe	
Union	Classe	Uma união é o mesmo que uma estrutura, exceto que uma união nunca terá mais de um conjunto de campos. Todos os outros campos serão definidos como Sem valor.
Array	TABELA PADRÃO	
Hash	TABELA COM HASH	A tabela com hash terá apenas duas colunas: uma

Tipo de dados do AWS	Tipo de dados do ABAP	Comentários
		KEY (string) e uma VALUE (classe).

Tipos de dados do AWS

As seguintes abordagens foram integradas para dar suporte aos Serviços da AWS no ABAP.

- Certos tipos de dados da AWS não podem ser representados no ABAP. Por exemplo, o tipo de dados `float` em ABAP não suporta os valores `NaN`, `Infinity` ou `-Infinity`. Portanto, o tipo de dados `float` é representado como `STRING` e traduzido para `DECFLOAT16` em runtime.
- Os dados da AWS são representados no fio como `JSON` ou `XML`, e os valores são opcionais. Por exemplo, veja os exemplos a seguir retornados por um Serviço da AWS em `JSON`.

```
Fullname: {
  Firstname: "Ana",
  Middlename: "Carolina",
  Lastname: "Silva"
}
```

Se Ana não tiver um segundo nome, o serviço retornará a seguinte saída.

```
Fullname: {
  Firstname: "Ana",
  Lastname: "Silva"
}
```

O ABAP não distingue entre uma string de comprimento 0 e uma string que não tem valor. Outras linguagens podem atribuir um valor `NULL` à string ou envolvê-la em uma construção (como o wrapper `Optional<>` do Java). Eles não são suportados no ABAP. Portanto, o SDK for SAP ABAP facilita a distinção de valores ao fornecer variantes do método `getter`.

Exemplo de programa do Amazon S3

Esta seção mostra um exemplo de programa simples para listar o conteúdo de um bucket do Amazon S3 por meio de uma chamada de `ListObjectsV2`.

Tópicos

- [Pré-requisitos](#)
- [Código](#)
- [Seções de código](#)

Pré-requisitos

Você deve cumprir os seguintes pré-requisitos para executar este programa de exemplo.

- Você tem um bucket do Amazon S3. Neste tutorial, o bucket tem um nome `demo-invoices.customer.com`.
- Transação `/AWS1/IMG`:
 - Tem um perfil SDK definido chamado `DEMO_S3`.
 - No perfil do SDK, o perfil lógico do IAM TESTUSER deve ser mapeado para um perfil do IAM, como a `arn:aws:iam::111122223333:role/SapDemoFinance` que concede a permissão `s3:ListBucket` para listar o conteúdo do seu bucket do Amazon S3.
 - Tem um recurso lógico chamado `DEMO_BUCKET` que é mapeado para seu bucket do Amazon S3 com o SID e o cliente do seu sistema SAP.
- Seu usuário tem uma função PFCG que:
 - Autoriza o usuário a acessar o perfil do SDK `DEMO_S3` por meio do objeto de autenticação `- /AWS1/SESS`.
 - Autoriza o usuário a acessar o perfil lógico do IAM TESTUSER por meio do objeto de autenticação `- /AWS1/LROL`.
- Seu sistema SAP pode se autenticar à AWS usando o método definido no perfil do SDK.
- Seu perfil de instância do Amazon EC2 concede ao seu sistema SAP os direitos sobre a `sts:assumeRole` no perfil do IAM `arn:aws:iam::111122223333:role/SapDemoFinance` mapeado no perfil do SDK.

Código

O bloco de código a seguir mostra como deve ser a aparência do seu código.

```
REPORT  zdemo_s3_listbuckets.

START-OF-SELECTION.
  PARAMETERS pv_lres TYPE  /aws1/rt_resource_logical
                DEFAULT 'DEMO_BUCKET' OBLIGATORY.

  DATA(go_session) = /aws1/cl_rt_session_aws=>create( 'DEMO_S3' ).
  DATA(gv_bucket)  = go_session->resolve_lresource( pv_lres ).

  DATA(go_s3)      = /aws1/cl_s3_factory=>create( go_session ).

  TRY.
    DATA(lo_output) = go_s3->listobjectsv2(
      iv_bucket = CONV string( gv_bucket )
      iv_maxkeys = 100
    ).
    LOOP AT lo_output->get_contents( ) INTO DATA(lo_object).
      DATA lv_mdate TYPE datum.
      CONVERT TIME STAMP lo_object->get_lastmodified( )
        TIME ZONE 'UTC'
        INTO DATE lv_mdate.
      WRITE: / CONV text30( lo_object->get_key( ) ),
             lv_mdate, lo_object->get_size( ).
    ENDLOOP.
  CATCH /aws1/cx_rt_generic INTO DATA(lo_ex).
    DATA(lv_msg) = lo_ex->if_message~get_text( ).
    MESSAGE lv_msg TYPE 'I'.
  ENDTRY.
```

Seções de código

A seguir, é apresentada uma análise do código nas seções.

```
PARAMETERS pv_lres TYPE  /aws1/rt_resource_logical
                DEFAULT 'DEMO_BUCKET' OBLIGATORY.
```

O usuário não pode especificar um nome de bucket físico. Eles especificam um bucket lógico e os administradores do sistema (especificamente o analista de negócios), em coordenação com o administrador da AWS, mapeiam os buckets lógicos para os buckets físicos no /AWS1/IMG. Na maioria dos cenários de negócios, o usuário não tem a chance de escolher o bucket lógico: o ID do recurso lógico tem codificação rígida ou configurado em uma tabela de configuração personalizada.

```
DATA(go_session) = /aws1/cl_rt_session_aws=>create( 'DEMO_S3' ).
```

Essa linha estabelece uma sessão de segurança e declara que esse programa ABAP espera usar o perfil do SDK DEMO_S3. Essa chamada é a conexão com a configuração do SDK e extrai a região padrão, as configurações de autenticação e o perfil do IAM desejado. Uma chamada para AUTHORIZATION-CHECK é feita automaticamente para garantir que o objeto de autorização /AWS1/SESS seja satisfeito. Além disso, serão feitas chamadas AUTHORIZATION-CHECK para determinar o perfil lógico do IAM mais poderoso (número de sequência inferior) para o qual o usuário está autorizado, com base no objeto de autorização /AWS1/LROL. O SDK assumirá que o perfil do IAM está mapeado para o perfil lógico do IAM para o SID e o cliente. Em seguida, o objeto da sessão ativa o rastreamento com base nas configurações de rastreamento no IMG.

Se o usuário não estiver autorizado para o perfil do SDK solicitado ou para qualquer perfil lógico do IAM disponível, uma exceção será gerada.

```
DATA(gv_bucket) = go_session->resolve_lresource( pv_lres ).
```

Essa linha resolve o recurso lógico para um nome de bucket físico. Se o recurso lógico não puder ser resolvido porque a configuração não tem mapeamento para essa combinação de SID/cliente, uma exceção será gerada.

```
DATA(go_s3) = /aws1/cl_s3_factory=>create( go_session ).
```

Essa linha cria um objeto de API para o Amazon S3 usando o método `create()` de `/aws1/cl_s3_factory`. O objeto retornado é do tipo `/aws1/if_s3` que é a interface de uma API do Amazon S3. Um objeto de API separado deve ser criado para cada serviço. Por exemplo, se um programa ABAP estiver consumindo o Amazon S3, o AWS Lambda e o DynamoDB, ele criará objetos de API a partir de `/aws1/cl_s3_factory`, `/aws1/cl_lmd_factory` e `/aws1/cl_dyn_factory`.

Existem alguns parâmetros opcionais para o construtor que permitem especificar a região se você quiser substituir a região padrão configurada IMG. Dessa forma, pode haver duas instâncias de /aws1/ifs3, uma para us-east-1 e outra para us-west-2, se você quiser copiar objetos de um bucket em uma região para um bucket em outra região. Da mesma forma, você pode criar dois objetos de sessão de segurança diferentes e usá-los para criar duas instâncias separadas de /aws1/c1_s3, se precisar de um relatório para ler de um bucket relacionado a finanças e gravar objetos em um bucket relacionado à logística.

```
DATA(lo_output) = go_s3->listobjectsv2(  
    iv_bucket = CONV string( gv_bucket )  
    iv_maxkeys = 100  
).
```

Esta linha é uma chamada para ListObjectsV2. Ela exige argumentos de entrada simples e retorna um único objeto. Esses objetos podem representar dados JSON e XML profundos, desserializados em uma estrutura orientada a objetos ABAP. Isso pode ser bastante complicado em alguns casos. Agora você só precisa processar a saída para listar o conteúdo do bucket.

```
LOOP AT lo_output->get_contents( ) INTO DATA(lo_object).  
    DATA lv_mdate TYPE datum.  
    CONVERT TIME STAMP lo_object->get_lastmodified( )  
        TIME ZONE 'UTC'  
        INTO DATE lv_mdate.  
    WRITE: / CONV text30( lo_object->get_key( ) ),  
        lv_mdate, lo_object->get_size( ).  
ENDLOOP.
```

Os dados são acessados usando um método de estilo GET...() que oculta a representação interna dos dados. GET_CONTENTS() retorna uma tabela ABAP e cada linha contém um objeto representando uma única entrada do Amazon S3. Na maioria dos casos, o AWS SDK adota essa abordagem orientada a objetos e todos os dados são representados como objetos e tabelas. O campo LastModified é representado como um carimbo de data/hora que pode ser convertido em uma data com o comando CONVERT TIME STAMP nativo do ABAP. O GET_SIZE() retorna um INT4 para facilitar as operações matemáticas e de formatação.

```
CATCH /aws1/cx_rt_generic INTO DATA(lo_ex).
```



```
DATA(lv_msg) = lo_ex->if_message~get_text( ).  
MESSAGE lv_msg TYPE 'I'.
```

Todos os erros, conexão, cliente 4xx, servidor 5xx ou qualquer erro ABAP, como erros de autorização ou configuração, são representados como exceções. Você pode lidar com cada exceção separadamente. Você pode escolher se uma exceção deve ser tratada como um erro informativo, uma nova tentativa, um aviso, um atalho ou qualquer outro tipo de tratamento.

Conceitos do SDK para SAP ABAP

Esta seção aborda os conceitos básicos de SDK da AWS para SAP ABAP.

Tópicos

- [Classes de API](#)
- [Objetos adicionais](#)
- [Classes de estrutura](#)
- [Matrizes](#)
- [Mapas](#)
- [Funções de nível superior](#)

Classes de API

Cada um Serviço da AWS recebe um acrônimo de três letras ou TLA. O serviço é representado por uma interface no formato /AWS1/IF_<TLA>. Chamaremos isso de interface de serviço. A classe de API está no pacote /AWS1/API_<TLA>. A interface de serviço consiste em um método para cada AWS operação (chamaremos esses métodos de Métodos de Operação). Para ver uma lista completa de módulos de SDK da AWS para SAP ABAP TLAs, consulte [SDK da AWS para SAP ABAP - Lista de módulos](#).

Cada método de operação tem alguns argumentos IMPORTING e, no máximo, um argumento RETURNING. Normalmente, esses argumentos serão objetos com construtores complicados e um longo conjunto de métodos GET...(). Em muitos casos, os objetos conterão objetos aninhados, referências recursivas, tabelas de objetos, tabelas de tabelas e assim por diante. Isso ocorre porque Serviços da AWS estão passando estruturas XML e JSON profundas, que não podem ser representadas por um conjunto simples de argumentos.

O RETURNING argumento é sempre uma classe, mesmo que a classe contenha somente um único atributo.

Objetos adicionais

Além de conter a classe de API primária, cada pacote de API contém vários objetos relacionados de repositório e dicionário de dados.

- Uma classe para cada objeto do tipo estrutura.
- Uma classe para qualquer tipo de dados primitivo que aparece em uma tabela. Por exemplo, se um serviço retornar uma tabela de strings, a API ABAP a representará como uma tabela de objetos, em que cada objeto é uma classe wrapper que encapsula uma string. Isso é para que a classe wrapper possa ocultar os detalhes da representação de uma string nula que não pode ser representada nativamente no ABAP.
- Uma classe de exceção para quaisquer erros específicos definidos pelo serviço.
- Elementos de dados para cada tipo de dados primitivo. Cada tipo de dados tem seu próprio elemento de dados para ser autodocumentado.
- Objetos adicionais para processamento interno, como transformações XSLT para serializar e desserializar cargas XML e JSON.

Classes de estrutura

A maioria dos AWS dados enviados e recebidos pelo serviço é representada pelo AWS SDK como classes. Essas classes representam estruturas de dados e ocultam os detalhes internos do armazenamento. Em particular, as classes ocultam que a forma como o SDK representa esse campo não tem valor.

Para cada campo em uma classe de estrutura, há três métodos.

GET_field()

O método GET_field()

- Retorna o valor do campo ou
- Se o campo não tiver valor, ele retornará um valor padrão, que você pode definir como um parâmetro opcional.

Por exemplo, considere o código a seguir que imprime a restrição de localização de um bucket.

```
DATA(lo_location) = go_s3->getbucketlocation( iv_bucket = CONV string( gv_bucket ) ).
WRITE: / 'Bucket Location: ',
       lo_location->get_locationconstraint( ).
```

Se o bucket não tiver nenhuma restrição de localização (como no caso de us-east-1), ele `GET_LOCATIONCONSTRAINT()` retornará a string vazia. Você pode substituir esse comportamento e especificar o valor desejado se o campo não tiver nenhum valor.

```
DATA(lo_location) = go_s3->getbucketlocation( iv_bucket = CONV string( gv_bucket ) ).
WRITE: / 'Bucket Location: ',
       lo_location->get_locationconstraint( iv_value_if_missing = 'assuming us-east-1' ).
```

Agora o programa gravará `Bucket Location: assuming us-east-1` se o resultado de `getbucketlocation()` não retornar um local.

É possível solicitar que o método `GET()` retorne um resultado específico se o valor solicitado estiver completamente ausente, veja o exemplo de código a seguir.

```
data(lo_location) = go_s3->GETBUCKETLOCATION(
    new /AWS1/CL_S3_GET_BUCKET_LOC_REQ( iv_bucket = gv_bucket )
).
write: / 'Location constraint: ',
       lo_location->GET_LOCATIONCONSTRAINT( 'NopeNopeNope' ).
```

Nesse caso, se não houver restrição de localização, `GET_LOCATIONCONSTRAINT()` retornará `NopeNopeNope`.

HAS_field()

O método `HAS_field()` é uma forma de descobrir se o campo tem um valor ou não. Veja o exemplo a seguir.

```
if NOT lo_location->HAS_LOCATIONCONSTRAINT( ).
    write: / 'There is no location constraint'.
endif.
```

Se um determinado campo sempre tiver um valor, não haverá método `HAS_field()`.

`ASK_field()`

O método `ASK_field()` retorna o valor do campo ou gera uma exceção se não tiver valor. Essa é uma maneira conveniente de processar vários campos, sair da lógica e adotar uma abordagem diferente se algum dos campos não tiver valor.

```
TRY.  
    WRITE: / 'Location constraint: ', lo_location->ask_locationconstraint( ).  
CATCH /aws1/cx_rt_value_missing.  
    WRITE: / 'Never mind, there is no location constraint'.  
ENDTRY.
```

Observe que `/AWS1/CX_RT_VALUE_MISSING` é uma exceção estática e você receberá um aviso se optar por não pegá-la.

Práticas recomendadas

Em geral, você pode usar o método `GET_field()`, pois ele trata uma string nula como uma string vazia e é a mais parecida com ABAP das três opções. No entanto, ele não permite distinguir facilmente entre situações em que o campo tem um valor em branco e em que o campo não tem valor. Se sua lógica de negócios depende da distinção entre dados ausentes e dados em branco, os métodos `ASK` ou `HAS` permitem lidar com esses casos.

Matrizes

As matrizes são representadas como tabelas de objetos padrão ABAP.

Uma matriz JSON pode conter valores nulos, como a seguinte matriz: `['cat', 'dog', null, 'horse']`. Isso é chamado de matriz esparsa. Ele é representado no ABAP como uma tabela interna de referências de objetos e o valor `null` é representado na tabela como um valor ABAP `null` verdadeiro. Ao iterar por meio de uma tabela esparsa, você deve verificar os valores `null` para evitar acessar um objeto `null` e obter uma exceção `CX_SY_REF_IS_INITIAL`. Na prática, matrizes esparsas são raras em serviços AWS.

Para inicializar uma matriz de objetos, é conveniente usar as novas estruturas do ABAP 7.40. Considere este lançamento de uma instância do Amazon EC2 com vários grupos de segurança atribuídos:

```
ao_ec2->runinstances(  
    iv_imageid                = lo_latest_ami->get_imageid( )  
    iv_instancetype           = 't2.micro'  
    iv_maxcount               = 1  
    iv_mincount               = 1  
    it_securitygroupids       = VALUE /aws1/  
cl_ec2secgrpiddstringlist_w=>tt_securitygroupidstringlist(  
    ( NEW /aws1/  
cl_ec2secgrpiddstringlist_w( 'sg-12345678' ) )  
    ( NEW /aws1/  
cl_ec2secgrpiddstringlist_w( 'sg-55555555' ) )  
    ( NEW /aws1/  
cl_ec2secgrpiddstringlist_w( 'sg-99999999' ) )  
    )  
    iv_subnetid              = ao_snet->get_subnetid( )  
    it_tagsspecifications     = make_tag_spec( 'instance' )  
)
```

Mapas

Os mapas JSON são representados em ABAP como `Hashed Tables` onde cada linha da tabela tem apenas dois componentes.

- **KEY:** uma string que é a **UNIQUE KEY** da tabela.
- **VALUE:** um objeto contendo o valor.

Um mapa é um dos poucos casos em que o AWS SDK usa uma estrutura verdadeira, em vez de uma classe. Isso é necessário porque as tabelas com hash ABAP não podem ter uma referência de objeto como o campo chave, e as chaves de AWS mapa são sempre cadeias de caracteres não nulas.

Funções de nível superior

O [Classes de API](#) descrito na seção anterior espelha com precisão as APIs AWS de serviço e representa essas APIs como classes ABAP familiares. Em alguns casos, o SDK também inclui funções de nível superior que se baseiam nas classes da API para simplificar determinadas operações. As funções de nível superior são incluídas para conveniência do programador e não substituem as classes de API de nível inferior.

Se o SDK incluir funções de nível superior para um módulo, elas serão incluídas no mesmo transporte e poderão ser acessadas por meio de uma classe de fábrica chamada `/AWS1/CL_TLA_L2_FACTORY`. A classe de fábrica inclui métodos para criar vários clientes de nível superior para o módulo que são documentados junto com o restante da API com a [documentação da API](#).

SDK da AWS para SAP ABAP features

SDK da AWS para SAP ABAP fornece os seguintes recursos.

Tópicos

- [Configuração programática](#)
- [Waiters](#)
- [Paginadores](#)
- [Comportamento de repetição](#)

Configuração programática

Use a transação `/n/AWS1/IMG IMG` para AWS SDK para SAP ABAP e o aplicativo Custom Business Configuration para AWS SDK for SAP ABAP - edição BTP para configuração programática.

Para iniciar a configuração programática, comece recuperando um objeto de configuração com o comando `get_config()`.

```
data(lo_config) = lo_s3->get_config( ).
```

Cada objeto de configuração implementa a interface `/AWS1/IF_RT_CONFIG` que inclui GETters e SETters correspondentes ao IMG. Por exemplo, a região padrão pode ser substituída. No exemplo de comando a seguir:

```
lo_s3->get_config( )->/aws1/if_rt_config~set_region( 'us-east-1' ).
```

Alguns objetos de configuração não têm representação IMG e só podem ser definidos programaticamente, como o máximo de tentativas. No exemplo de comando a seguir:

```
lo_s3->get_config( )->/aws1/if_rt_config~set_max_attempts( 10 ).
```

O objeto de configuração do também Serviços da AWS pode incluir métodos específicos de serviço que não estão representados em `/aws1/if_rt_config`. Por exemplo, o Amazon S3 pode endereçar um bucket chamado foobucket usando um endpoint virtual `foobucket.s3.region.amazonaws.com` virtual ou um estilo de caminho `s3.region.amazonaws.com/foobucket`. Você pode impor o uso do estilo de caminho com o comando de exemplo a seguir.

```
lo_s3->get_config( )->set_forcepathstyle( abap_true ).
```

Para obter mais informações sobre configurações de serviço, consulte [SDK da AWS para SAP ABAP — Guia de referência da API](#).

Waiters

Ao trabalhar com AWS APIs assíncronas, você precisa esperar que um determinado recurso fique disponível antes de realizar outras ações. Por exemplo, a `CREATETABLE()` API do Amazon DynamoDB responde imediatamente com o status `CREATING` da tabela. Você pode iniciar operações de leitura ou gravação somente depois que o status da tabela for alterado para `ACTIVE`. Os garçons permitem que você confirme se AWS os recursos estão em um determinado estado antes de realizar ações neles.

Os garçons usam operações de serviço para pesquisar o status dos AWS recursos até que o recurso atinja o estado pretendido ou até que seja determinado que o recurso não atinge o estado desejado. Escrever o código para pesquisar recursos da AWS continuamente pode ser demorado e propenso a erros. Os waiters ajudam a simplificar essa complexidade, assumindo a responsabilidade de realizar pesquisas em seu nome.

Veja o exemplo a seguir do Amazon S3 usando o waiter.

```
DATA(lo_session) = /aws1/cl_rt_session_aws=>create( cv_pfl ).
DATA(lo_s3) = /aws1/cl_s3_factory=>create( lo_session ).

" Create a bucket - initiates the process of creating an S3 bucket and might return
before the bucket exists
lo_s3#createbucket( iv_bucket = |example-bucket| ).

" Wait until the newly created bucket becomes available
lo_s3->get_waiter( )->bucketexists(
    iv_max_wait_time = 200
```

```
    iv_bucket = |example-bucket|
  ).
```

- Neste exemplo, o cliente Amazon S3 é usado para criar um bucket. O comando `get_waiter()` é implementado para especificar quando o bucket existe.
- Você deve especificar o parâmetro `iv_max_wait_time` para cada waiter. Representa a quantidade total de tempo que um waiter deve esperar antes de terminar. No exemplo anterior, um waiter pode correr por 200 segundos.
- Talvez seja necessário fornecer entradas adicionais para os parâmetros necessários. No exemplo anterior, o nome do bucket do Amazon S3 é obrigatório para o parâmetro `iv_bucket`.
- A exceção `/AWS1/CX_RT_WAITER_FAILURE` indica que o waiter excedeu o tempo máximo especificado no parâmetro `iv_max_wait_time`.
- A exceção `/AWS1/CX_RT_WAITER_TIMEOUT` indica que o waiter parou por não atingir o estado desejado.

Paginadores

Algumas Serviço da AWS operações oferecem respostas paginadas. Elas são paginadas para retornar uma quantidade fixa de dados com cada resposta. Você precisa fazer solicitações subsequentes com um token ou marcador para recuperar todo o conjunto de resultados. Por exemplo, a operação `ListObjectsV2` do Amazon S3 retorna só até 1.000 objetos por vez. Você deve fazer solicitações subsequentes com o token apropriado para obter a próxima página de resultados.

Paginação é o processo de enviar solicitações sucessivas para continuar de onde uma solicitação anterior parou. Paginadores são iteradores de resultados fornecidos pelo SDK para SAP ABAP. Você pode usar APIs paginadas com facilidade e sem entender o mecanismo subjacente da API usando tokens de paginação.

Trabalhando com paginadores

Você pode criar paginadores com o método `get_paginador()` que retorna um objeto paginador. O objeto paginador chama a operação que está sendo paginada. O objeto paginador aceita os parâmetros obrigatórios a serem fornecidos à API subjacente. Esse processo retorna um objeto iterador que pode ser usado para iterar sobre resultados paginados, usando os métodos `has_next()` e `get_next()`.

- `has_next()` retorna um valor booleano indicando se há mais respostas ou páginas disponíveis para a operação chamada.
- `get_next()` retorna a resposta da operação.

O exemplo a seguir lista todos os objetos em um bucket do S3 recuperados usando o paginador.

```
DATA(lo_session) = /aws1/cl_rt_session_aws=>create( 'DEMO' ).
DATA(lo_s3) = /aws1/cl_s3_factory=>create( lo_session ).

TRY.
  DATA(lo_paginator) = lo_s3->get_paginator( ).
  DATA(lo_iterator) = lo_paginator->listobjectsv2(
    iv_bucket = 'example_bucket'
  ).
  WHILE lo_iterator->has_next( ).
    DATA(lo_output) = lo_iterator->get_next( ).
    LOOP AT lo_output->get_contents( ) INTO DATA(lo_object).
      WRITE: / lo_object->get_key( ), lo_object->get_size( ).
    ENDLLOOP.
  ENDWHILE.
CATCH /aws1/cx_rt_generic INTO DATA(lo_ex).
  MESSAGE lo_ex->if_message~get_text( ) TYPE 'I'.
ENDTRY.
```

Comportamento de repetição

O SDK para SAP ABAP permite que você configure o número máximo de novas tentativas de solicitações de Serviços da AWS que falham devido a erros transitórios ou de controle de utilização. O número de novas tentativas permitidas no nível do cliente de serviço, ou seja, o número de vezes que o SDK repete a operação antes de falhar e gerar uma exceção, é especificado pelo atributo `AV_MAX_ATTEMPTS` no objeto de configuração do serviço. Quando um objeto de cliente de serviço é criado, o SDK configura o atributo `AV_MAX_ATTEMPTS` com um valor padrão de 3. O objeto de configuração do serviço pode ser usado para definir programaticamente o número máximo de novas tentativas para um valor desejado. Consulte mais detalhes no exemplo a seguir.

```
" Retrieve configuration object using Amazon S3 service's get_config( ) method
DATA(lo_config) = lo_s3->get_config( ).

" Set the maximum number of retries to 5
lo_config->/aws1/if_rt_config~set_max_attempts( 5 ).
```

```
" Get the value of the maximum retry attempt.  
DATA(lv_max_retry_attempts) = lo_config->/aws1/if_rt_config~get_max_attempts( ).
```

Note

Embora o objeto de configuração do ABAP SDK permita que o modo de nova tentativa seja definido com o método `/AWS1/IF_RT_CONFIG~SET_RETRY_MODE()`, o SDK só permite o modo de nova tentativa `standard`. Para obter mais informações, consulte o [Guia de referência do comportamento](#) de novas tentativas no AWS SDKs and Tools.

Criar produtos com SDK

Um produto ou complemento do ABAP que consome Serviços da AWS pode aprimorar e ampliar os recursos do SDK. É possível criar esses produtos para usar com o SDK.

Tópicos

- [Configurar um ID de produto](#)

Configurar um ID de produto

É recomendável que você defina um ID de produto ao estabelecer uma sessão dentro de um produto ou complemento. Consulte mais detalhes no exemplo a seguir.

```
DATA(lo_session) = /aws1/cl_rt_session_aws=>create( 'DEMO' ).  
lo_session->set_product_id( 'INVOICE_ANALYZER' ).
```

O ID do produto deve conter somente letras, números e sublinhados sem espaços ou caracteres especiais. Você pode associá-lo ao nome técnico do produto ou a qualquer outro identificador. Se você desenvolver vários produtos ou complementos, o ID de produto deve ser exclusivo para cada produto. Por exemplo, os IDs dos produtos Invoice Analyzer, Tax Calculator e Pricing Engine podem ser `INVOICE_ANALYZER`, `TAX_CALCULATOR` e `PRICING_ENGINE`.

Adicionar um ID de produto à sessão aprimora a telemetria enviada à AWS com cada chamada de serviço. O ID de produto e o namespace do objeto que está fazendo a chamada estão incluídos na telemetria. Com essa telemetria, o AWS Support pode identificar o produto que está fazendo

a chamada caso o cliente enfrente problemas com o SDK. Isso pode ajudar a esclarecer que a chamada está realmente sendo feita pelo produto, e não pelo código do cliente.

Limitações

SDK da AWS para SAP ABAP inclui módulos SDK para todos Serviços da AWS. Alguns desses módulos podem ter limitações, conforme descrito aqui.

- Módulos que dependem de vinculações de MQTT protocolo, como `iotevents`, não funcionarão. MQTT não é um protocolo baseado em HTTP e atualmente não é suportado pelo SDK da AWS para SAP ABAP
- Módulos que dependem de atributos de streaming HTTP/2 ainda não são suportados. Algumas operações de serviços que funcionam com streams de eventos ainda não são suportadas, e as operações de streaming de mídia de serviços, como o Amazon Kinesis Video Streams, não funcionarão.

SDK da AWS para SAP ABAP tem as seguintes limitações de recursos.

- Os seguintes atributos do Amazon S3 não são suportados.
 - Pontos de acesso multirregionais
 - Criptografia do lado do cliente do Amazon S3

AWS O SDK para SAP ABAP - A edição BTP tem as seguintes limitações durante a versão prévia para desenvolvedores.

- Alguns módulos podem não estar disponíveis.
- Ele não pode ser desinstalado.
- Ele é atualizado com menos frequência.

Exemplos de código SDK para SAP ABAP

Os exemplos de código neste tópico mostram como usar o AWS SDK para SAP ABAP com. AWS

Ações são trechos de código de programas maiores e devem ser executadas em contexto.

Embora as ações mostrem como chamar funções de serviço específicas, é possível ver as ações contextualizadas em seus devidos cenários e exemplos entre serviços.

Cenários são exemplos de código que mostram como realizar uma tarefa específica chamando várias funções dentro do mesmo serviço.

Exemplos entre serviços são amostras de aplicações que funcionam em vários Serviços da AWS.

Exemplos

- [Ações e cenários usando o SDK para SAP ABAP](#)

Ações e cenários usando o SDK para SAP ABAP

Os exemplos de código a seguir mostram como realizar ações e implementar cenários comuns usando o AWS SDK para SAP ABAP com. Serviços da AWS

Ações são trechos de código de programas maiores e devem ser executadas em contexto.

Embora as ações mostrem como chamar funções de serviço específicas, é possível ver as ações contextualizadas em seus devidos cenários e exemplos entre serviços.

Cenários são exemplos de código que mostram como realizar uma tarefa específica chamando várias funções dentro do mesmo serviço.

Serviços

- [Exemplos de Amazon Bedrock Runtime usando o SDK para SAP ABAP](#)
- [CloudWatch exemplos usando SDK para SAP ABAP](#)
- [Exemplos do DynamoDB usando o SDK para SAP ABAP](#)
- [Exemplos do Amazon EC2 usando o SDK para SAP ABAP](#)
- [Exemplos do Kinesis usando o SDK para SAP ABAP](#)
- [Exemplos do Lambda usando o SDK para SAP ABAP](#)
- [Exemplos do Amazon S3 usando o SDK para SAP ABAP](#)
- [SageMaker exemplos usando SDK para SAP ABAP](#)

- [Exemplos do Amazon SNS usando o SDK para SAP ABAP](#)
- [Exemplos do Amazon SQS usando o SDK para SAP ABAP](#)
- [Exemplos do Amazon Textract usando o SDK para SAP ABAP](#)
- [Exemplos do Amazon Translate usando o SDK para SAP ABAP](#)

Exemplos de Amazon Bedrock Runtime usando o SDK para SAP ABAP

Os exemplos de código a seguir mostram como realizar ações e implementar cenários comuns usando o AWS SDK para SAP ABAP com o Amazon Bedrock Runtime.

Ações são trechos de código de programas maiores e devem ser executadas em contexto. Embora as ações mostrem como chamar funções de serviço específicas, é possível ver as ações contextualizadas em seus devidos cenários e exemplos entre serviços.

Cenários são exemplos de código que mostram como realizar uma tarefa específica chamando várias funções dentro do mesmo serviço.

Cada exemplo inclui um link para GitHub, onde você pode encontrar instruções sobre como configurar e executar o código no contexto.

Tópicos

- [Anthropic Claude](#)
- [Stable Diffusion](#)

Anthropic Claude

InvokeModel

O exemplo de código a seguir mostra como enviar uma mensagem de texto para Anthropic Claude usando a API Invoke Model.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

Invoque o modelo de base Claude 2 da Anthropic para gerar texto. Este exemplo usa recursos de /US2/CL_JSON que podem não estar disponíveis em algumas versões. NetWeaver

```

"Claude V2 Input Parameters should be in a format like this:
*   {
*     "prompt": "\n\nHuman:\n\nTell me a joke\n\nAssistant:\n",
*     "max_tokens_to_sample": 2048,
*     "temperature": 0.5,
*     "top_k": 250,
*     "top_p": 1.0,
*     "stop_sequences": []
*   }

DATA: BEGIN OF ls_input,
      prompt                TYPE string,
      max_tokens_to_sample TYPE /aws1/rt_shape_integer,
      temperature           TYPE /aws1/rt_shape_float,
      top_k                 TYPE /aws1/rt_shape_integer,
      top_p                 TYPE /aws1/rt_shape_float,
      stop_sequences        TYPE /aws1/rt_stringtab,
END OF ls_input.

"Leave ls_input-stop_sequences empty.
ls_input-prompt = |\n\nHuman:\n\n{ iv_prompt }\n\nAssistant:\n|.
ls_input-max_tokens_to_sample = 2048.
ls_input-temperature = '0.5'.
ls_input-top_k = 250.
ls_input-top_p = 1.

"Serialize into JSON with /ui2/cl_json -- this assumes SAP_UI is installed.
DATA(lv_json) = /ui2/cl_json=>serialize(
  data = ls_input
  pretty_name = /ui2/cl_json=>pretty_mode-low_case ).

TRY.
  DATA(lo_response) = lo_bdr->invokemodel(
    iv_body = /aws1/cl_rt_util=>string_to_xstring( lv_json )
    iv_modelid = 'anthropic.claude-v2'
    iv_accept = 'application/json'
    iv_contenttype = 'application/json' ).

"Claude V2 Response format will be:
*   {

```

```

*      "completion": "Knock Knock...",
*      "stop_reason": "stop_sequence"
*    }
DATA: BEGIN OF ls_response,
        completion TYPE string,
        stop_reason TYPE string,
      END OF ls_response.

/ui2/cl_json=>deserialize(
  EXPORTING jsonx = lo_response->get_body( )
            pretty_name = /ui2/cl_json=>pretty_mode-camel_case
  CHANGING data = ls_response ).

DATA(lv_answer) = ls_response-completion.
CATCH /aws1/cx_bdraccessdeniedex INTO DATA(lo_ex).
WRITE / lo_ex->get_text( ).
WRITE / |Don't forget to enable model access at https://
console.aws.amazon.com/bedrock/home?#/modelaccess|.

ENDTRY.

```

Invoke o modelo básico Anthropic Claude 2 para gerar texto usando o cliente de alto nível L2.

```

TRY.
  DATA(lo_bdr_l2_claude) = /aws1/cl_bdr_l2_factory=>create_claude_2( lo_bdr ).
  " iv_prompt can contain a prompt like 'tell me a joke about Java
  programmers'.
  DATA(lv_answer) = lo_bdr_l2_claude->prompt_for_text( iv_prompt ).
  CATCH /aws1/cx_bdraccessdeniedex INTO DATA(lo_ex).
  WRITE / lo_ex->get_text( ).
  WRITE / |Don't forget to enable model access at https://
console.aws.amazon.com/bedrock/home?#/modelaccess|.

ENDTRY.

```

- Para obter detalhes da API, consulte a [InvokeModel](#) referência da API AWS SDK for SAP ABAP.

Stable Diffusion

InvokeModel

O exemplo de código a seguir mostra como invocar o Stability.ai Stable Diffusion XL no Amazon Bedrock para gerar uma imagem.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

Crie uma imagem com difusão estável.

```
"Stable Diffusion Input Parameters should be in a format like this:
*  {
*    "text_prompts": [
*      {"text":"Draw a dolphin with a mustache"},
*      {"text":"Make it photorealistic"}
*    ],
*    "cfg_scale":10,
*    "seed":0,
*    "steps":50
*  }
TYPES: BEGIN OF prompt_ts,
        text TYPE /aws1/rt_shape_string,
      END OF prompt_ts.

DATA: BEGIN OF ls_input,
        text_prompts TYPE STANDARD TABLE OF prompt_ts,
        cfg_scale    TYPE /aws1/rt_shape_integer,
        seed         TYPE /aws1/rt_shape_integer,
        steps        TYPE /aws1/rt_shape_integer,
      END OF ls_input.

APPEND VALUE prompt_ts( text = iv_prompt ) TO ls_input-text_prompts.
ls_input-cfg_scale = 10.
ls_input-seed = 0. "or better, choose a random integer.
ls_input-steps = 50.
```



```

DATA(lv_json) = /ui2/cl_json=>serialize(
  data = ls_input
  pretty_name = /ui2/cl_json=>pretty_mode-low_case ).

TRY.
  DATA(lo_response) = lo_bdr->invokemodel(
    iv_body = /aws1/cl_rt_util=>string_to_xstring( lv_json )
    iv_modelid = 'stability.stable-diffusion-xl-v0'
    iv_accept = 'application/json'
    iv_contenttype = 'application/json' ).

  "Stable Diffusion Result Format:
  *
  * {
  *   "result": "success",
  *   "artifacts": [
  *     {
  *       "seed": 0,
  *       "base64": "iVBORw0KGgoAAAANSUhEUgAAAgAAA...
  *       "finishReason": "SUCCESS"
  *     }
  *   ]
  * }
  TYPES: BEGIN OF artifact_ts,
    seed          TYPE /aws1/rt_shape_integer,
    base64        TYPE /aws1/rt_shape_string,
    finishreason  TYPE /aws1/rt_shape_string,
  END OF artifact_ts.

  DATA: BEGIN OF ls_response,
    result        TYPE /aws1/rt_shape_string,
    artifacts     TYPE STANDARD TABLE OF artifact_ts,
  END OF ls_response.

  /ui2/cl_json=>deserialize(
    EXPORTING jsonx = lo_response->get_body( )
    pretty_name = /ui2/cl_json=>pretty_mode-camel_case
    CHANGING data = ls_response ).
  IF ls_response-artifacts IS NOT INITIAL.
    DATA(lv_image) =
      cl_http_utility=>if_http_utility~decode_x_base64( ls_response-artifacts[ 1 ]-
        base64 ).
    ENDIF.
  CATCH /aws1/cx_bdraccessdeniedex INTO DATA(lo_ex).
  WRITE / lo_ex->get_text( ).

```

```
WRITE / |Don't forget to enable model access at https://
console.aws.amazon.com/bedrock/home?#/modelaccess|.

ENDTRY.
```

Invoque o modelo básico Stability.ai Stable Diffusion XL para gerar imagens usando o cliente de alto nível L2.

```
TRY.
    DATA(lo_bdr_l2_sd) = /aws1/
cl_bdr_l2_factory=>create_stable_diffusion_10( lo_bdr ).
    " iv_prompt contains a prompt like 'Show me a picture of a unicorn reading an
enterprise financial report'.
    DATA(lv_image) = lo_bdr_l2_sd->text_to_image( iv_prompt ).
    CATCH /aws1/cx_bdraccessdeniedex INTO DATA(lo_ex).
    WRITE / lo_ex->get_text( ).
    WRITE / |Don't forget to enable model access at https://
console.aws.amazon.com/bedrock/home?#/modelaccess|.

ENDTRY.
```

- Para obter detalhes da API, consulte a [InvokeModel](#) referência da API AWS SDK for SAP ABAP.

CloudWatch exemplos usando SDK para SAP ABAP

Os exemplos de código a seguir mostram como realizar ações e implementar cenários comuns usando o AWS SDK para SAP ABAP com. CloudWatch

Ações são trechos de código de programas maiores e devem ser executadas em contexto. Embora as ações mostrem como chamar funções de serviço específicas, é possível ver as ações contextualizadas em seus devidos cenários e exemplos entre serviços.

Cenários são exemplos de código que mostram como realizar uma tarefa específica chamando várias funções dentro do mesmo serviço.

Cada exemplo inclui um link para GitHub, onde você pode encontrar instruções sobre como configurar e executar o código no contexto.

Tópicos

- [Ações](#)
- [Cenários](#)

Ações

DeleteAlarms

O código de exemplo a seguir mostra como usar DeleteAlarms.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.  
    lo_cwt->deletealarms(  
        it_alarmnames = it_alarm_names  
    ).  
    MESSAGE 'Alarms deleted.' TYPE 'I'.  
CATCH /aws1/cx_cwtresourcenotfound .  
    MESSAGE 'Resource being accessed is not found.' TYPE 'E'.  
ENDTRY.
```

- Para obter detalhes da API, consulte a [DeleteAlarms](#) referência da API AWS SDK for SAP ABAP.

DescribeAlarms

O código de exemplo a seguir mostra como usar DescribeAlarms.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.  
    oo_result = lo_cwt->describealarms(                " oo_result is returned  
for testing purposes. "  
    it_alarmnames = it_alarm_names  
    ).  
    MESSAGE 'Alarms retrieved.' TYPE 'I'.  
    CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).  
    DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception->  
>av_err_msg }|.  
    MESSAGE lv_error TYPE 'E'.  
ENDTRY.
```

- Para obter detalhes da API, consulte a [DescribeAlarms](#) referência da API AWS SDK for SAP ABAP.

DisableAlarmActions

O código de exemplo a seguir mostra como usar `DisableAlarmActions`.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
"Disables actions on the specified alarm. "  
TRY.  
    lo_cwt->disablealarmactions(  

```

```

        it_alarmnames = it_alarm_names
    ).
    MESSAGE 'Alarm actions disabled.' TYPE 'I'.
    CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
        DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
        MESSAGE lv_error TYPE 'E'.
    ENDTRY.

```

- Para obter detalhes da API, consulte a [DisableAlarmActions](#) referência da API AWS SDK for SAP ABAP.

EnableAlarmActions

O código de exemplo a seguir mostra como usar EnableAlarmActions.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```

"Enable actions on the specified alarm."
TRY.
    lo_cwt->enablealarmactions(
        it_alarmnames = it_alarm_names
    ).
    MESSAGE 'Alarm actions enabled.' TYPE 'I'.
    CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
        DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
        MESSAGE lv_error TYPE 'E'.
    ENDTRY.

```

- Para obter detalhes da API, consulte a [EnableAlarmActions](#) referência da API AWS SDK for SAP ABAP.

ListMetrics

O código de exemplo a seguir mostra como usar ListMetrics.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
"The following list-metrics example displays the metrics for Amazon CloudWatch."
TRY.
    oo_result = lo_cwt->listmetrics(           " oo_result is returned for
testing purposes. "
    iv_namespace = iv_namespace
    ).
    DATA(lt_metrics) = oo_result->get_metrics( ).
    MESSAGE 'Metrics retrieved.' TYPE 'I'.
CATCH /aws1/cx_cwtinvparamvalueex .
    MESSAGE 'The specified argument was not valid.' TYPE 'E'.
ENDTRY.
```

- Para obter detalhes da API, consulte a [ListMetrics](#) referência da API AWS SDK for SAP ABAP.

PutMetricAlarm

O código de exemplo a seguir mostra como usar PutMetricAlarm.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.
```

```
lo_cwt->putmetricalarm(  
    iv_alarmname           = iv_alarm_name  
    iv_comparisonoperator  = iv_comparison_operator  
    iv_evaluationperiods   = iv_evaluation_periods  
    iv_metricname         = iv_metric_name  
    iv_namespace          = iv_namespace  
    iv_statistic          = iv_statistic  
    iv_threshold          = iv_threshold  
    iv_actionsenabled     = iv_actions_enabled  
    iv_alarmdescription   = iv_alarm_description  
    iv_unit               = iv_unit  
    iv_period             = iv_period  
    it_dimensions         = it_dimensions  
).  
MESSAGE 'Alarm created.' TYPE 'I'.  
CATCH /aws1/cx_cwtlimitexceededfault.  
    MESSAGE 'The request processing has exceeded the limit' TYPE 'E'.  
ENDTRY.
```

- Para obter detalhes da API, consulte a [PutMetricAlarm](#) referência da API AWS SDK for SAP ABAP.

Cenários

Começar a usar alarmes

O exemplo de código a seguir mostra como:

- Criar um alarme.
- Desabilitar ações de alarme.
- Descrever um alarme.
- Excluir um alarme.

SDK para SAP ABAP

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
DATA lt_alarmnames TYPE /aws1/cl_cwtalarmnames_w=>tt_alarmnames.
DATA lo_alarmname TYPE REF TO /aws1/cl_cwtalarmnames_w.

"Create an alarm"
TRY.
    lo_cwt->putmetricalarm(
        iv_alarmname           = iv_alarm_name
        iv_comparisonoperator  = iv_comparison_operator
        iv_evaluationperiods   = iv_evaluation_periods
        iv_metricname          = iv_metric_name
        iv_namespace           = iv_namespace
        iv_statistic            = iv_statistic
        iv_threshold            = iv_threshold
        iv_actionsenabled       = iv_actions_enabled
        iv_alarmdescription     = iv_alarm_description
        iv_unit                  = iv_unit
        iv_period                = iv_period
        it_dimensions           = it_dimensions
    ).
    MESSAGE 'Alarm created' TYPE 'I'.
CATCH /aws1/cx_cwtlimitexceededfault.
    MESSAGE 'The request processing has exceeded the limit' TYPE 'E'.
ENDTRY.

"Create an ABAP internal table for the created alarm."
CREATE OBJECT lo_alarmname EXPORTING iv_value = iv_alarm_name.
INSERT lo_alarmname INTO TABLE lt_alarmnames.

"Disable alarm actions."
TRY.
    lo_cwt->disablealarmactions(
        it_alarmnames          = lt_alarmnames
    ).
```



```

    MESSAGE 'Alarm actions disabled' TYPE 'I'.
    CATCH /aws1/cx_rt_service_generic INTO DATA(lo_disablealarm_exception).
    DATA(lv_disablealarm_error) = |"{ lo_disablealarm_exception->av_err_code }"|
- { lo_disablealarm_exception->av_err_msg }|.
    MESSAGE lv_disablealarm_error TYPE 'E'.
    ENTRY.

"Describe alarm using the same ABAP internal table."
TRY.
    oo_result = lo_cwt->describealarms(                                " oo_result is
returned for testing purpose "
    it_alarmnames = lt_alarmnames
    ).
    MESSAGE 'Alarms retrieved' TYPE 'I'.
    CATCH /aws1/cx_rt_service_generic INTO DATA(lo_describealarms_exception).
    DATA(lv_describealarms_error) = |"{ lo_describealarms_exception-
>av_err_code }"| - { lo_describealarms_exception->av_err_msg }|.
    MESSAGE lv_describealarms_error TYPE 'E'.
    ENTRY.

"Delete alarm."
TRY.
    lo_cwt->deletealarms(
    it_alarmnames = lt_alarmnames
    ).
    MESSAGE 'Alarms deleted' TYPE 'I'.
    CATCH /aws1/cx_cwtresource_notfound .
    MESSAGE 'Resource being access is not found.' TYPE 'E'.
    ENTRY.

```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK para SAP ABAP.
 - [DeleteAlarms](#)
 - [DescribeAlarms](#)
 - [DisableAlarmActions](#)
 - [PutMetricAlarm](#)

Exemplos do DynamoDB usando o SDK para SAP ABAP

Os exemplos de código a seguir mostram como realizar ações e implementar cenários comuns usando o AWS SDK para SAP ABAP com o DynamoDB.

Ações são trechos de código de programas maiores e devem ser executadas em contexto. Embora as ações mostrem como chamar funções de serviço específicas, é possível ver as ações contextualizadas em seus devidos cenários e exemplos entre serviços.

Cenários são exemplos de código que mostram como realizar uma tarefa específica chamando várias funções dentro do mesmo serviço.

Cada exemplo inclui um link para GitHub, onde você pode encontrar instruções sobre como configurar e executar o código no contexto.

Tópicos

- [Ações](#)
- [Cenários](#)

Ações

CreateTable

O código de exemplo a seguir mostra como usar CreateTable.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.  
  DATA(lt_keyschema) = VALUE /aws1/cl_dynkeyschemaelement=>tt_keyschema(  
    ( NEW /aws1/cl_dynkeyschemaelement( iv_attributename = 'year'  
                                          iv_keytype = 'HASH' ) )  
    ( NEW /aws1/cl_dynkeyschemaelement( iv_attributename = 'title'  
                                          iv_keytype = 'RANGE' ) ) ).
```

```

DATA(lt_attributedefinitions) = VALUE /aws1/
cl_dynattributedefn=>tt_attributedefinitions(
  ( NEW /aws1/cl_dynattributedefn( iv_attributename = 'year'
                                   iv_attributetype = 'N' ) )
  ( NEW /aws1/cl_dynattributedefn( iv_attributename = 'title'
                                   iv_attributetype = 'S' ) ) ).

" Adjust read/write capacities as desired.
DATA(lo_dynprovthroughput) = NEW /aws1/cl_dynprovthroughput(
  iv_readcapacityunits = 5
  iv_writecapacityunits = 5 ).
oo_result = lo_dyn->createtable(
  it_keyschema = lt_keyschema
  iv_tablename = iv_table_name
  it_attributedefinitions = lt_attributedefinitions
  io_provisionedthroughput = lo_dynprovthroughput ).
" Table creation can take some time. Wait till table exists before
returning.
lo_dyn->get_waiter( )->tableexists(
  iv_max_wait_time = 200
  iv_tablename      = iv_table_name ).
MESSAGE 'DynamoDB Table' && iv_table_name && 'created.' TYPE 'I'.
" This exception can happen if the table already exists.
CATCH /aws1/cx_dynresourceinuseex INTO DATA(lo_resourceinuseex).
DATA(lv_error) = |"{ lo_resourceinuseex->av_err_code }" -
{ lo_resourceinuseex->av_err_msg }|.
MESSAGE lv_error TYPE 'E'.
ENDTRY.

```

- Para obter detalhes da API, consulte a [CreateTable](#) referência da API AWS SDK for SAP ABAP.

DeleteItem

O código de exemplo a seguir mostra como usar DeleteItem.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.
  DATA(lo_resp) = lo_dyn->deleteitem(
    iv_tablename      = iv_table_name
    it_key            = it_key_input ).
  MESSAGE 'Deleted one item.' TYPE 'I'.
CATCH /aws1/cx_dyncondalcheckfaile00.
  MESSAGE 'A condition specified in the operation could not be evaluated.'
TYPE 'E'.
CATCH /aws1/cx_dynresourcenotfoundex.
  MESSAGE 'The table or index does not exist' TYPE 'E'.
CATCH /aws1/cx_dyntransactconflictex.
  MESSAGE 'Another transaction is using the item' TYPE 'E'.
ENDTRY.
```

- Para obter detalhes da API, consulte a [DeleteItem](#) referência da API AWS SDK for SAP ABAP.

DeleteTable

O código de exemplo a seguir mostra como usar DeleteTable.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.
  lo_dyn->deletetable( iv_tablename = iv_table_name ).
  " Wait till the table is actually deleted.
  lo_dyn->get_waiter( )->tablenotexists(
    iv_max_wait_time = 200
    iv_tablename     = iv_table_name ).
  MESSAGE 'Table ' && iv_table_name && ' deleted.' TYPE 'I'.
CATCH /aws1/cx_dynresourcenotfoundex.
  MESSAGE 'The table ' && iv_table_name && ' does not exist' TYPE 'E'.
CATCH /aws1/cx_dynresourceinuseex.
  MESSAGE 'The table cannot be deleted since it is in use' TYPE 'E'.
ENDTRY.
```

- Para obter detalhes da API, consulte a [DeleteTable](#) referência da API AWS SDK for SAP ABAP.

DescribeTable

O código de exemplo a seguir mostra como usar DescribeTable.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).


```
TRY.  
    oo_result = lo_dyn->describetable( iv_tablename = iv_table_name ).  
    DATA(lv_tablename) = oo_result->get_table( )->ask_tablename( ).  
    DATA(lv_tablearn) = oo_result->get_table( )->ask_tablearn( ).  
    DATA(lv_tablestatus) = oo_result->get_table( )->ask_tablestatus( ).  
    DATA(lv_itemcount) = oo_result->get_table( )->ask_itemcount( ).  
    MESSAGE 'The table name is ' && lv_tablename  
           && '. The table ARN is ' && lv_tablearn  
           && '. The tablestatus is ' && lv_tablestatus  
           && '. Item count is ' && lv_itemcount TYPE 'I'.  
CATCH /aws1/cx_dynresourcenotfoundex.  
    MESSAGE 'The table ' && lv_tablename && ' does not exist' TYPE 'E'.  
ENDTRY.
```

- Para obter detalhes da API, consulte a [DescribeTable](#) referência da API AWS SDK for SAP ABAP.

GetItem

O código de exemplo a seguir mostra como usar GetItem.

SDK para SAP ABAP

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).


```
TRY.  
  oo_item = lo_dyn->getitem(  
    iv_tablename          = iv_table_name  
    it_key                = it_key ).  
  DATA(lt_attr) = oo_item->get_item( ).  
  DATA(lo_title) = lt_attr[ key = 'title' ]-value.  
  DATA(lo_year) = lt_attr[ key = 'year' ]-value.  
  DATA(lo_rating) = lt_attr[ key = 'rating' ]-value.  
  MESSAGE 'Movie name is: ' && lo_title->get_s( )  
    && 'Movie year is: ' && lo_year->get_n( )  
    && 'Moving rating is: ' && lo_rating->get_n( ) TYPE 'I'.  
  CATCH /aws1/cx_dynresourcenotfoundex.  
    MESSAGE 'The table or index does not exist' TYPE 'E'.  
ENDTRY.
```

- Para obter detalhes da API, consulte a [GetItem](#) referência da API AWS SDK for SAP ABAP.

ListTables

O código de exemplo a seguir mostra como usar ListTables.

SDK para SAP ABAP

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.
```

```

oo_result = lo_dyn->listtables( ).
" You can loop over the oo_result to get table properties like this.
LOOP AT oo_result->get_tablenames( ) INTO DATA(lo_table_name).
  DATA(lv_tablename) = lo_table_name->get_value( ).
ENDLOOP.
DATA(lv_tablecount) = lines( oo_result->get_tablenames( ) ).
MESSAGE 'Found ' && lv_tablecount && ' tables' TYPE 'I'.
CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
  DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
  MESSAGE lv_error TYPE 'E'.
ENDTRY.

```

- Para obter detalhes da API, consulte a [ListTables](#) referência da API AWS SDK for SAP ABAP.

PutItem

O código de exemplo a seguir mostra como usar PutItem.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```

TRY.
  DATA(lo_resp) = lo_dyn->putitem(
    iv_tablename = iv_table_name
    it_item      = it_item ).
  MESSAGE '1 row inserted into DynamoDB Table' && iv_table_name TYPE 'I'.
CATCH /aws1/cx_dyncondalcheckfaile00.
  MESSAGE 'A condition specified in the operation could not be evaluated.'
TYPE 'E'.
CATCH /aws1/cx_dynresourcenotfoundex.
  MESSAGE 'The table or index does not exist' TYPE 'E'.
CATCH /aws1/cx_dyntransactconflictex.
  MESSAGE 'Another transaction is using the item' TYPE 'E'.
ENDTRY.

```

- Para obter detalhes da API, consulte a [PutItem](#) referência da API AWS SDK for SAP ABAP.

Query

O código de exemplo a seguir mostra como usar Query.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```

TRY.
  " Query movies for a given year .
  DATA(lt_attributelist) = VALUE /aws1/
cl_dynattributevalue=>tt_attributevaluelist(
    ( NEW /aws1/cl_dynattributevalue( iv_n = |{ iv_year }| ) ) ).
  DATA(lt_key_conditions) = VALUE /aws1/cl_dyncondition=>tt_keyconditions(
    ( VALUE /aws1/cl_dyncondition=>ts_keyconditions_maprow(
      key = 'year'
      value = NEW /aws1/cl_dyncondition(
        it_attributevaluelist = lt_attributelist
        iv_comparisonoperator = |EQ|
      ) ) ) ).
  oo_result = lo_dyn->query(
    iv_tablename = iv_table_name
    it_keyconditions = lt_key_conditions ).
  DATA(lt_items) = oo_result->get_items( ).
  "You can loop over the results to get item attributes.
  LOOP AT lt_items INTO DATA(lt_item).
    DATA(lo_title) = lt_item[ key = 'title' ]-value.
    DATA(lo_year) = lt_item[ key = 'year' ]-value.
  ENDLLOOP.
  DATA(lv_count) = oo_result->get_count( ).
  MESSAGE 'Item count is: ' && lv_count TYPE 'I'.
  CATCH /aws1/cx_dynresourcenotfoundex.
  MESSAGE 'The table or index does not exist' TYPE 'E'.

```



```
ENDTRY.
```

- Para obter os detalhes da API, consulte [Query](#) na Referência da API AWS SDK para SAP ABAP.

Scan

O código de exemplo a seguir mostra como usar Scan.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.
    " Scan movies for rating greater than or equal to the rating specified
    DATA(lt_attributelist) = VALUE /aws1/
cl_dynattributevalue=>tt_attributevaluelist(
    ( NEW /aws1/cl_dynattributevalue( iv_n = |{ iv_rating }| ) ) ).
    DATA(lt_filter_conditions) = VALUE /aws1/
cl_dyncondition=>tt_filterconditionmap(
    ( VALUE /aws1/cl_dyncondition=>ts_filterconditionmap_maprow(
    key = 'rating'
    value = NEW /aws1/cl_dyncondition(
    it_attributevaluelist = lt_attributelist
    iv_comparisonoperator = |GE|
    ) ) ) ).
    oo_scan_result = lo_dyn->scan( iv_tablename = iv_table_name
    it_scanfilter = lt_filter_conditions ).
    DATA(lt_items) = oo_scan_result->get_items( ).
    LOOP AT lt_items INTO DATA(lo_item).
        " You can loop over to get individual attributes.
        DATA(lo_title) = lo_item[ key = 'title' ]-value.
        DATA(lo_year) = lo_item[ key = 'year' ]-value.
    ENDLLOOP.
    DATA(lv_count) = oo_scan_result->get_count( ).
    MESSAGE 'Found ' && lv_count && ' items' TYPE 'I'.
```

```
CATCH /aws1/cx_dynresourcenotfoundex.  
    MESSAGE 'The table or index does not exist' TYPE 'E'.  
ENDTRY.
```

- Para obter os detalhes da API, consulte [Scan](#) na Referência da API AWS SDK para SAP ABAP.

UpdateItem

O código de exemplo a seguir mostra como usar UpdateItem.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.  
    oo_output = lo_dyn->updateitem(  
        iv_tablename      = iv_table_name  
        it_key            = it_item_key  
        it_attributeupdates = it_attribute_updates ).  
    MESSAGE '1 item updated in DynamoDB Table' && iv_table_name TYPE 'I'.  
CATCH /aws1/cx_dyncondalcheckfaile00.  
    MESSAGE 'A condition specified in the operation could not be evaluated.'  
TYPE 'E'.  
CATCH /aws1/cx_dynresourcenotfoundex.  
    MESSAGE 'The table or index does not exist' TYPE 'E'.  
CATCH /aws1/cx_dyntransactconflictex.  
    MESSAGE 'Another transaction is using the item' TYPE 'E'.  
ENDTRY.
```

- Para obter detalhes da API, consulte a [UpdateItem](#) referência da API AWS SDK for SAP ABAP.

Cenários

Conceitos básicos de tabelas, itens e consultas

O exemplo de código a seguir mostra como:

- Criar uma tabela que possa conter dados de filmes.
- Colocar, obter e atualizar um único filme na tabela.
- Gravar dados de filmes na tabela usando um arquivo JSON de exemplo.
- Consultar filmes que foram lançados em determinado ano.
- Verificar filmes que foram lançados em um intervalo de anos.
- Excluir um filme da tabela e, depois, excluir a tabela.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
" Create an Amazon Dynamo DB table.

TRY.
  DATA(lo_session) = /aws1/cl_rt_session_aws=>create( cv_pfl ).
  DATA(lo_dyn) = /aws1/cl_dyn_factory=>create( lo_session ).
  DATA(lt_keyschema) = VALUE /aws1/cl_dynkeyschemaelement=>tt_keyschema(
    ( NEW /aws1/cl_dynkeyschemaelement( iv_attributename = 'year'
                                          iv_keytype = 'HASH' ) )
    ( NEW /aws1/cl_dynkeyschemaelement( iv_attributename = 'title'
                                          iv_keytype = 'RANGE' ) ) ).
  DATA(lt_attributedefinitions) = VALUE /aws1/
cl_dynattributedefn=>tt_attributedefinitions(
    ( NEW /aws1/cl_dynattributedefn( iv_attributename = 'year'
                                     iv_attributetype = 'N' ) )
    ( NEW /aws1/cl_dynattributedefn( iv_attributename = 'title'
                                     iv_attributetype = 'S' ) ) ).

" Adjust read/write capacities as desired.
DATA(lo_dynprovthroughput) = NEW /aws1/cl_dynprovthroughput(
```

```

        iv_readcapacityunits = 5
        iv_writecapacityunits = 5 ).
DATA(oo_result) = lo_dyn->createtable(
    it_keyschema = lt_keyschema
    iv_tablename = iv_table_name
    it_attributedefinitions = lt_attributedefinitions
    io_provisionedthroughput = lo_dynprovthroughput ).
" Table creation can take some time. Wait till table exists before
returning.
lo_dyn->get_waiter( )->tableexists(
    iv_max_wait_time = 200
    iv_tablename      = iv_table_name ).
MESSAGE 'DynamoDB Table' && iv_table_name && 'created.' TYPE 'I'.
" It throws exception if the table already exists.
CATCH /aws1/cx_dynresourceinuseex INTO DATA(lo_resourceinuseex).
    DATA(lv_error) = |"{ lo_resourceinuseex->av_err_code }" -
{ lo_resourceinuseex->av_err_msg }|.
    MESSAGE lv_error TYPE 'E'.
ENDTRY.

" Describe table
TRY.
    DATA(lo_table) = lo_dyn->describetable( iv_tablename = iv_table_name ).
    DATA(lv_tablename) = lo_table->get_table( )->ask_tablename( ).
    MESSAGE 'The table name is ' && lv_tablename TYPE 'I'.
CATCH /aws1/cx_dynresourcefoundex.
    MESSAGE 'The table does not exist' TYPE 'E'.
ENDTRY.

" Put items into the table.
TRY.
    DATA(lo_resp_putitem) = lo_dyn->putitem(
        iv_tablename = iv_table_name
        it_item       = VALUE /aws1/
cl_dynattributevalue=>tt_putiteminputattributemap(
        ( VALUE /aws1/cl_dynattributevalue=>ts_putiteminputattrmap_maprow(
            key = 'title' value = NEW /aws1/cl_dynattributevalue( iv_s =
'Jaws' ) ) )
        ( VALUE /aws1/cl_dynattributevalue=>ts_putiteminputattrmap_maprow(
            key = 'year' value = NEW /aws1/cl_dynattributevalue( iv_n = |
{ '1975' }| ) ) )
        ( VALUE /aws1/cl_dynattributevalue=>ts_putiteminputattrmap_maprow(
            key = 'rating' value = NEW /aws1/cl_dynattributevalue( iv_n = |
{ '7.5' }| ) ) )

```

```

    ) ).
    lo_resp_putitem = lo_dyn->putitem(
      iv_tablename = iv_table_name
      it_item      = VALUE /aws1/
cl_dynattributevalue=>tt_putiteminputattributemap(
  ( VALUE /aws1/cl_dynattributevalue=>ts_putiteminputattrmap_maprow(
    key = 'title' value = NEW /aws1/cl_dynattributevalue( iv_s = 'Star
Wars' ) ) ) )
  ( VALUE /aws1/cl_dynattributevalue=>ts_putiteminputattrmap_maprow(
    key = 'year' value = NEW /aws1/cl_dynattributevalue( iv_n = |
{ '1978' }| ) ) ) )
  ( VALUE /aws1/cl_dynattributevalue=>ts_putiteminputattrmap_maprow(
    key = 'rating' value = NEW /aws1/cl_dynattributevalue( iv_n = |
{ '8.1' }| ) ) ) )
  ) ).
    lo_resp_putitem = lo_dyn->putitem(
      iv_tablename = iv_table_name
      it_item      = VALUE /aws1/
cl_dynattributevalue=>tt_putiteminputattributemap(
  ( VALUE /aws1/cl_dynattributevalue=>ts_putiteminputattrmap_maprow(
    key = 'title' value = NEW /aws1/cl_dynattributevalue( iv_s =
'Speed' ) ) ) )
  ( VALUE /aws1/cl_dynattributevalue=>ts_putiteminputattrmap_maprow(
    key = 'year' value = NEW /aws1/cl_dynattributevalue( iv_n = |
{ '1994' }| ) ) ) )
  ( VALUE /aws1/cl_dynattributevalue=>ts_putiteminputattrmap_maprow(
    key = 'rating' value = NEW /aws1/cl_dynattributevalue( iv_n = |
{ '7.9' }| ) ) ) )
  ) ).
    " TYPE REF TO ZCL_AWS1_dyn_PUT_ITEM_OUTPUT
    MESSAGE '3 rows inserted into DynamoDB Table' && iv_table_name TYPE 'I'.
    CATCH /aws1/cx_dyncondalcheckfaile00.
    MESSAGE 'A condition specified in the operation could not be evaluated.'
    TYPE 'E'.
    CATCH /aws1/cx_dynresourcenotfoundex.
    MESSAGE 'The table or index does not exist' TYPE 'E'.
    CATCH /aws1/cx_dyntransactconflictex.
    MESSAGE 'Another transaction is using the item' TYPE 'E'.
    ENDTRY.

    " Get item from table.
    TRY.
      DATA(lo_resp_getitem) = lo_dyn->getitem(
        iv_tablename      = iv_table_name

```

```

        it_key          = VALUE /aws1/cl_dynattributevalue=>tt_key(
          ( VALUE /aws1/cl_dynattributevalue=>ts_key_maprow(
            key = 'title' value = NEW /aws1/cl_dynattributevalue( iv_s =
'Jaws' ) ) )
          ( VALUE /aws1/cl_dynattributevalue=>ts_key_maprow(
            key = 'year' value = NEW /aws1/cl_dynattributevalue( iv_n =
'1975' ) ) )
          ) ).
DATA(lt_attr) = lo_resp_getitem->get_item( ).
DATA(lo_title) = lt_attr[ key = 'title' ]-value.
DATA(lo_year) = lt_attr[ key = 'year' ]-value.
DATA(lo_rating) = lt_attr[ key = 'year' ]-value.
MESSAGE 'Movie name is: ' && lo_title->get_s( ) TYPE 'I'.
MESSAGE 'Movie year is: ' && lo_year->get_n( ) TYPE 'I'.
MESSAGE 'Movie rating is: ' && lo_rating->get_n( ) TYPE 'I'.
CATCH /aws1/cx_dynresourcenotfoundex.
  MESSAGE 'The table or index does not exist' TYPE 'E'.
ENDTRY.

" Query item from table.
TRY.
  DATA(lt_attributelist) = VALUE /aws1/
cl_dynattributevalue=>tt_attributevaluelist(
    ( NEW /aws1/cl_dynattributevalue( iv_n = '1975' ) ) ).
  DATA(lt_keyconditions) = VALUE /aws1/cl_dyncondition=>tt_keyconditions(
    ( VALUE /aws1/cl_dyncondition=>ts_keyconditions_maprow(
      key = 'year'
      value = NEW /aws1/cl_dyncondition(
        it_attributelist = lt_attributelist
        iv_comparisonoperator = |EQ|
      ) ) ) ).
  DATA(lo_query_result) = lo_dyn->query(
    iv_tablename = iv_table_name
    it_keyconditions = lt_keyconditions ).
  DATA(lt_items) = lo_query_result->get_items( ).
  READ TABLE lo_query_result->get_items( ) INTO DATA(lt_item) INDEX 1.
  lo_title = lt_item[ key = 'title' ]-value.
  lo_year = lt_item[ key = 'year' ]-value.
  lo_rating = lt_item[ key = 'rating' ]-value.
  MESSAGE 'Movie name is: ' && lo_title->get_s( ) TYPE 'I'.
  MESSAGE 'Movie year is: ' && lo_year->get_n( ) TYPE 'I'.
  MESSAGE 'Movie rating is: ' && lo_rating->get_n( ) TYPE 'I'.
  CATCH /aws1/cx_dynresourcenotfoundex.
    MESSAGE 'The table or index does not exist' TYPE 'E'.

```

```

ENDTRY.

" Scan items from table.
TRY.
  DATA(lo_scan_result) = lo_dyn->scan( iv_tablename = iv_table_name ).
  lt_items = lo_scan_result->get_items( ).
  " Read the first item and display the attributes.
  READ TABLE lo_query_result->get_items( ) INTO lt_item INDEX 1.
  lo_title = lt_item[ key = 'title' ]-value.
  lo_year = lt_item[ key = 'year' ]-value.
  lo_rating = lt_item[ key = 'rating' ]-value.
  MESSAGE 'Movie name is: ' && lo_title->get_s( ) TYPE 'I'.
  MESSAGE 'Movie year is: ' && lo_year->get_n( ) TYPE 'I'.
  MESSAGE 'Movie rating is: ' && lo_rating->get_n( ) TYPE 'I'.
CATCH /aws1/cx_dynresourcenotfoundex.
  MESSAGE 'The table or index does not exist' TYPE 'E'.
ENDTRY.

" Update items from table.
TRY.
  DATA(lt_attributeupdates) = VALUE /aws1/
cl_dynattrvalueupdate=>tt_attributeupdates(
  ( VALUE /aws1/cl_dynattrvalueupdate=>ts_attributeupdates_maprow(
    key = 'rating' value = NEW /aws1/cl_dynattrvalueupdate(
      io_value = NEW /aws1/cl_dynattributevalue( iv_n = '7.6' )
      iv_action = |PUT| ) ) ) ).
  DATA(lt_key) = VALUE /aws1/cl_dynattributevalue=>tt_key(
    ( VALUE /aws1/cl_dynattributevalue=>ts_key_maprow(
      key = 'year' value = NEW /aws1/cl_dynattributevalue( iv_n = '1975' ) ) )
    ( VALUE /aws1/cl_dynattributevalue=>ts_key_maprow(
      key = 'title' value = NEW /aws1/cl_dynattributevalue( iv_s =
'1980' ) ) ) ).
  DATA(lo_resp) = lo_dyn->updateitem(
    iv_tablename      = iv_table_name
    it_key            = lt_key
    it_attributeupdates = lt_attributeupdates ).
  MESSAGE '1 item updated in DynamoDB Table' && iv_table_name TYPE 'I'.
CATCH /aws1/cx_dyncondalcheckfaile00.
  MESSAGE 'A condition specified in the operation could not be evaluated.'
TYPE 'E'.
CATCH /aws1/cx_dynresourcenotfoundex.
  MESSAGE 'The table or index does not exist' TYPE 'E'.
CATCH /aws1/cx_dyntransactconflictex.
  MESSAGE 'Another transaction is using the item' TYPE 'E'.

```

```
ENDTRY.  
  
" Delete table.  
TRY.  
    lo_dyn->deletetable( iv_tablename = iv_table_name ).  
    lo_dyn->get_waiter( )->tablenotexists(  
        iv_max_wait_time = 200  
        iv_tablename     = iv_table_name ).  
    MESSAGE 'DynamoDB Table deleted.' TYPE 'I'.  
CATCH /aws1/cx_dynresourcenotfoundex.  
    MESSAGE 'The table or index does not exist' TYPE 'E'.  
CATCH /aws1/cx_dynresourceinuseex.  
    MESSAGE 'The table cannot be deleted as it is in use' TYPE 'E'.  
ENDTRY.
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK para SAP ABAP.
 - [BatchWriteItem](#)
 - [CreateTable](#)
 - [DeleteItem](#)
 - [DeleteTable](#)
 - [DescribeTable](#)
 - [GetItem](#)
 - [PutItem](#)
 - [Query](#)
 - [Scan](#)
 - [UpdateItem](#)

Exemplos do Amazon EC2 usando o SDK para SAP ABAP

Os exemplos de código a seguir mostram como realizar ações e implementar cenários comuns usando o AWS SDK para SAP ABAP com o Amazon EC2.

Ações são trechos de código de programas maiores e devem ser executadas em contexto.

Embora as ações mostrem como chamar funções de serviço específicas, é possível ver as ações contextualizadas em seus devidos cenários e exemplos entre serviços.

Cenários são exemplos de código que mostram como realizar uma tarefa específica chamando várias funções dentro do mesmo serviço.

Cada exemplo inclui um link para GitHub, onde você pode encontrar instruções sobre como configurar e executar o código no contexto.

Tópicos

- [Ações](#)

Ações

AllocateAddress

O código de exemplo a seguir mostra como usar `AllocateAddress`.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.  
    oo_result = lo_ec2->allocateaddress( iv_domain = 'vpc' ).    " oo_result is  
returned for testing purposes. "  
    MESSAGE 'Allocated an Elastic IP address.' TYPE 'I'.  
    CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).  
    DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-  
>av_err_msg }|.  
    MESSAGE lv_error TYPE 'E'.  
ENDTRY.
```

- Para obter detalhes da API, consulte a [AllocateAddress](#) referência da API AWS SDK for SAP ABAP.

AssociateAddress

O código de exemplo a seguir mostra como usar `AssociateAddress`.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.
    oo_result = lo_ec2->associateaddress(
        iv_allocationid = iv_allocation_id
        iv_instanceid = iv_instance_id
    ).
    MESSAGE 'Associated an Elastic IP address with an EC2 instance.' TYPE 'I'.
CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
    DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
    MESSAGE lv_error TYPE 'E'.
ENDTRY.
```

- Para obter detalhes da API, consulte a [AssociateAddress](#) referência da API AWS SDK for SAP ABAP.

CreateKeyPair

O código de exemplo a seguir mostra como usar CreateKeyPair.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.
    oo_result = lo_ec2->createkeypair( iv_keyname = iv_key_name ).
    " oo_result is returned for testing purposes. "
```

```

    MESSAGE 'Amazon EC2 key pair created.' TYPE 'I'.
    CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
    DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
    MESSAGE lv_error TYPE 'E'.
  ENTRY.

```

- Para obter detalhes da API, consulte a [CreateKeyPair](#) referência da API AWS SDK for SAP ABAP.

CreateSecurityGroup

O código de exemplo a seguir mostra como usar CreateSecurityGroup.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```

  TRY.
    oo_result = lo_ec2->createsecuritygroup(
      iv_description = 'Security group example'
      iv_groupname = iv_security_group_name
      iv_vpcid = iv_vpc_id
    ).
    MESSAGE 'Security group created.' TYPE 'I'.
  CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
  DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
  MESSAGE lv_error TYPE 'E'.
  ENTRY.

```

- Para obter detalhes da API, consulte a [CreateSecurityGroup](#) referência da API AWS SDK for SAP ABAP.

DeleteKeyPair

O código de exemplo a seguir mostra como usar DeleteKeyPair.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.  
    lo_ec2->deletekeypair( iv_keyname = iv_key_name ).  
    MESSAGE 'Amazon EC2 key pair deleted.' TYPE 'I'.  
    CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).  
    DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception->  
>av_err_msg }|.  
    MESSAGE lv_error TYPE 'E'.  
ENDTRY.
```

- Para obter detalhes da API, consulte a [DeleteKeyPair](#) referência da API AWS SDK for SAP ABAP.

DeleteSecurityGroup

O código de exemplo a seguir mostra como usar DeleteSecurityGroup.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.  
    lo_ec2->deletesecuritygroup( iv_groupid = iv_security_group_id ).  
    MESSAGE 'Security group deleted.' TYPE 'I'.  
ENDTRY.
```

```

    CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
      DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
      MESSAGE lv_error TYPE 'E'.
    ENDTRY.

```

- Para obter detalhes da API, consulte a [DeleteSecurityGroup](#) referência da API AWS SDK for SAP ABAP.

DescribeAddresses

O código de exemplo a seguir mostra como usar DescribeAddresses.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```

    TRY.
      oo_result = lo_ec2->describeaddresses( ) .
      oo_result is returned for testing purposes. "
      DATA(lt_addresses) = oo_result->get_addresses( ).
      MESSAGE 'Retrieved information about Elastic IP addresses.' TYPE 'I'.
    CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
      DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
      MESSAGE lv_error TYPE 'E'.
    ENDTRY.

```

- Para obter detalhes da API, consulte a [DescribeAddresses](#) referência da API AWS SDK for SAP ABAP.

DescribeAvailabilityZones

O código de exemplo a seguir mostra como usar DescribeAvailabilityZones.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```

TRY.
    oo_result = lo_ec2->describeavailabilityzones( ) .
oo_result is returned for testing purposes. "
    DATA(lt_zones) = oo_result->get_availabilityzones( ).
    MESSAGE 'Retrieved information about Availability Zones.' TYPE 'I'.

    CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
    DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
    MESSAGE lv_error TYPE 'E'.
ENDTRY.

```

- Para obter detalhes da API, consulte a [DescribeAvailabilityZones](#) referência da API AWS SDK for SAP ABAP.

DescribeInstances

O código de exemplo a seguir mostra como usar DescribeInstances.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```

TRY.
    oo_result = lo_ec2->describeinstances( ) .
oo_result is returned for testing purposes. "

```

```

" Retrieving details of EC2 instances. "
DATA: lv_instance_id    TYPE /aws1/ec2string,
      lv_status         TYPE /aws1/ec2instancename,
      lv_instance_type  TYPE /aws1/ec2instancetype,
      lv_image_id      TYPE /aws1/ec2string.
LOOP AT oo_result->get_reservations( ) INTO DATA(lo_reservation).
  LOOP AT lo_reservation->get_instances( ) INTO DATA(lo_instance).
    lv_instance_id = lo_instance->get_instanceid( ).
    lv_status = lo_instance->get_state( )->get_name( ).
    lv_instance_type = lo_instance->get_instancetype( ).
    lv_image_id = lo_instance->get_imageid( ).
  ENDLLOOP.
ENDLOOP.
MESSAGE 'Retrieved information about EC2 instances.' TYPE 'I'.
CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
  DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
  MESSAGE lv_error TYPE 'E'.
ENDTRY.

```

- Para obter detalhes da API, consulte a [DescribeInstances](#) referência da API AWS SDK for SAP ABAP.

DescribeKeyPairs

O código de exemplo a seguir mostra como usar DescribeKeyPairs.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```

TRY.
  oo_result = lo_ec2->describekeypairs( ) .
  " oo_result
is returned for testing purposes. "
  DATA(lt_key_pairs) = oo_result->get_keypairs( ).

```

```
MESSAGE 'Retrieved information about key pairs.' TYPE 'I'.
CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
MESSAGE lv_error TYPE 'E'.
ENDTRY.
```

- Para obter detalhes da API, consulte a [DescribeKeyPairs](#) referência da API AWS SDK for SAP ABAP.

DescribeRegions

O código de exemplo a seguir mostra como usar DescribeRegions.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.
    oo_result = lo_ec2->describeregions( ) .                                " oo_result
is returned for testing purposes. "
    DATA(lt_regions) = oo_result->get_regions( ).
    MESSAGE 'Retrieved information about Regions.' TYPE 'I'.
    CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
    DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
    MESSAGE lv_error TYPE 'E'.
ENDTRY.
```

- Para obter detalhes da API, consulte a [DescribeRegions](#) referência da API AWS SDK for SAP ABAP.

DescribeSecurityGroups

O código de exemplo a seguir mostra como usar DescribeSecurityGroups.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.  
  DATA lt_group_ids TYPE /aws1/cl_ec2groupidstrlist_w=>tt_groupidstringlist.  
  APPEND NEW /aws1/cl_ec2groupidstrlist_w( iv_value = iv_group_id ) TO  
lt_group_ids.  
  oo_result = lo_ec2->describesecuritygroups( it_groupids = lt_group_ids ).  
  " oo_result is returned for testing purposes. "  
  DATA(lt_security_groups) = oo_result->get_securitygroups( ).  
  MESSAGE 'Retrieved information about security groups.' TYPE 'I'.  
  CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).  
  DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-  
>av_err_msg }|.  
  MESSAGE lv_error TYPE 'E'.  
ENDTRY.
```

- Para obter detalhes da API, consulte a [DescribeSecurityGroups](#) referência da API AWS SDK for SAP ABAP.

MonitorInstances

O código de exemplo a seguir mostra como usar MonitorInstances.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```

    DATA lt_instance_ids TYPE /aws1/
    cl_ec2instidstringlist_w=>tt_instanceidstringlist.
    APPEND NEW /aws1/cl_ec2instidstringlist_w( iv_value = iv_instance_id ) TO
    lt_instance_ids.

    "Perform dry run"
    TRY.
        " DryRun is set to true. This checks for the required permissions to monitor
        the instance without actually making the request. "
        lo_ec2->monitorinstances(
            it_instanceids = lt_instance_ids
            iv_dryrun = abap_true
        ).
    CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
        " If the error code returned is `DryRunOperation`, then you have the
        required permissions to monitor this instance. "
        IF lo_exception->av_err_code = 'DryRunOperation'.
            MESSAGE 'Dry run to enable detailed monitoring completed.' TYPE 'I'.
            " DryRun is set to false to enable detailed monitoring. "
            lo_ec2->monitorinstances(
                it_instanceids = lt_instance_ids
                iv_dryrun = abap_false
            ).
            MESSAGE 'Detailed monitoring enabled.' TYPE 'I'.
        " If the error code returned is `UnauthorizedOperation`, then you don't
        have the required permissions to monitor this instance. "
        ELSEIF lo_exception->av_err_code = 'UnauthorizedOperation'.
            MESSAGE 'Dry run to enable detailed monitoring failed. User does not have
            the permissions to monitor the instance.' TYPE 'E'.
        ELSE.
            DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
            >av_err_msg }|.
            MESSAGE lv_error TYPE 'E'.
        ENDIF.
    ENDTRY.

```

- Para obter detalhes da API, consulte a [MonitorInstances](#) referência da API AWS SDK for SAP ABAP.

RebootInstances

O código de exemplo a seguir mostra como usar RebootInstances.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
DATA lt_instance_ids TYPE /aws1/
cl_ec2instidstringlist_w=>tt_instanceidstringlist.
  APPEND NEW /aws1/cl_ec2instidstringlist_w( iv_value = iv_instance_id ) TO
  lt_instance_ids.

  "Perform dry run"
  TRY.
    " DryRun is set to true. This checks for the required permissions to reboot
    the instance without actually making the request. "
    lo_ec2->rebootinstances(
      it_instanceids = lt_instance_ids
      iv_dryrun = abap_true
    ).
  CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
    " If the error code returned is `DryRunOperation`, then you have the
    required permissions to reboot this instance. "
    IF lo_exception->av_err_code = 'DryRunOperation'.
      MESSAGE 'Dry run to reboot instance completed.' TYPE 'I'.
      " DryRun is set to false to make a reboot request. "
      lo_ec2->rebootinstances(
        it_instanceids = lt_instance_ids
        iv_dryrun = abap_false
      ).
      MESSAGE 'Instance rebooted.' TYPE 'I'.
    " If the error code returned is `UnauthorizedOperation`, then you don't
    have the required permissions to reboot this instance. "
    ELSEIF lo_exception->av_err_code = 'UnauthorizedOperation'.
      MESSAGE 'Dry run to reboot instance failed. User does not have permissions
      to reboot the instance.' TYPE 'E'.
    ELSE.
```

```
DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-  
>av_err_msg }|.
MESSAGE lv_error TYPE 'E'.
ENDIF.
ENDTRY.
```

- Para obter detalhes da API, consulte a [RebootInstances](#) referência da API AWS SDK for SAP ABAP.

ReleaseAddress

O código de exemplo a seguir mostra como usar ReleaseAddress.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).


```
TRY.
    lo_ec2->releaseaddress( iv_allocationid = iv_allocation_id ).
    MESSAGE 'Elastic IP address released.' TYPE 'I'.
    CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
    DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-  
>av_err_msg }|.
    MESSAGE lv_error TYPE 'E'.
ENDTRY.
```

- Para obter detalhes da API, consulte a [ReleaseAddress](#) referência da API AWS SDK for SAP ABAP.

RunInstances

O código de exemplo a seguir mostra como usar RunInstances.

SDK para SAP ABAP

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```

" Create tags for resource created during instance launch. "
DATA lt_tagsspecifications TYPE /aws1/
cl_ec2tagsspecification=>tt_tagsspecificationlist.
DATA ls_tagsspecifications LIKE LINE OF lt_tagsspecifications.
ls_tagsspecifications = NEW /aws1/cl_ec2tagsspecification(
  iv_resourcetype = 'instance'
  it_tags = VALUE /aws1/cl_ec2tag=>tt_taglist(
    ( NEW /aws1/cl_ec2tag( iv_key = 'Name' iv_value = iv_tag_value ) )
  )
).
APPEND ls_tagsspecifications TO lt_tagsspecifications.

TRY.
  " Create/launch Amazon Elastic Compute Cloud (Amazon EC2) instance. "
  oo_result = lo_ec2->runinstances(                                " oo_result is
returned for testing purposes. "
  iv_imageid = iv_ami_id
  iv_instancetype = 't2.micro'
  iv_maxcount = 1
  iv_mincount = 1
  it_tagsspecifications = lt_tagsspecifications
  iv_subnetid = iv_subnet_id
).
MESSAGE 'EC2 instance created.' TYPE 'I'.
CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
MESSAGE lv_error TYPE 'E'.
ENDTRY.

```

- Para obter detalhes da API, consulte a [RunInstances](#) referência da API AWS SDK for SAP ABAP.

StartInstances

O código de exemplo a seguir mostra como usar StartInstances.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
DATA lt_instance_ids TYPE /aws1/
cl_ec2instidstringlist_w=>tt_instanceidstringlist.
  APPEND NEW /aws1/cl_ec2instidstringlist_w( iv_value = iv_instance_id ) TO
  lt_instance_ids.

  "Perform dry run"
  TRY.
    " DryRun is set to true. This checks for the required permissions to start
    the instance without actually making the request. "
    lo_ec2->startinstances(
      it_instanceids = lt_instance_ids
      iv_dryrun = abap_true
    ).
  CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
    " If the error code returned is `DryRunOperation`, then you have the
    required permissions to start this instance. "
    IF lo_exception->av_err_code = 'DryRunOperation'.
      MESSAGE 'Dry run to start instance completed.' TYPE 'I'.
      " DryRun is set to false to start instance. "
      oo_result = lo_ec2->startinstances(          " oo_result is returned for
      testing purposes. "
        it_instanceids = lt_instance_ids
        iv_dryrun = abap_false
      ).
      MESSAGE 'Successfully started the EC2 instance.' TYPE 'I'.
      " If the error code returned is `UnauthorizedOperation`, then you don't
      have the required permissions to start this instance. "
      ELSEIF lo_exception->av_err_code = 'UnauthorizedOperation'.
        MESSAGE 'Dry run to start instance failed. User does not have permissions
        to start the instance.' TYPE 'E'.
```

```

ELSE.
    DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
    MESSAGE lv_error TYPE 'E'.
ENDIF.
ENDTRY.

```

- Para obter detalhes da API, consulte a [StartInstances](#) referência da API AWS SDK for SAP ABAP.

StopInstances

O código de exemplo a seguir mostra como usar StopInstances.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```

DATA lt_instance_ids TYPE /aws1/
cl_ec2instidstringlist_w=>tt_instanceidstringlist.
APPEND NEW /aws1/cl_ec2instidstringlist_w( iv_value = iv_instance_id ) TO
lt_instance_ids.

"Perform dry run"
TRY.
    " DryRun is set to true. This checks for the required permissions to stop
the instance without actually making the request. "
    lo_ec2->stopinstances(
        it_instanceids = lt_instance_ids
        iv_dryrun = abap_true
    ).
CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
    " If the error code returned is `DryRunOperation`, then you have the
required permissions to stop this instance. "
    IF lo_exception->av_err_code = 'DryRunOperation'.

```

```
MESSAGE 'Dry run to stop instance completed.' TYPE 'I'.
" DryRun is set to false to stop instance. "
oo_result = lo_ec2->stopinstances(          " oo_result is returned for
testing purposes. "
    it_instanceids = lt_instance_ids
    iv_dryrun = abap_false
).
MESSAGE 'Successfully stopped the EC2 instance.' TYPE 'I'.
" If the error code returned is `UnauthorizedOperation`, then you don't
have the required permissions to stop this instance. "
ELSEIF lo_exception->av_err_code = 'UnauthorizedOperation'.
MESSAGE 'Dry run to stop instance failed. User does not have permissions
to stop the instance.' TYPE 'E'.
ELSE.
DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
MESSAGE lv_error TYPE 'E'.
ENDIF.
ENDTRY.
```

- Para obter detalhes da API, consulte a [StopInstances](#) referência da API AWS SDK for SAP ABAP.

Exemplos do Kinesis usando o SDK para SAP ABAP

Os exemplos de código a seguir mostram como realizar ações e implementar cenários comuns usando o AWS SDK para SAP ABAP com Kinesis.

Ações são trechos de código de programas maiores e devem ser executadas em contexto. Embora as ações mostrem como chamar funções de serviço específicas, é possível ver as ações contextualizadas em seus devidos cenários e exemplos entre serviços.

Cenários são exemplos de código que mostram como realizar uma tarefa específica chamando várias funções dentro do mesmo serviço.

Cada exemplo inclui um link para GitHub, onde você pode encontrar instruções sobre como configurar e executar o código no contexto.

Tópicos

- [Ações](#)


- [Cenários](#)

Ações

CreateStream

O código de exemplo a seguir mostra como usar CreateStream.

SDK para SAP ABAP

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.  
    lo_kns->createstream(  
        iv_streamname = iv_stream_name  
        iv_shardcount = iv_shard_count  
    ).  
    MESSAGE 'Stream created.' TYPE 'I'.  
CATCH /aws1/cx_knsinvalidargumentex.  
    MESSAGE 'The specified argument was not valid.' TYPE 'E'.  
CATCH /aws1/cx_knslimitexceeddex .  
    MESSAGE 'The request processing has failed because of a limit exceed  
exception.' TYPE 'E'.  
CATCH /aws1/cx_knsresourceinuseex .  
    MESSAGE 'The request processing has failed because the resource is in use.'  
TYPE 'E'.  
ENDTRY.
```

- Para obter detalhes da API, consulte a [CreateStream](#) referência da API AWS SDK for SAP ABAP.

DeleteStream

O código de exemplo a seguir mostra como usar DeleteStream.

SDK para SAP ABAP

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).


```
TRY.  
    lo_kns->deletestream(  
        iv_streamname = iv_stream_name  
    ).  
    MESSAGE 'Stream deleted.' TYPE 'I'.  
CATCH /aws1/cx_knslimitexceeddex .  
    MESSAGE 'The request processing has failed because of a limit exceed  
exception.' TYPE 'E'.  
CATCH /aws1/cx_knsresourceinuseex .  
    MESSAGE 'The request processing has failed because the resource is in use.'  
TYPE 'E'.  
ENDTRY.
```

- Para obter detalhes da API, consulte a [DeleteStream](#) referência da API AWS SDK for SAP ABAP.

DescribeStream

O código de exemplo a seguir mostra como usar DescribeStream.

SDK para SAP ABAP

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.  
    oo_result = lo_kns->describestream(  

```

```

        iv_streamname = iv_stream_name
    ).
    DATA(lt_stream_description) = oo_result->get_streamdescription( ).
    MESSAGE 'Streams retrieved.' TYPE 'I'.
    CATCH /aws1/cx_knslimitexceedex .
        MESSAGE 'The request processing has failed because of a limit exceed
exception.' TYPE 'E'.
    CATCH /aws1/cx_knsresourcenotfoundex .
        MESSAGE 'Resource being accessed is not found.' TYPE 'E'.
    ENDRTRY.

```

- Para obter detalhes da API, consulte a [DescribeStream](#) referência da API AWS SDK for SAP ABAP.

GetRecords

O código de exemplo a seguir mostra como usar GetRecords.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```

    TRY.
        oo_result = lo_kns->getrecords(           " oo_result is returned for
testing purposes. "
        iv_sharditerator = iv_shard_iterator
    ).
    DATA(lt_records) = oo_result->get_records( ).
    MESSAGE 'Record retrieved.' TYPE 'I'.
    CATCH /aws1/cx_knsexpirediteratorex .
        MESSAGE 'Iterator expired.' TYPE 'E'.
    CATCH /aws1/cx_knsinvalidargumentex .
        MESSAGE 'The specified argument was not valid.' TYPE 'E'.
    CATCH /aws1/cx_knskmsaccessdeniedex .
        MESSAGE 'You do not have permission to perform this AWS KMS action.' TYPE
'E'.

```

```

CATCH /aws1/cx_knskmsdisabledex .
    MESSAGE 'KMS key used is disabled.' TYPE 'E'.
CATCH /aws1/cx_knskmsinvalidstateex .
    MESSAGE 'KMS key used is in an invalid state. ' TYPE 'E'.
CATCH /aws1/cx_knskmsnotfoundex .
    MESSAGE 'KMS key used is not found.' TYPE 'E'.
CATCH /aws1/cx_knskmsoptinrequired .
    MESSAGE 'KMS key option is required.' TYPE 'E'.
CATCH /aws1/cx_knskmsstrottingex .
    MESSAGE 'The rate of requests to AWS KMS is exceeding the request quotas.'
TYPE 'E'.
CATCH /aws1/cx_knsprovthruputexcdex .
    MESSAGE 'The request rate for the stream is too high, or the requested data
is too large for the available throughput.' TYPE 'E'.
CATCH /aws1/cx_knsresourcenotfoundex .
    MESSAGE 'Resource being accessed is not found.' TYPE 'E'.
ENDTRY.

```

- Para obter detalhes da API, consulte a [GetRecords](#) referência da API AWS SDK for SAP ABAP.

ListStreams

O código de exemplo a seguir mostra como usar ListStreams.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```

TRY.
    oo_result = lo_kns->liststreams(          " oo_result is returned for testing
purposes. "
        "Set Limit to specify that a maximum of streams should be returned."
        iv_limit = iv_limit
    ).
    DATA(lt_streams) = oo_result->get_streamnames( ).
    MESSAGE 'Streams listed.' TYPE 'I'.

```

```

    CATCH /aws1/cx_knslimitexceedex .
      MESSAGE 'The request processing has failed because of a limit exceed
exception.' TYPE 'E'.
    ENDTRY.

```

- Para obter detalhes da API, consulte a [ListStreams](#) referência da API AWS SDK for SAP ABAP.

PutRecord

O código de exemplo a seguir mostra como usar PutRecord.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```

TRY.
  oo_result = lo_kns->putrecord(           " oo_result is returned for
testing purposes. "
    iv_streamname = iv_stream_name
    iv_data       = iv_data
    iv_partitionkey = iv_partition_key
  ).
  MESSAGE 'Record created.' TYPE 'I'.
  CATCH /aws1/cx_knsinvalidargumentex .
    MESSAGE 'The specified argument was not valid.' TYPE 'E'.
  CATCH /aws1/cx_knskmsaccessdeniedex .
    MESSAGE 'You do not have permission to perform this AWS KMS action.' TYPE
'E'.
  CATCH /aws1/cx_knskmsdisabledex .
    MESSAGE 'KMS key used is disabled.' TYPE 'E'.
  CATCH /aws1/cx_knskmsinvalidstateex .
    MESSAGE 'KMS key used is in an invalid state. ' TYPE 'E'.
  CATCH /aws1/cx_knskmsnotfoundex .
    MESSAGE 'KMS key used is not found.' TYPE 'E'.
  CATCH /aws1/cx_knskmsoptinrequired .
    MESSAGE 'KMS key option is required.' TYPE 'E'.
  CATCH /aws1/cx_knskms throttlingex .

```

```

    MESSAGE 'The rate of requests to AWS KMS is exceeding the request quotas.'
TYPE 'E'.
    CATCH /aws1/cx_knsprovthruputexcdex .
    MESSAGE 'The request rate for the stream is too high, or the requested data
is too large for the available throughput.' TYPE 'E'.
    CATCH /aws1/cx_knsresourcenotfoundex .
    MESSAGE 'Resource being accessed is not found.' TYPE 'E'.
ENDTRY.

```

- Para obter detalhes da API, consulte a [PutRecord](#) referência da API AWS SDK for SAP ABAP.

RegisterStreamConsumer

O código de exemplo a seguir mostra como usar RegisterStreamConsumer.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```

TRY.
    oo_result = lo_kns->registerstreamconsumer(      " oo_result is returned
for testing purposes. "
    iv_streamarn = iv_stream_arn
    iv_consumername = iv_consumer_name
    ).
    MESSAGE 'Stream consumer registered.' TYPE 'I'.
    CATCH /aws1/cx_knsinvalidargumentex .
    MESSAGE 'The specified argument was not valid.' TYPE 'E'.
    CATCH /aws1/cx_sgmresourcecelimitexcd.
    MESSAGE 'You have reached the limit on the number of resources.' TYPE 'E'.
    CATCH /aws1/cx_sgmresourceinuse.
    MESSAGE 'Resource being accessed is in use.' TYPE 'E'.
    CATCH /aws1/cx_sgmresourcenotfound.
    MESSAGE 'Resource being accessed is not found.' TYPE 'E'.
ENDTRY.

```

- Para obter detalhes da API, consulte a [RegisterStreamConsumer](#) referência da API AWS SDK for SAP ABAP.

Cenários

Conceitos básicos de fluxos de dados

O exemplo de código a seguir mostra como:

- Criar um stream e inserir um registro nele.
- Criar um iterador de fragmento.
- Ler o registro e, em seguida, limpar os recursos.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
DATA lo_stream_describe_result TYPE REF TO /aws1/cl_knsdescrstreamoutput.
DATA lo_stream_description TYPE REF TO /aws1/cl_knsstreamdescription.
DATA lo_sharditerator TYPE REF TO /aws1/cl_knsgetsharditerator01.
DATA lo_record_result TYPE REF TO /aws1/cl_knsputrecordoutput.

"Create stream."
TRY.
    lo_kns->createstream(
        iv_streamname = iv_stream_name
        iv_shardcount = iv_shard_count
    ).
    MESSAGE 'Stream created.' TYPE 'I'.
CATCH /aws1/cx_knsinvalidargumentex.
    MESSAGE 'The specified argument was not valid.' TYPE 'E'.
CATCH /aws1/cx_knslimitexceededx .
    MESSAGE 'The request processing has failed because of a limit exceeded
exception.' TYPE 'E'.
CATCH /aws1/cx_knsresourceinuseex .
```

```
    MESSAGE 'The request processing has failed because the resource is in use.'
TYPE 'E'.
    ENDRTRY.

    "Wait for stream to becomes active."
    lo_stream_describe_result = lo_kns->describestream( iv_streamname =
iv_stream_name ).
    lo_stream_description = lo_stream_describe_result->get_streamdescription( ).
    WHILE lo_stream_description->get_streamstatus( ) <> 'ACTIVE'.
        IF sy-index = 30.
            EXIT.                "maximum 5 minutes"
        ENDIF.
        WAIT UP TO 10 SECONDS.
        lo_stream_describe_result = lo_kns->describestream( iv_streamname =
iv_stream_name ).
        lo_stream_description = lo_stream_describe_result->get_streamdescription( ).
    ENDWHILE.

    "Create record."
    TRY.
        lo_record_result = lo_kns->putrecord(
            iv_streamname = iv_stream_name
            iv_data        = iv_data
            iv_partitionkey = iv_partition_key
        ).
        MESSAGE 'Record created.' TYPE 'I'.
    CATCH /aws1/cx_knsinvalidargumentex .
        MESSAGE 'The specified argument was not valid.' TYPE 'E'.
    CATCH /aws1/cx_knskmsaccessdeniedex .
        MESSAGE 'You do not have permission to perform this AWS KMS action.' TYPE
'E'.
    CATCH /aws1/cx_knskmsdisabledex .
        MESSAGE 'KMS key used is disabled.' TYPE 'E'.
    CATCH /aws1/cx_knskmsinvalidstateex .
        MESSAGE 'KMS key used is in an invalid state. ' TYPE 'E'.
    CATCH /aws1/cx_knskmsnotfoundex .
        MESSAGE 'KMS key used is not found.' TYPE 'E'.
    CATCH /aws1/cx_knskmsoptinrequired .
        MESSAGE 'KMS key option is required.' TYPE 'E'.
    CATCH /aws1/cx_knskmsstrottlingex .
        MESSAGE 'The rate of requests to AWS KMS is exceeding the request quotas.'
TYPE 'E'.
    CATCH /aws1/cx_knsprovthruputexcdex .
```



```

    MESSAGE 'The request rate for the stream is too high, or the requested data
is too large for the available throughput.' TYPE 'E'.
    CATCH /aws1/cx_knsresourcenotfoundex .
    MESSAGE 'Resource being accessed is not found.' TYPE 'E'.
ENDTRY.

"Create a shard iterator in order to read the record."
TRY.
    lo_sharditerator = lo_kns->getsharditerator(
        iv_shardid = lo_record_result->get_shardid( )
        iv_sharditeratortype = iv_sharditeratortype
        iv_streamname = iv_stream_name
    ).
    MESSAGE 'Shard iterator created.' TYPE 'I'.
    CATCH /aws1/cx_knsinvalidargumentex.
    MESSAGE 'The specified argument was not valid.' TYPE 'E'.
    CATCH /aws1/cx_knsprovthruputexcdex .
    MESSAGE 'The request rate for the stream is too high, or the requested data
is too large for the available throughput.' TYPE 'E'.
    CATCH /aws1/cx_sgmresourcenotfound.
    MESSAGE 'Resource being accessed is not found.' TYPE 'E'.
ENDTRY.

"Read the record."
TRY.
    oo_result = lo_kns->getrecords( " oo_result is returned
for testing purposes. "
        iv_sharditerator = lo_sharditerator->get_sharditerator( )
    ).
    MESSAGE 'Shard iterator created.' TYPE 'I'.
    CATCH /aws1/cx_knsexpirediteratorex .
    MESSAGE 'Iterator expired.' TYPE 'E'.
    CATCH /aws1/cx_knsinvalidargumentex .
    MESSAGE 'The specified argument was not valid.' TYPE 'E'.
    CATCH /aws1/cx_knskmsaccessdeniedex .
    MESSAGE 'You do not have permission to perform this AWS KMS action.' TYPE
'E'.
    CATCH /aws1/cx_knskmsdisabledex .
    MESSAGE 'KMS key used is disabled.' TYPE 'E'.
    CATCH /aws1/cx_knskmsinvalidstateex .
    MESSAGE 'KMS key used is in an invalid state. ' TYPE 'E'.
    CATCH /aws1/cx_knskmsnotfoundex .
    MESSAGE 'KMS key used is not found.' TYPE 'E'.
    CATCH /aws1/cx_knskmsoptinrequired .

```

```
    MESSAGE 'KMS key option is required.' TYPE 'E'.
  CATCH /aws1/cx_knsksthrottlingex .
    MESSAGE 'The rate of requests to AWS KMS is exceeding the request quotas.'
  TYPE 'E'.
  CATCH /aws1/cx_knsprovthruputexcdex .
    MESSAGE 'The request rate for the stream is too high, or the requested data
  is too large for the available throughput.' TYPE 'E'.
  CATCH /aws1/cx_knsresourcenotfoundex .
    MESSAGE 'Resource being accessed is not found.' TYPE 'E'.
  ENDRY.

  "Delete stream."
  TRY.
    lo_kns->deletestream(
      iv_streamname = iv_stream_name
    ).
    MESSAGE 'Stream deleted.' TYPE 'I'.
  CATCH /aws1/cx_knslimitexceeddex .
    MESSAGE 'The request processing has failed because of a limit exceeded
  exception.' TYPE 'E'.
  CATCH /aws1/cx_knsresourceinuseex .
    MESSAGE 'The request processing has failed because the resource is in use.'
  TYPE 'E'.
  ENDRY.
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK para SAP ABAP.
 - [CreateStream](#)
 - [DeleteStream](#)
 - [GetRecords](#)
 - [GetShardIterator](#)
 - [PutRecord](#)

Exemplos do Lambda usando o SDK para SAP ABAP

Os exemplos de código a seguir mostram como realizar ações e implementar cenários comuns usando o AWS SDK para SAP ABAP com Lambda.

Ações são trechos de código de programas maiores e devem ser executadas em contexto. Embora as ações mostrem como chamar funções de serviço específicas, é possível ver as ações contextualizadas em seus devidos cenários e exemplos entre serviços.

Cenários são exemplos de código que mostram como realizar uma tarefa específica chamando várias funções dentro do mesmo serviço.

Cada exemplo inclui um link para GitHub, onde você pode encontrar instruções sobre como configurar e executar o código no contexto.

Tópicos

- [Ações](#)
- [Cenários](#)

Ações

CreateFunction

O código de exemplo a seguir mostra como usar CreateFunction.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.  
  lo_lmd->createfunction(  
    iv_functionname = iv_function_name  
    iv_runtime = `python3.9`  
    iv_role = iv_role_arn  
    iv_handler = iv_handler  
    io_code = io_zip_file  
    iv_description = 'AWS Lambda code example'  
  ).  
  MESSAGE 'Lambda function created.' TYPE 'I'.  
CATCH /aws1/cx_lmdcodesigningcfgno00.
```

```
    MESSAGE 'Code signing configuration does not exist.' TYPE 'E'.
  CATCH /aws1/cx_lmdcodestorageexclex.
    MESSAGE 'Maximum total code size per account exceeded.' TYPE 'E'.
  CATCH /aws1/cx_lmdcodeverification00.
    MESSAGE 'Code signature failed one or more validation checks for signature
mismatch or expiration.' TYPE 'E'.
  CATCH /aws1/cx_lmdinvalidcodesigex.
    MESSAGE 'Code signature failed the integrity check.' TYPE 'E'.
  CATCH /aws1/cx_lmdinvparamvalueex.
    MESSAGE 'The request contains a non-valid parameter.' TYPE 'E'.
  CATCH /aws1/cx_lmdresourceconflictex.
    MESSAGE 'Resource already exists or another operation is in progress.' TYPE
'E'.
  CATCH /aws1/cx_lmdresourcenotfoundex.
    MESSAGE 'The requested resource does not exist.' TYPE 'E'.
  CATCH /aws1/cx_lmdserviceexception.
    MESSAGE 'An internal problem was encountered by the AWS Lambda service.'
TYPE 'E'.
  CATCH /aws1/cx_lmdtoomanyrequestsex.
    MESSAGE 'The maximum request throughput was reached.' TYPE 'E'.
  ENDTRY.
```

- Para obter detalhes da API, consulte a [CreateFunction](#) referência da API AWS SDK for SAP ABAP.

DeleteFunction

O código de exemplo a seguir mostra como usar DeleteFunction.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.
  lo_lmd->deletefunction( iv_functionname = iv_function_name ).
  MESSAGE 'Lambda function deleted.' TYPE 'I'.
```

```

CATCH /aws1/cx_lmdinvparamvalueex.
    MESSAGE 'The request contains a non-valid parameter.' TYPE 'E'.
CATCH /aws1/cx_lmdresourceconflictex.
    MESSAGE 'Resource already exists or another operation is in progress.' TYPE
'E'.
CATCH /aws1/cx_lmdresourcenotfoundex.
    MESSAGE 'The requested resource does not exist.' TYPE 'E'.
CATCH /aws1/cx_lmdserviceexception.
    MESSAGE 'An internal problem was encountered by the AWS Lambda service.'
TYPE 'E'.
CATCH /aws1/cx_lmdtoomanyrequestsex.
    MESSAGE 'The maximum request throughput was reached.' TYPE 'E'.
ENDTRY.

```

- Para obter detalhes da API, consulte a [DeleteFunction](#) referência da API AWS SDK for SAP ABAP.

GetFunction

O código de exemplo a seguir mostra como usar GetFunction.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```

TRY.
    oo_result = lo_lmd->getfunction( iv_functionname = iv_function_name ).
" oo_result is returned for testing purposes. "
    MESSAGE 'Lambda function information retrieved.' TYPE 'I'.
CATCH /aws1/cx_lmdinvparamvalueex.
    MESSAGE 'The request contains a non-valid parameter.' TYPE 'E'.
CATCH /aws1/cx_lmdserviceexception.
    MESSAGE 'An internal problem was encountered by the AWS Lambda service.'
TYPE 'E'.
CATCH /aws1/cx_lmdtoomanyrequestsex.
    MESSAGE 'The maximum request throughput was reached.' TYPE 'E'.

```

```
ENDTRY.
```

- Para obter detalhes da API, consulte a [GetFunction](#) referência da API AWS SDK for SAP ABAP.

Invoke

O código de exemplo a seguir mostra como usar Invoke.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.
  DATA(lv_json) = /aws1/cl_rt_util=>string_to_xstring(
    `{` &&
    `  "action": "increment",` &&
    `  "number": 10` &&
    `}`
  ).
  oo_result = lo_lmd->invoke(
    " oo_result is returned for
testing purposes. "
    iv_functionname = iv_function_name
    iv_payload = lv_json
  ).
  MESSAGE 'Lambda function invoked.' TYPE 'I'.
CATCH /aws1/cx_lmdinvparamvalueex.
  MESSAGE 'The request contains a non-valid parameter.' TYPE 'E'.
CATCH /aws1/cx_lmdinvrequestctx.
  MESSAGE 'Unable to parse request body as JSON.' TYPE 'E'.
CATCH /aws1/cx_lmdinvalidzipfileex.
  MESSAGE 'The deployment package could not be unzipped.' TYPE 'E'.
CATCH /aws1/cx_lmdrequesttoolargeex.
  MESSAGE 'Invoke request body JSON input limit was exceeded by the request
payload.' TYPE 'E'.
CATCH /aws1/cx_lmdresourceconflictex.
  MESSAGE 'Resource already exists or another operation is in progress.' TYPE
'E'.
```

```

CATCH /aws1/cx_lmdresourceindex.
  MESSAGE 'The requested resource does not exist.' TYPE 'E'.
CATCH /aws1/cx_lmdserviceexception.
  MESSAGE 'An internal problem was encountered by the AWS Lambda service.'
TYPE 'E'.
CATCH /aws1/cx_lmdtoomanyrequestsex.
  MESSAGE 'The maximum request throughput was reached.' TYPE 'E'.
CATCH /aws1/cx_lmdunsuppedmediatyp00.
  MESSAGE 'Invoke request body does not have JSON as its content type.' TYPE
'E'.
ENDTRY.

```

- Para obter os detalhes da API, consulte [Invoke](#) na Referência da API AWS SDK para SAP ABAP.

ListFunctions

O código de exemplo a seguir mostra como usar ListFunctions.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```

TRY.
  oo_result = lo_lmd->listfunctions( ).      " oo_result is returned for
testing purposes. "
  DATA(lt_functions) = oo_result->get_functions( ).
  MESSAGE 'Retrieved list of Lambda functions.' TYPE 'I'.
CATCH /aws1/cx_lmdinvparamvalueex.
  MESSAGE 'The request contains a non-valid parameter.' TYPE 'E'.
CATCH /aws1/cx_lmdserviceexception.
  MESSAGE 'An internal problem was encountered by the AWS Lambda service.'
TYPE 'E'.
CATCH /aws1/cx_lmdtoomanyrequestsex.
  MESSAGE 'The maximum request throughput was reached.' TYPE 'E'.
ENDTRY.

```

- Para obter detalhes da API, consulte a [ListFunctions](#) referência da API AWS SDK for SAP ABAP.

UpdateFunctionCode

O código de exemplo a seguir mostra como usar UpdateFunctionCode.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.
    oo_result = lo_lmd->updatefunctioncode(      " oo_result is returned for
testing purposes. "
        iv_functionname = iv_function_name
        iv_zipfile = io_zip_file
    ).

    MESSAGE 'Lambda function code updated.' TYPE 'I'.
CATCH /aws1/cx_lmdcodesigningcfgno00.
    MESSAGE 'Code signing configuration does not exist.' TYPE 'E'.
CATCH /aws1/cx_lmdcodestorageexc dex.
    MESSAGE 'Maximum total code size per account exceeded.' TYPE 'E'.
CATCH /aws1/cx_lmdcodeverification00.
    MESSAGE 'Code signature failed one or more validation checks for signature
mismatch or expiration.' TYPE 'E'.
CATCH /aws1/cx_lmdinvalidcodesigex.
    MESSAGE 'Code signature failed the integrity check.' TYPE 'E'.
CATCH /aws1/cx_lmdinvparamvalueex.
    MESSAGE 'The request contains a non-valid parameter.' TYPE 'E'.
CATCH /aws1/cx_lmdresourceconflictex.
    MESSAGE 'Resource already exists or another operation is in progress.' TYPE
'E'.
CATCH /aws1/cx_lmdresourcenotfoundex.
    MESSAGE 'The requested resource does not exist.' TYPE 'E'.
```



```

CATCH /aws1/cx_lmdserviceexception.
    MESSAGE 'An internal problem was encountered by the AWS Lambda service.'
TYPE 'E'.
CATCH /aws1/cx_lmdtoomanyrequestsex.
    MESSAGE 'The maximum request throughput was reached.' TYPE 'E'.
ENDTRY.

```

- Para obter detalhes da API, consulte a [UpdateFunctionCode](#) referência da API AWS SDK for SAP ABAP.

UpdateFunctionConfiguration

O código de exemplo a seguir mostra como usar UpdateFunctionConfiguration.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```

TRY.
    oo_result = lo_lmd->updatefunctionconfiguration(      " oo_result is returned
for testing purposes. "
        iv_functionname = iv_function_name
        iv_runtime = iv_runtime
        iv_description = 'Updated Lambda function'
        iv_memorysize = iv_memory_size
    ).

    MESSAGE 'Lambda function configuration/settings updated.' TYPE 'I'.
CATCH /aws1/cx_lmdcodesigningcfgno00.
    MESSAGE 'Code signing configuration does not exist.' TYPE 'E'.
CATCH /aws1/cx_lmdcodeverification00.
    MESSAGE 'Code signature failed one or more validation checks for signature
mismatch or expiration.' TYPE 'E'.
CATCH /aws1/cx_lmdinvalidcodesigex.
    MESSAGE 'Code signature failed the integrity check.' TYPE 'E'.
CATCH /aws1/cx_lmdinvparamvalueex.

```

```
    MESSAGE 'The request contains a non-valid parameter.' TYPE 'E'.
  CATCH /aws1/cx_lmdresourceconflictex.
    MESSAGE 'Resource already exists or another operation is in progress.' TYPE
'E'.
  CATCH /aws1/cx_lmdresourceconflictex.
    MESSAGE 'The requested resource does not exist.' TYPE 'E'.
  CATCH /aws1/cx_lmdserviceexception.
    MESSAGE 'An internal problem was encountered by the AWS Lambda service.'
TYPE 'E'.
  CATCH /aws1/cx_lmdtoomanyrequestsex.
    MESSAGE 'The maximum request throughput was reached.' TYPE 'E'.
  ENDTRY.
```

- Para obter detalhes da API, consulte a [UpdateFunctionConfiguration](#) referência da API AWS SDK for SAP ABAP.

Cenários

Conceitos básicos de funções

O exemplo de código a seguir mostra como:

- Criar um perfil do IAM e uma função do Lambda e carregar o código de manipulador.
- Invocar essa função com um único parâmetro e receber resultados.
- Atualizar o código de função e configurar usando uma variável de ambiente.
- Invocar a função com novos parâmetros e receber resultados. Exibir o log de execução retornado.
- Listar as funções para sua conta e limpar os recursos.

Para obter mais informações, consulte [Criar uma função do Lambda no console](#).

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```

TRY.
  "Create an AWS Identity and Access Management (IAM) role that grants AWS
  Lambda permission to write to logs."
  DATA(lv_policy_document) = `{` &&
    `"Version": "2012-10-17",` &&
    `"Statement": [` &&
      `{` &&
        `"Effect": "Allow",` &&
        `"Action": [` &&
          `"sts:AssumeRole"` &&
        `],` &&
        `"Principal": {` &&
          `"Service": [` &&
            `"lambda.amazonaws.com"` &&
          `]` &&
        `}` &&
      `}` &&
    `]` &&
  `}`.

TRY.
  DATA(lo_create_role_output) = lo_iam->createrole(
    iv_rolename = iv_role_name
    iv_assumerolepolicydocument = lv_policy_document
    iv_description = 'Grant lambda permission to write to logs'
  ).
  MESSAGE 'IAM role created.' TYPE 'I'.
  WAIT UP TO 10 SECONDS.           " Make sure that the IAM role is ready
for use. "
  CATCH /aws1/cx_iamentityalrddyexex.
    MESSAGE 'IAM role already exists.' TYPE 'E'.
  CATCH /aws1/cx_iaminvalidinputex.
    MESSAGE 'The request contains a non-valid parameter.' TYPE 'E'.
  CATCH /aws1/cx_iammalformedplydocex.
    MESSAGE 'Policy document in the request is malformed.' TYPE 'E'.
ENDTRY.

TRY.
  lo_iam->attachrolepolicy(
    iv_rolename = iv_role_name
    iv_policyarn = 'arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole'
  ).

```

```

    MESSAGE 'Attached policy to the IAM role.' TYPE 'I'.
  CATCH /aws1/cx_iaminvalidinputex.
    MESSAGE 'The request contains a non-valid parameter.' TYPE 'E'.
  CATCH /aws1/cx_iamnosuchentityex.
    MESSAGE 'The requested resource entity does not exist.' TYPE 'E'.
  CATCH /aws1/cx_iamplynottattachableex.
    MESSAGE 'Service role policies can only be attached to the service-
linked role for their service.' TYPE 'E'.
  CATCH /aws1/cx_iamunmodableentityex.
    MESSAGE 'Service that depends on the service-linked role is not
modifiable.' TYPE 'E'.
  ENDTRY.

" Create a Lambda function and upload handler code. "
" Lambda function performs 'increment' action on a number. "
TRY.
  lo_lmd->createfunction(
    iv_functionname = iv_function_name
    iv_runtime = `python3.9`
    iv_role = lo_create_role_output->get_role( )->get_arn( )
    iv_handler = iv_handler
    io_code = io_initial_zip_file
    iv_description = 'AWS Lambda code example'
  ).
  MESSAGE 'Lambda function created.' TYPE 'I'.
  CATCH /aws1/cx_lmdcodestorageexcex.
    MESSAGE 'Maximum total code size per account exceeded.' TYPE 'E'.
  CATCH /aws1/cx_lmdinvparamvalueex.
    MESSAGE 'The request contains a non-valid parameter.' TYPE 'E'.
  CATCH /aws1/cx_lmdresourcenotfoundex.
    MESSAGE 'The requested resource does not exist.' TYPE 'E'.
  ENDTRY.

" Verify the function is in Active state "
  WHILE lo_lmd->getfunction( iv_functionname = iv_function_name )-
>get_configuration( )->ask_state( ) <> 'Active'.
    IF sy-index = 10.
      EXIT.          " Maximum 10 seconds. "
    ENDIF.
    WAIT UP TO 1 SECONDS.
  ENDWHILE.

"Invoke the function with a single parameter and get results."
TRY.

```

```

DATA(lv_json) = /aws1/cl_rt_util=>string_to_xstring(
  `{`  &&
  `  "action": "increment",`  &&
  `  "number": 10`  &&
  `}`
).
DATA(lo_initial_invoke_output) = lo_lmd->invoke(
  iv_functionname = iv_function_name
  iv_payload      = lv_json
).
ov_initial_invoke_payload = lo_initial_invoke_output->get_payload( ).
" ov_initial_invoke_payload is returned for testing purposes. "
DATA(lo_writer_json) = cl_sxml_string_writer=>create( type =
if_sxml=>co_xt_json ).
CALL TRANSFORMATION id SOURCE XML ov_initial_invoke_payload RESULT XML
lo_writer_json.
DATA(lv_result) = cl_abap_codepage=>convert_from( lo_writer_json-
>get_output( ) ).
MESSAGE 'Lambda function invoked.' TYPE 'I'.
CATCH /aws1/cx_lmdinvparamvalueex.
MESSAGE 'The request contains a non-valid parameter.' TYPE 'E'.
CATCH /aws1/cx_lmdinvrequestcontex.
MESSAGE 'Unable to parse request body as JSON.' TYPE 'E'.
CATCH /aws1/cx_lmdresourcenotfoundex.
MESSAGE 'The requested resource does not exist.' TYPE 'E'.
CATCH /aws1/cx_lmdunsuppedmediatyp00.
MESSAGE 'Invoke request body does not have JSON as its content type.'
TYPE 'E'.
ENDTRY.

" Update the function code and configure its Lambda environment with an
environment variable. "
" Lambda function is updated to perform 'decrement' action also. "
TRY.
  lo_lmd->updatefunctioncode(
    iv_functionname = iv_function_name
    iv_zipfile      = io_updated_zip_file
  ).
  WAIT UP TO 10 SECONDS.           " Make sure that the update is
completed. "
  MESSAGE 'Lambda function code updated.' TYPE 'I'.
  CATCH /aws1/cx_lmdcodestorageexcex.
  MESSAGE 'Maximum total code size per account exceeded.' TYPE 'E'.
  CATCH /aws1/cx_lmdinvparamvalueex.

```

```

        MESSAGE 'The request contains a non-valid parameter.' TYPE 'E'.
    CATCH /aws1/cx_lmdresourcenotfoundex.
        MESSAGE 'The requested resource does not exist.' TYPE 'E'.
    ENDTRY.

    TRY.
        DATA lt_variables TYPE /aws1/
cl_lmdenvironmentvaria00=>tt_environmentvariables.
        DATA ls_variable LIKE LINE OF lt_variables.
        ls_variable-key = 'LOG_LEVEL'.
        ls_variable-value = NEW /aws1/cl_lmdenvironmentvaria00( iv_value =
'info' ).
        INSERT ls_variable INTO TABLE lt_variables.

        lo_lmd->updatefunctionconfiguration(
            iv_functionname = iv_function_name
            io_environment = NEW /aws1/cl_lmdenvironment( it_variables =
lt_variables )
        ).
        WAIT UP TO 10 SECONDS.           " Make sure that the update is
completed. "
        MESSAGE 'Lambda function configuration/settings updated.' TYPE 'I'.
    CATCH /aws1/cx_lmdinvparamvalueex.
        MESSAGE 'The request contains a non-valid parameter.' TYPE 'E'.
    CATCH /aws1/cx_lmdresourceconflictex.
        MESSAGE 'Resource already exists or another operation is in progress.'
TYPE 'E'.
    CATCH /aws1/cx_lmdresourcenotfoundex.
        MESSAGE 'The requested resource does not exist.' TYPE 'E'.
    ENDTRY.

    "Invoke the function with new parameters and get results. Display the
execution log that's returned from the invocation."
    TRY.
        lv_json = /aws1/cl_rt_util=>string_to_xstring(
            `{` ` &&
            ` "action": "decrement", ` ` &&
            ` "number": 10 ` ` &&
            `}` `
        ).
        DATA(lo_updated_invoke_output) = lo_lmd->invoke(
            iv_functionname = iv_function_name
            iv_payload = lv_json
        ).

```

```

        ov_updated_invoke_payload = lo_updated_invoke_output->get_payload( ).
    " ov_updated_invoke_payload is returned for testing purposes. "
        lo_writer_json = cl_sxml_string_writer=>create( type =
if_sxml=>co_xt_json ).
        CALL TRANSFORMATION id SOURCE XML ov_updated_invoke_payload RESULT XML
lo_writer_json.
        lv_result = cl_abap_codepage=>convert_from( lo_writer_json-
>get_output( ) ).
        MESSAGE 'Lambda function invoked.' TYPE 'I'.
    CATCH /aws1/cx_lmdinvparamvalueex.
        MESSAGE 'The request contains a non-valid parameter.' TYPE 'E'.
    CATCH /aws1/cx_lmdinvrequestcontex.
        MESSAGE 'Unable to parse request body as JSON.' TYPE 'E'.
    CATCH /aws1/cx_lmdresourcenotfoundex.
        MESSAGE 'The requested resource does not exist.' TYPE 'E'.
    CATCH /aws1/cx_lmdunsuppmediatyp00.
        MESSAGE 'Invoke request body does not have JSON as its content type.'
TYPE 'E'.
    ENDRY.

" List the functions for your account. "
    TRY.
        DATA(lo_list_output) = lo_lmd->listfunctions( ).
        DATA(lt_functions) = lo_list_output->get_functions( ).
        MESSAGE 'Retrieved list of Lambda functions.' TYPE 'I'.
    CATCH /aws1/cx_lmdinvparamvalueex.
        MESSAGE 'The request contains a non-valid parameter.' TYPE 'E'.
    ENDRY.

" Delete the Lambda function. "
    TRY.
        lo_lmd->deletefunction( iv_functionname = iv_function_name ).
        MESSAGE 'Lambda function deleted.' TYPE 'I'.
    CATCH /aws1/cx_lmdinvparamvalueex.
        MESSAGE 'The request contains a non-valid parameter.' TYPE 'E'.
    CATCH /aws1/cx_lmdresourcenotfoundex.
        MESSAGE 'The requested resource does not exist.' TYPE 'E'.
    ENDRY.

" Detach role policy. "
    TRY.
        lo_iam->detachrolepolicy(
            iv_rolename = iv_role_name

```

```
        iv_policyarn = 'arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole'
    ).
    MESSAGE 'Detached policy from the IAM role.' TYPE 'I'.
    CATCH /aws1/cx_iaminvalidinputex.
        MESSAGE 'The request contains a non-valid parameter.' TYPE 'E'.
    CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'The requested resource entity does not exist.' TYPE 'E'.
    CATCH /aws1/cx_iamplynnotattachableex.
        MESSAGE 'Service role policies can only be attached to the service-
linked role for their service.' TYPE 'E'.
    CATCH /aws1/cx_iamunmodableentityex.
        MESSAGE 'Service that depends on the service-linked role is not
modifiable.' TYPE 'E'.
    ENDMETHOD.

" Delete the IAM role. "
TRY.
    lo_iam->deleterole( iv_rolename = iv_role_name ).
    MESSAGE 'IAM role deleted.' TYPE 'I'.
    CATCH /aws1/cx_iamnosuchentityex.
        MESSAGE 'The requested resource entity does not exist.' TYPE 'E'.
    CATCH /aws1/cx_iamunmodableentityex.
        MESSAGE 'Service that depends on the service-linked role is not
modifiable.' TYPE 'E'.
    ENDMETHOD.

    CATCH /aws1/cx_rt_service_generic INTO lo_exception.
        DATA(lv_error) = lo_exception->get_longtext( ).
        MESSAGE lv_error TYPE 'E'.
    ENDMETHOD.
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK para SAP ABAP.
 - [CreateFunction](#)
 - [DeleteFunction](#)
 - [GetFunction](#)
 - [Invoke](#)
 - [ListFunctions](#)
 - [UpdateFunctionCode](#)

- [UpdateFunctionConfiguration](#)

Exemplos do Amazon S3 usando o SDK para SAP ABAP

Os exemplos de código a seguir mostram como realizar ações e implementar cenários comuns usando o AWS SDK para SAP ABAP com o Amazon S3.

Ações são trechos de código de programas maiores e devem ser executadas em contexto. Embora as ações mostrem como chamar funções de serviço específicas, é possível ver as ações contextualizadas em seus devidos cenários e exemplos entre serviços.

Cenários são exemplos de código que mostram como realizar uma tarefa específica chamando várias funções dentro do mesmo serviço.

Cada exemplo inclui um link para GitHub, onde você pode encontrar instruções sobre como configurar e executar o código no contexto.

Tópicos

- [Ações](#)
- [Cenários](#)

Ações

CopyObject

O código de exemplo a seguir mostra como usar CopyObject.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

TRY.

```
lo_s3->copyobject(
```

```
        iv_bucket = iv_dest_bucket
        iv_key = iv_dest_object
        iv_copysource = |{ iv_src_bucket }/{ iv_src_object }|
    ).
    MESSAGE 'Object copied to another bucket.' TYPE 'I'.
CATCH /aws1/cx_s3_nosuchbucket.
    MESSAGE 'Bucket does not exist.' TYPE 'E'.
CATCH /aws1/cx_s3_nosuchkey.
    MESSAGE 'Object key does not exist.' TYPE 'E'.
ENDTRY.
```

- Para obter detalhes da API, consulte a [CopyObject](#) referência da API AWS SDK for SAP ABAP.

CreateBucket

O código de exemplo a seguir mostra como usar CreateBucket.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.
    lo_s3->createbucket(
        iv_bucket = iv_bucket_name
    ).
    MESSAGE 'S3 bucket created.' TYPE 'I'.
CATCH /aws1/cx_s3_bucketalrddyexists.
    MESSAGE 'Bucket name already exists.' TYPE 'E'.
CATCH /aws1/cx_s3_bktalrddyownedbyyou.
    MESSAGE 'Bucket already exists and is owned by you.' TYPE 'E'.
ENDTRY.
```

- Para obter detalhes da API, consulte a [CreateBucket](#) referência da API AWS SDK for SAP ABAP.

DeleteBucket

O código de exemplo a seguir mostra como usar DeleteBucket.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.  
  
    lo_s3->deletebucket(  
        iv_bucket = iv_bucket_name  
    ).  
    MESSAGE 'Deleted S3 bucket.' TYPE 'I'.  
CATCH /aws1/cx_s3_nosuchbucket.  
    MESSAGE 'Bucket does not exist.' TYPE 'E'.  
ENDTRY.
```

- Para obter detalhes da API, consulte a [DeleteBucket](#) referência da API AWS SDK for SAP ABAP.

DeleteObject

O código de exemplo a seguir mostra como usar DeleteObject.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.
```

```

lo_s3->deleteobject(
    iv_bucket = iv_bucket_name
    iv_key = iv_object_key
).
MESSAGE 'Object deleted from S3 bucket.' TYPE 'I'.
CATCH /aws1/cx_s3_nosuchbucket.
MESSAGE 'Bucket does not exist.' TYPE 'E'.
ENDTRY.

```

- Para obter detalhes da API, consulte a [DeleteObject](#) referência da API AWS SDK for SAP ABAP.

GetObject

O código de exemplo a seguir mostra como usar GetObject.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```

TRY.
    oo_result = lo_s3->getobject(           " oo_result is returned for testing
purposes. "
        iv_bucket = iv_bucket_name
        iv_key = iv_object_key
    ).
    DATA(lv_object_data) = oo_result->get_body( ).
    MESSAGE 'Object retrieved from S3 bucket.' TYPE 'I'.
CATCH /aws1/cx_s3_nosuchbucket.
    MESSAGE 'Bucket does not exist.' TYPE 'E'.
CATCH /aws1/cx_s3_nosuchkey.
    MESSAGE 'Object key does not exist.' TYPE 'E'.
ENDTRY.

```

- Para obter detalhes da API, consulte a [GetObject](#) referência da API AWS SDK for SAP ABAP.

ListObjectsV2

O código de exemplo a seguir mostra como usar ListObjectsV2.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.  
    oo_result = lo_s3->listobjectsv2(          " oo_result is returned for  
testing purposes. "  
    iv_bucket = iv_bucket_name  
    ).  
    MESSAGE 'Retrieved list of objects in S3 bucket.' TYPE 'I'.  
CATCH /aws1/cx_s3_nosuchbucket.  
    MESSAGE 'Bucket does not exist.' TYPE 'E'.  
ENDTRY.
```

- Para obter detalhes da API, consulte [ListObjectsV2 na referência](#) da API ABAP do AWS SDK para SAP.

PutObject

O código de exemplo a seguir mostra como usar PutObject.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
"Get contents of file from application server."
```

```
DATA lv_body TYPE xstring.
OPEN DATASET iv_file_name FOR INPUT IN BINARY MODE.
READ DATASET iv_file_name INTO lv_body.
CLOSE DATASET iv_file_name.

"Upload/put an object to an S3 bucket."
TRY.
  lo_s3->putobject(
    iv_bucket = iv_bucket_name
    iv_key = iv_file_name
    iv_body = lv_body
  ).
  MESSAGE 'Object uploaded to S3 bucket.' TYPE 'I'.
CATCH /aws1/cx_s3_nosuchbucket.
  MESSAGE 'Bucket does not exist.' TYPE 'E'.
ENDTRY.
```

- Para obter detalhes da API, consulte a [PutObject](#) referência da API AWS SDK for SAP ABAP.

Cenários

Conceitos básicos de buckets e objetos

O exemplo de código a seguir mostra como:

- Criar um bucket e fazer upload de um arquivo para ele.
- Baixar um objeto de um bucket.
- Copiar um objeto em uma subpasta em um bucket.
- Listar os objetos em um bucket.
- Excluir os objetos do bucket e o bucket.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
DATA(lo_session) = /aws1/cl_rt_session_aws=>create( cv_pfl ).
DATA(lo_s3) = /aws1/cl_s3_factory=>create( lo_session ).

" Create an Amazon Simple Storage Service (Amazon S3) bucket. "
TRY.
  lo_s3->createbucket(
    iv_bucket = iv_bucket_name
  ).
  MESSAGE 'S3 bucket created.' TYPE 'I'.
CATCH /aws1/cx_s3_bucketalrddyexists.
  MESSAGE 'Bucket name already exists.' TYPE 'E'.
CATCH /aws1/cx_s3_bktalrddyownedbyyou.
  MESSAGE 'Bucket already exists and is owned by you.' TYPE 'E'.
ENDTRY.

"Upload an object to an S3 bucket."
TRY.
  "Get contents of file from application server."
  DATA lv_file_content TYPE xstring.
  OPEN DATASET iv_key FOR INPUT IN BINARY MODE.
  READ DATASET iv_key INTO lv_file_content.
  CLOSE DATASET iv_key.

  lo_s3->putobject(
    iv_bucket = iv_bucket_name
    iv_key = iv_key
    iv_body = lv_file_content
  ).
  MESSAGE 'Object uploaded to S3 bucket.' TYPE 'I'.
CATCH /aws1/cx_s3_nosuchbucket.
  MESSAGE 'Bucket does not exist.' TYPE 'E'.
ENDTRY.

" Get an object from a bucket. "
TRY.
  DATA(lo_result) = lo_s3->getobject(
    iv_bucket = iv_bucket_name
    iv_key = iv_key
  ).
  DATA(lv_object_data) = lo_result->get_body( ).
  MESSAGE 'Object retrieved from S3 bucket.' TYPE 'I'.
CATCH /aws1/cx_s3_nosuchbucket.
```

```
    MESSAGE 'Bucket does not exist.' TYPE 'E'.
  CATCH /aws1/cx_s3_nosuchkey.
    MESSAGE 'Object key does not exist.' TYPE 'E'.
ENDTRY.

" Copy an object to a subfolder in a bucket. "
TRY.
  lo_s3->copyobject(
    iv_bucket = iv_bucket_name
    iv_key = |{ iv_copy_to_folder }/{ iv_key }|
    iv_copysource = |{ iv_bucket_name }/{ iv_key }|
  ).
  MESSAGE 'Object copied to a subfolder.' TYPE 'I'.
  CATCH /aws1/cx_s3_nosuchbucket.
    MESSAGE 'Bucket does not exist.' TYPE 'E'.
  CATCH /aws1/cx_s3_nosuchkey.
    MESSAGE 'Object key does not exist.' TYPE 'E'.
ENDTRY.

" List objects in the bucket. "
TRY.
  DATA(lo_list) = lo_s3->listobjects(
    iv_bucket = iv_bucket_name
  ).
  MESSAGE 'Retrieved list of objects in S3 bucket.' TYPE 'I'.
  CATCH /aws1/cx_s3_nosuchbucket.
    MESSAGE 'Bucket does not exist.' TYPE 'E'.
ENDTRY.
DATA text TYPE string VALUE 'Object List - '.
DATA lv_object_key TYPE /aws1/s3_objectkey.
LOOP AT lo_list->get_contents( ) INTO DATA(lo_object).
  lv_object_key = lo_object->get_key( ).
  CONCATENATE lv_object_key ', ' INTO text.
ENDLOOP.
MESSAGE text TYPE 'I'.

" Delete the objects in a bucket. "
TRY.
  lo_s3->deleteobject(
    iv_bucket = iv_bucket_name
    iv_key = iv_key
  ).
  lo_s3->deleteobject(
    iv_bucket = iv_bucket_name
```



```
        iv_key = |{ iv_copy_to_folder }/{ iv_key }|
    ).
    MESSAGE 'Objects deleted from S3 bucket.' TYPE 'I'.
    CATCH /aws1/cx_s3_nosuchbucket.
        MESSAGE 'Bucket does not exist.' TYPE 'E'.
    ENDTRY.

" Delete the bucket. "
TRY.
    lo_s3->deletebucket(
        iv_bucket = iv_bucket_name
    ).
    MESSAGE 'Deleted S3 bucket.' TYPE 'I'.
    CATCH /aws1/cx_s3_nosuchbucket.
        MESSAGE 'Bucket does not exist.' TYPE 'E'.
    ENDTRY.
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK para SAP ABAP.
 - [CopyObject](#)
 - [CreateBucket](#)
 - [DeleteBucket](#)
 - [DeleteObjects](#)
 - [GetObject](#)
 - [ListObjectsV2](#)
 - [PutObject](#)

SageMaker exemplos usando SDK para SAP ABAP

Os exemplos de código a seguir mostram como realizar ações e implementar cenários comuns usando o AWS SDK para SAP ABAP com. SageMaker

Ações são trechos de código de programas maiores e devem ser executadas em contexto. Embora as ações mostrem como chamar funções de serviço específicas, é possível ver as ações contextualizadas em seus devidos cenários e exemplos entre serviços.

Cenários são exemplos de código que mostram como realizar uma tarefa específica chamando várias funções dentro do mesmo serviço.

Cada exemplo inclui um link para GitHub, onde você pode encontrar instruções sobre como configurar e executar o código no contexto.

Tópicos

- [Ações](#)
- [Cenários](#)

Ações

CreateEndpoint

O código de exemplo a seguir mostra como usar CreateEndpoint.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
DATA lt_production_variants TYPE /aws1/
cl_sgmproductionvariant=>tt_productionvariantlist.
DATA lo_production_variants TYPE REF TO /aws1/cl_sgmproductionvariant.
DATA oo_ep_config_result TYPE REF TO /aws1/cl_sgmcreateendptcfgout.

"Create a production variant as an ABAP object."
"Identifies a model that you want to host and the resources chosen to deploy for
hosting it."
CREATE OBJECT lo_production_variants
  EXPORTING
    iv_variantname      = iv_variant_name
    iv_modelname        = iv_model_name
    iv_initialinstancecount = iv_initial_instance_count
    iv_instancetype     = iv_instance_type.
```

```
INSERT lo_production_variants INTO TABLE lt_production_variants.

"Create an endpoint configuration."
TRY.
    oo_ep_config_result = lo_sgm->createendpointconfig(
        iv_endpointconfigname = iv_endpoint_config_name
        it_productionvariants = lt_production_variants
    ).
    MESSAGE 'Endpoint configuration created.' TYPE 'I'.
CATCH /aws1/cx_sgmresourcecelimitexcd.
    MESSAGE 'You have reached the limit on the number of resources.' TYPE 'E'.
ENDTRY.

"Create an endpoint."
TRY.
    oo_result = lo_sgm->createendpoint(      " oo_result is returned for testing
purposes. "
        iv_endpointconfigname = iv_endpoint_config_name
        iv_endpointname = iv_endpoint_name
    ).
    MESSAGE 'Endpoint created.' TYPE 'I'.
CATCH /aws1/cx_sgmresourcecelimitexcd.
    MESSAGE 'You have reached the limit on the number of resources.' TYPE 'E'.
ENDTRY.
```

- Para obter detalhes da API, consulte a [CreateEndpoint](#) referência da API AWS SDK for SAP ABAP.

CreateModel

O código de exemplo a seguir mostra como usar CreateModel.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```

DATA lo_primarycontainer TYPE REF TO /aws1/cl_sgmcontainerdefn.

"Create an ABAP object for the container image based on input variables."
CREATE OBJECT lo_primarycontainer
  EXPORTING
    iv_image          = iv_container_image
    iv_modeldataurl  = iv_model_data_url.

"Create an Amazon SageMaker model."
TRY.
  oo_result = lo_sgm->createmodel(      " oo_result is returned for testing
purposes. "
    iv_executionrolearn = iv_execution_role_arn
    iv_modelname        = iv_model_name
    io_primarycontainer = lo_primarycontainer
  ).
  MESSAGE 'Model created.' TYPE 'I'.
CATCH /aws1/cx_sgmresourcecelimitexcd.
  MESSAGE 'You have reached the limit on the number of resources.' TYPE 'E'.
ENDTRY.

```

- Para obter detalhes da API, consulte a [CreateModel](#) referência da API AWS SDK for SAP ABAP.

CreateTrainingJob

O código de exemplo a seguir mostra como usar CreateTrainingJob.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```

DATA lo_hyperparameters_w TYPE REF TO /aws1/cl_sgmhyperparameters_w.
DATA lt_hyperparameters TYPE /aws1/c1_sgmhyperparameters_w=>tt_hyperparameters.

```

```
DATA lt_input_data_config TYPE /aws1/cl_sgmchannel=>tt_inputdataconfig.
DATA lo_trn_channel TYPE REF TO /aws1/cl_sgmchannel.
DATA lo_trn_datasource TYPE REF TO /aws1/cl_sgmdatasource.
DATA lo_trn_s3datasource TYPE REF TO /aws1/cl_sgms3datasource.
DATA lo_val_channel TYPE REF TO /aws1/cl_sgmchannel.
DATA lo_val_datasource TYPE REF TO /aws1/cl_sgmdatasource.
DATA lo_val_s3datasource TYPE REF TO /aws1/cl_sgms3datasource.
DATA lo_algorithm_specification TYPE REF TO /aws1/cl_sgmalgorithm-spec.
DATA lo_resource_config TYPE REF TO /aws1/cl_sgmresourceconfig.
DATA lo_output_data_config TYPE REF TO /aws1/cl_sgmoutputdataconfig.
DATA lo_stopping_condition TYPE REF TO /aws1/cl_sgmstoppingcondition.
```

"Create ABAP internal table for hyperparameters based on input variables."

"These hyperparameters are based on the Amazon SageMaker built-in algorithm, XGBoost."

```
CREATE OBJECT lo_hyperparameters_w EXPORTING iv_value = iv_hp_max_depth.
INSERT VALUE #( key = 'max_depth' value = lo_hyperparameters_w ) INTO TABLE
lt_hyperparameters.
```

```
CREATE OBJECT lo_hyperparameters_w EXPORTING iv_value = iv_hp_eta.
INSERT VALUE #( key = 'eta' value = lo_hyperparameters_w ) INTO TABLE
lt_hyperparameters.
```

```
CREATE OBJECT lo_hyperparameters_w EXPORTING iv_value = iv_hp_eval_metric.
INSERT VALUE #( key = 'eval_metric' value = lo_hyperparameters_w ) INTO TABLE
lt_hyperparameters.
```

```
CREATE OBJECT lo_hyperparameters_w EXPORTING iv_value = iv_hp_scale_pos_weight.
INSERT VALUE #( key = 'scale_pos_weight' value = lo_hyperparameters_w ) INTO
TABLE lt_hyperparameters.
```

```
CREATE OBJECT lo_hyperparameters_w EXPORTING iv_value = iv_hp_subsample.
INSERT VALUE #( key = 'subsample' value = lo_hyperparameters_w ) INTO TABLE
lt_hyperparameters.
```

```
CREATE OBJECT lo_hyperparameters_w EXPORTING iv_value = iv_hp_objective.
INSERT VALUE #( key = 'objective' value = lo_hyperparameters_w ) INTO TABLE
lt_hyperparameters.
```

```
CREATE OBJECT lo_hyperparameters_w EXPORTING iv_value = iv_hp_num_round.
INSERT VALUE #( key = 'num_round' value = lo_hyperparameters_w ) INTO TABLE
lt_hyperparameters.
```

"Create ABAP objects for training data sources."

```
CREATE OBJECT lo_trn_s3datasource
EXPORTING
  iv_s3datatype           = iv_trn_data_s3datatype
  iv_s3datadistributiontype = iv_trn_data_s3datadistribution
  iv_s3uri                 = iv_trn_data_s3uri.

CREATE OBJECT lo_trn_datasource
EXPORTING
  io_s3datasource = lo_trn_s3datasource.

CREATE OBJECT lo_trn_channel
EXPORTING
  iv_channelname      = 'train'
  io_datasource       = lo_trn_datasource
  iv_compressiontype = iv_trn_data_compressiontype
  iv_contenttype      = iv_trn_data_contenttype.

INSERT lo_trn_channel INTO TABLE lt_input_data_config.

"Create ABAP objects for validation data sources."
CREATE OBJECT lo_val_s3datasource
EXPORTING
  iv_s3datatype           = iv_val_data_s3datatype
  iv_s3datadistributiontype = iv_val_data_s3datadistribution
  iv_s3uri                 = iv_val_data_s3uri.

CREATE OBJECT lo_val_datasource
EXPORTING
  io_s3datasource = lo_val_s3datasource.

CREATE OBJECT lo_val_channel
EXPORTING
  iv_channelname      = 'validation'
  io_datasource       = lo_val_datasource
  iv_compressiontype = iv_val_data_compressiontype
  iv_contenttype      = iv_val_data_contenttype.

INSERT lo_val_channel INTO TABLE lt_input_data_config.

"Create an ABAP object for algorithm specification."
CREATE OBJECT lo_algorithm_specification
EXPORTING
  iv_trainingimage      = iv_training_image
  iv_traininginputmode = iv_training_input_mode.
```

```
"Create an ABAP object for resource configuration."
CREATE OBJECT lo_resource_config
EXPORTING
  iv_instancecount = iv_instance_count
  iv_instancetype  = iv_instance_type
  iv_volumesizeingb = iv_volume_sizeingb.

"Create an ABAP object for output data configuration."
CREATE OBJECT lo_output_data_config
EXPORTING
  iv_s3outputpath = iv_s3_output_path.

"Create an ABAP object for stopping condition."
CREATE OBJECT lo_stopping_condition
EXPORTING
  iv_maxruntimeinseconds = iv_max_runtime_in_seconds.

"Create a training job."
TRY.
  oo_result = lo_sgm->createtrainingjob( " oo_result is returned for
testing purposes. "
    iv_trainingjobname      = iv_training_job_name
    iv_rolearn              = iv_role_arn
    it_hyperparameters      = lt_hyperparameters
    it_inputdataconfig      = lt_input_data_config
    io_algorithmspecification = lo_algorithm_specification
    io_outputdataconfig     = lo_output_data_config
    io_resourceconfig       = lo_resource_config
    io_stoppingcondition    = lo_stopping_condition
  ).
  MESSAGE 'Training job created.' TYPE 'I'.
CATCH /aws1/cx_sgmresourceinuse.
  MESSAGE 'Resource being accessed is in use.' TYPE 'E'.
CATCH /aws1/cx_sgmresourcenotfound.
  MESSAGE 'Resource being accessed is not found.' TYPE 'E'.
CATCH /aws1/cx_sgmresourcecelimitexcd.
  MESSAGE 'You have reached the limit on the number of resources.' TYPE 'E'.
ENDTRY.
```

- Para obter detalhes da API, consulte a [CreateTrainingJob](#) referência da API AWS SDK for SAP ABAP.

CreateTransformJob

O código de exemplo a seguir mostra como usar CreateTransformJob.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
DATA lo_transforminput TYPE REF TO /aws1/cl_sgmtransforminput.  
DATA lo_transformoutput TYPE REF TO /aws1/cl_sgmtransformoutput.  
DATA lo_transformresources TYPE REF TO /aws1/cl_sgmtransformresources.  
DATA lo_datasource TYPE REF TO /aws1/cl_sgmtransformdatasrc.  
DATA lo_s3datasource TYPE REF TO /aws1/cl_sgmtransformS3datasrc.
```

"Create an ABAP object for an Amazon Simple Storage Service (Amazon S3) data source."

```
CREATE OBJECT lo_s3datasource  
EXPORTING  
    iv_s3uri      = iv_tf_data_s3uri  
    iv_s3datatype = iv_tf_data_s3datatype.
```

"Create an ABAP object for data source."

```
CREATE OBJECT lo_datasource  
EXPORTING  
    io_s3datasource = lo_s3datasource.
```

"Create an ABAP object for transform data source."

```
CREATE OBJECT lo_transforminput  
EXPORTING  
    io_datasource      = lo_datasource  
    iv_contenttype     = iv_tf_data_contenttype  
    iv_compressiontype = iv_tf_data_compressiontype.
```

"Create an ABAP object for resource configuration."

```
CREATE OBJECT lo_transformresources  
EXPORTING  
    iv_instancecount = iv_instance_count  
    iv_instancetype  = iv_instance_type.
```



```
"Create an ABAP object for output data configuration."
CREATE OBJECT lo_transformoutput
  EXPORTING
    iv_s3outputpath = iv_s3_output_path.

"Create a transform job."
TRY.
  oo_result = lo_sgm->createtransformjob( " oo_result is returned for
testing purposes. "
    iv_modelname = iv_tf_model_name
    iv_transformjobname = iv_tf_job_name
    io_transforminput = lo_transforminput
    io_transformoutput = lo_transformoutput
    io_transformresources = lo_transformresources
  ).
  MESSAGE 'Transform job created.' TYPE 'I'.
CATCH /aws1/cx_sgmresourceinuse.
  MESSAGE 'Resource being accessed is in use.' TYPE 'E'.
CATCH /aws1/cx_sgmresourceNotFound.
  MESSAGE 'Resource being accessed is not found.' TYPE 'E'.
CATCH /aws1/cx_sgmresourceLimitExcd.
  MESSAGE 'You have reached the limit on the number of resources.' TYPE 'E'.
ENDTRY.
```

- Para obter detalhes da API, consulte a [CreateTransformJob](#) referência da API AWS SDK for SAP ABAP.

DeleteEndpoint

O código de exemplo a seguir mostra como usar DeleteEndpoint.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
"Delete an endpoint."
TRY.
  lo_sgm->deleteendpoint(
    iv_endpointname = iv_endpoint_name
  ).
  MESSAGE 'Endpoint configuration deleted.' TYPE 'I'.
CATCH /aws1/cx_rt_service_generic INTO DATA(lo_endpoint_exception).
  DATA(lv_endpoint_error) = |"{ lo_endpoint_exception->av_err_code }" -
{ lo_endpoint_exception->av_err_msg }|.
  MESSAGE lv_endpoint_error TYPE 'E'.
ENDTRY.

"Delete an endpoint configuration."
TRY.
  lo_sgm->deleteendpointconfig(
    iv_endpointconfigname = iv_endpoint_config_name
  ).
  MESSAGE 'Endpoint deleted.' TYPE 'I'.
CATCH /aws1/cx_rt_service_generic INTO DATA(lo_endpointconfig_exception).
  DATA(lv_endpointconfig_error) = |"{ lo_endpointconfig_exception-
>av_err_code }" - { lo_endpointconfig_exception->av_err_msg }|.
  MESSAGE lv_endpointconfig_error TYPE 'E'.
ENDTRY.
```

- Para obter detalhes da API, consulte a [DeleteEndpoint](#) referência da API AWS SDK for SAP ABAP.

DeleteModel

O código de exemplo a seguir mostra como usar DeleteModel.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.
```

```

lo_sgm->deletemodel(
    iv_modelname = iv_model_name
).
MESSAGE 'Model deleted.' TYPE 'I'.
CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
MESSAGE lv_error TYPE 'E'.
ENDTRY.

```

- Para obter detalhes da API, consulte a [DeleteModel](#) referência da API AWS SDK for SAP ABAP.

DescribeTrainingJob

O código de exemplo a seguir mostra como usar DescribeTrainingJob.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```

TRY.
    oo_result = lo_sgm->describetrainingjob( " oo_result is returned for
testing purposes. "
    iv_trainingjobname = iv_training_job_name
).
MESSAGE 'Retrieved description of training job.' TYPE 'I'.
CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
MESSAGE lv_error TYPE 'E'.
ENDTRY.

```

- Para obter detalhes da API, consulte a [DescribeTrainingJob](#) referência da API AWS SDK for SAP ABAP.

ListAlgorithms

O código de exemplo a seguir mostra como usar ListAlgorithms.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.
    oo_result = lo_sgm->listalgorithms(           " oo_result is returned for
testing purposes. "
    iv_namecontains = iv_name_contains
    ).
    MESSAGE 'Retrieved list of algorithms.' TYPE 'I'.
    CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
    DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
    MESSAGE lv_error TYPE 'E'.
ENDTRY.
```

- Para obter detalhes da API, consulte a [ListAlgorithms](#) referência da API AWS SDK for SAP ABAP.

ListModels

O código de exemplo a seguir mostra como usar ListModels.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.
    oo_result = lo_sgm->listmodels(           " oo_result is returned for
testing purposes. "
        iv_namecontains = iv_name_contains
    ).
    MESSAGE 'Retrieved list of models.' TYPE 'I'.
    CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
    DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
    MESSAGE lv_error TYPE 'E'.
ENDTRY.
```

- Para obter detalhes da API, consulte a [ListModels](#) referência da API AWS SDK for SAP ABAP.

ListNotebookInstances

O código de exemplo a seguir mostra como usar ListNotebookInstances.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.
    oo_result = lo_sgm->listnotebookinstances(           " oo_result is returned
for testing purposes. "
        iv_namecontains = iv_name_contains
    ).
    MESSAGE 'Retrieved list of notebook instances.' TYPE 'I'.
    CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).
    DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-
>av_err_msg }|.
    MESSAGE lv_error TYPE 'E'.
ENDTRY.
```

- Para obter detalhes da API, consulte a [ListNotebookInstances](#) referência da API AWS SDK for SAP ABAP.

ListTrainingJobs

O código de exemplo a seguir mostra como usar ListTrainingJobs.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.  
    oo_result = lo_sgm->listtrainingjobs(      " oo_result is returned for  
testing purposes. "  
        iv_namecontains = iv_name_contains  
        iv_maxresults = iv_max_results  
    ).  
    MESSAGE 'Retrieved list of training jobs.' TYPE 'I'.  
    CATCH /aws1/cx_rt_service_generic INTO DATA(lo_exception).  
    DATA(lv_error) = |"{ lo_exception->av_err_code }" - { lo_exception-  
>av_err_msg }|.  
    MESSAGE lv_error TYPE 'E'.  
ENDTRY.
```

- Para obter detalhes da API, consulte a [ListTrainingJobs](#) referência da API AWS SDK for SAP ABAP.

Cenários

Conceitos básicos de usar modelos e endpoints

O exemplo de código a seguir mostra como:

- Comece um trabalho de treinamento e crie um SageMaker modelo.
- Criar uma configuração de endpoint.

- Criar um endpoint e, em seguida, limpar os recursos.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```

DATA lo_hyperparameters_w TYPE REF TO /aws1/cl_sgmhyperparameters_w.
DATA lo_trn_channel TYPE REF TO /aws1/cl_sgmchannel.
DATA lo_trn_datasource TYPE REF TO /aws1/cl_sgmdatasource.
DATA lo_trn_s3datasource TYPE REF TO /aws1/cl_sgms3datasource.
DATA lo_val_channel TYPE REF TO /aws1/cl_sgmchannel.
DATA lo_val_datasource TYPE REF TO /aws1/cl_sgmdatasource.
DATA lo_val_s3datasource TYPE REF TO /aws1/cl_sgms3datasource.
DATA lo_algorithm_specification TYPE REF TO /aws1/cl_sgmalgorithm-spec.
DATA lo_resource_config TYPE REF TO /aws1/cl_sgmresourceconfig.
DATA lo_output_data_config TYPE REF TO /aws1/cl_sgmoutputdataconfig.
DATA lo_stopping_condition TYPE REF TO /aws1/cl_sgmstoppingcondition.
DATA lo_primarycontainer TYPE REF TO /aws1/cl_sgmcontainerdefn.
DATA lo_production_variants TYPE REF TO /aws1/cl_sgmproductionvariant.
DATA lo_ep_config_result TYPE REF TO /aws1/cl_sgmcreateendptcfgout.
DATA lo_training_result TYPE REF TO /aws1/cl_sgmdescrtrnjobrsp.
DATA lt_production_variants TYPE /aws1/
cl_sgmproductionvariant=>tt_productionvariantlist.
DATA lt_input_data_config TYPE /aws1/cl_sgmchannel=>tt_inputdataconfig.
DATA lt_hyperparameters TYPE /aws1/cl_sgmhyperparameters_w=>tt_hyperparameters.
DATA lv_model_data_url TYPE /aws1/sgmurl.

lv_model_data_url = iv_s3_output_path && iv_training_job_name && '/output/
model.tar.gz'.

"Create ABAP internal table for hyperparameters based on input variables."
"These hyperparameters are based on Amazon SageMaker built-in algorithm -
XGBoost"
CREATE OBJECT lo_hyperparameters_w EXPORTING iv_value = iv_hp_max_depth.
INSERT VALUE #( key = 'max_depth' value = lo_hyperparameters_w ) INTO TABLE
lt_hyperparameters.

```

```
CREATE OBJECT lo_hyperparameters_w EXPORTING iv_value = iv_hp_eta.
INSERT VALUE #( key = 'eta' value = lo_hyperparameters_w ) INTO TABLE
lt_hyperparameters.

CREATE OBJECT lo_hyperparameters_w EXPORTING iv_value = iv_hp_eval_metric.
INSERT VALUE #( key = 'eval_metric' value = lo_hyperparameters_w ) INTO TABLE
lt_hyperparameters.

CREATE OBJECT lo_hyperparameters_w EXPORTING iv_value = iv_hp_scale_pos_weight.
INSERT VALUE #( key = 'scale_pos_weight' value = lo_hyperparameters_w ) INTO
TABLE lt_hyperparameters.

CREATE OBJECT lo_hyperparameters_w EXPORTING iv_value = iv_hp_subsample.
INSERT VALUE #( key = 'subsample' value = lo_hyperparameters_w ) INTO TABLE
lt_hyperparameters.

CREATE OBJECT lo_hyperparameters_w EXPORTING iv_value = iv_hp_objective.
INSERT VALUE #( key = 'objective' value = lo_hyperparameters_w ) INTO TABLE
lt_hyperparameters.

CREATE OBJECT lo_hyperparameters_w EXPORTING iv_value = iv_hp_num_round.
INSERT VALUE #( key = 'num_round' value = lo_hyperparameters_w ) INTO TABLE
lt_hyperparameters.

"Create ABAP internal table for data based on input variables."
"Training data."
CREATE OBJECT lo_trn_s3datasource
EXPORTING
    iv_s3datatype           = iv_trn_data_s3datatype
    iv_s3datadistributiontype = iv_trn_data_s3datadistribution
    iv_s3uri                 = iv_trn_data_s3uri.

CREATE OBJECT lo_trn_datasource EXPORTING io_s3datasource = lo_trn_s3datasource.

CREATE OBJECT lo_trn_channel
EXPORTING
    iv_channelname      = 'train'
    io_datasource       = lo_trn_datasource
    iv_compressiontype  = iv_trn_data_compressiontype
    iv_contenttype      = iv_trn_data_contenttype.
INSERT lo_trn_channel INTO TABLE lt_input_data_config.

"Validation data."
CREATE OBJECT lo_val_s3datasource
```



```
EXPORTING
  iv_s3datatype           = iv_val_data_s3datatype
  iv_s3datadistributiontype = iv_val_data_s3datadistribution
  iv_s3uri                = iv_val_data_s3uri.

CREATE OBJECT lo_val_datasource EXPORTING io_s3datasource = lo_val_s3datasource.

CREATE OBJECT lo_val_channel
  EXPORTING
    iv_channelname      = 'validation'
    io_datasource       = lo_val_datasource
    iv_compressiontype  = iv_val_data_compressiontype
    iv_contenttype      = iv_val_data_contenttype.
INSERT lo_val_channel INTO TABLE lt_input_data_config.

"Create an ABAP object for algorithm specification based on input variables."
CREATE OBJECT lo_algorithm_specification
  EXPORTING
    iv_trainingimage      = iv_training_image
    iv_traininginputmode  = iv_training_input_mode.

"Create an ABAP object for resource configuration."
CREATE OBJECT lo_resource_config
  EXPORTING
    iv_instancecount     = iv_instance_count
    iv_instancetype      = iv_instance_type
    iv_volumesizeingb    = iv_volume_sizeingb.

"Create an ABAP object for output data configuration."
CREATE OBJECT lo_output_data_config EXPORTING iv_s3outputpath =
iv_s3_output_path.

"Create an ABAP object for stopping condition."
CREATE OBJECT lo_stopping_condition EXPORTING iv_maxruntimeinseconds =
iv_max_runtime_in_seconds.

TRY.
  lo_sgm->createtrainingjob(
    iv_trainingjobname      = iv_training_job_name
    iv_rolearn              = iv_role_arn
    it_hyperparameters      = lt_hyperparameters
    it_inputdataconfig      = lt_input_data_config
    io_algorithmspecification = lo_algorithm_specification
    io_outputdataconfig     = lo_output_data_config
```

```
        io_resourceconfig          = lo_resource_config
        io_stoppingcondition       = lo_stopping_condition
    ).
    MESSAGE 'Training job created.' TYPE 'I'.
    CATCH /aws1/cx_sgmresourceinuse.
        MESSAGE 'Resource being accessed is in use.' TYPE 'E'.
    CATCH /aws1/cx_sgmresourceNotFound.
        MESSAGE 'Resource being accessed is not found.' TYPE 'E'.
    CATCH /aws1/cx_sgmresourceLimitExcd.
        MESSAGE 'You have reached the limit on the number of resources.' TYPE 'E'.
    ENDTRY.

    "Wait for training job to be completed."
    lo_training_result = lo_sgm->describetrainingjob( iv_trainingjobname =
iv_training_job_name ).
    WHILE lo_training_result->get_trainingjobstatus( ) <> 'Completed'.
        IF sy-index = 30.
            EXIT.                "Maximum 900 seconds."
        ENDIF.
        WAIT UP TO 30 SECONDS.
        lo_training_result = lo_sgm->describetrainingjob( iv_trainingjobname =
iv_training_job_name ).
    ENDWHILE.

    "Create ABAP object for the container image based on input variables."
    CREATE OBJECT lo_primarycontainer
    EXPORTING
        iv_image          = iv_training_image
        iv_modeldataurl = lv_model_data_url.

    "Create an Amazon SageMaker model."
    TRY.
        lo_sgm->createmodel(
            iv_executionrolearn = iv_role_arn
            iv_modelname = iv_model_name
            io_primarycontainer = lo_primarycontainer
        ).
        MESSAGE 'Model created.' TYPE 'I'.
    CATCH /aws1/cx_sgmresourceLimitExcd.
        MESSAGE 'You have reached the limit on the number of resources.' TYPE 'E'.
    ENDTRY.

    "Create an endpoint production variant."
    CREATE OBJECT lo_production_variants
```

```
EXPORTING
  iv_variantname          = iv_ep_variant_name
  iv_modelname           = iv_model_name
  iv_initialinstancecount = iv_ep_initial_instance_count
  iv_instancetype        = iv_ep_instance_type.
INSERT lo_production_variants INTO TABLE lt_production_variants.

TRY.
  "Create an endpoint configuration."
  lo_ep_config_result = lo_sgm->createendpointconfig(
    iv_endpointconfigname = iv_ep_cfg_name
    it_productionvariants = lt_production_variants
  ).
  MESSAGE 'Endpoint configuration created.' TYPE 'I'.

  "Create an endpoint."
  oo_ep_output = lo_sgm->createendpoint(           " oo_ep_output is returned for
testing purposes. "
    iv_endpointconfigname = iv_ep_cfg_name
    iv_endpointname       = iv_ep_name
  ).
  MESSAGE 'Endpoint created.' TYPE 'I'.
CATCH /aws1/cx_sgmresource-limit-excd.
  MESSAGE 'You have reached the limit on the number of resources.' TYPE 'E'.
ENDTRY.

"Wait for endpoint creation to be completed."
DATA(lo_endpoint_result) = lo_sgm->describeendpoint( iv_endpointname =
iv_ep_name ).
WHILE lo_endpoint_result->get_endpointstatus( ) <> 'InService'.
  IF sy-index = 30.
    EXIT.           "Maximum 900 seconds."
  ENDIF.
  WAIT UP TO 30 SECONDS.
  lo_endpoint_result = lo_sgm->describeendpoint( iv_endpointname = iv_ep_name ).
ENDWHILE.

TRY.
  "Delete an endpoint."
  lo_sgm->deleteendpoint(
    iv_endpointname = iv_ep_name
  ).
  MESSAGE 'Endpoint deleted' TYPE 'I'.
```

```
"Delete an endpoint configuration."
lo_sgm->deleteendpointconfig(
    iv_endpointconfigname = iv_ep_cfg_name
).
MESSAGE 'Endpoint configuration deleted.' TYPE 'I'.

"Delete model."
lo_sgm->deletemodel(
    iv_modelname = iv_model_name
).
MESSAGE 'Model deleted.' TYPE 'I'.
CATCH /aws1/cx_rt_service_generic INTO DATA(lo_endpointconfig_exception).
    DATA(lv_endpointconfig_error) = |"{ lo_endpointconfig_exception-
>av_err_code }" - { lo_endpointconfig_exception->av_err_msg }|.
    MESSAGE lv_endpointconfig_error TYPE 'E'.
ENDTRY.
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK para SAP ABAP.
 - [CreateEndpoint](#)
 - [CreateEndpointConfig](#)
 - [CreateModel](#)
 - [CreateTrainingJob](#)
 - [DeleteEndpoint](#)
 - [DeleteEndpointConfig](#)
 - [DeleteModel](#)
 - [DescribeEndpoint](#)
 - [DescribeTrainingJob](#)

Exemplos do Amazon SNS usando o SDK para SAP ABAP

Os exemplos de código a seguir mostram como realizar ações e implementar cenários comuns usando o AWS SDK para SAP ABAP com o Amazon SNS.

Ações são trechos de código de programas maiores e devem ser executadas em contexto. Embora as ações mostrem como chamar funções de serviço específicas, é possível ver as ações contextualizadas em seus devidos cenários e exemplos entre serviços.

Cenários são exemplos de código que mostram como realizar uma tarefa específica chamando várias funções dentro do mesmo serviço.

Cada exemplo inclui um link para GitHub, onde você pode encontrar instruções sobre como configurar e executar o código no contexto.

Tópicos

- [Ações](#)
- [Cenários](#)

Ações

CreateTopic

O código de exemplo a seguir mostra como usar CreateTopic.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.  
    oo_result = lo_sns->createtopic( iv_name = iv_topic_name ). " oo_result is  
returned for testing purposes. "  
    MESSAGE 'SNS topic created' TYPE 'I'.  
    CATCH /aws1/cx_snstopiclimitexcdex.  
        MESSAGE 'Unable to create more topics. You have reached the maximum number  
of topics allowed.' TYPE 'E'.  
ENDTRY.
```

- Para obter detalhes da API, consulte a [CreateTopic](#) referência da API AWS SDK for SAP ABAP.

DeleteTopic

O código de exemplo a seguir mostra como usar DeleteTopic.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.  
    lo_sns->deletetopic( iv_topicarn = iv_topic_arn ).  
    MESSAGE 'SNS topic deleted.' TYPE 'I'.  
CATCH /aws1/cx_snsnotfoundexception.  
    MESSAGE 'Topic does not exist.' TYPE 'E'.  
ENDTRY.
```

- Para obter detalhes da API, consulte a [DeleteTopic](#) referência da API AWS SDK for SAP ABAP.

GetTopicAttributes

O código de exemplo a seguir mostra como usar GetTopicAttributes.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.  
    oo_result = lo_sns->gettopicattributes( iv_topicarn = iv_topic_arn ). "  
oo_result is returned for testing purposes. "  
    DATA(lt_attributes) = oo_result->get_attributes( ).  
    MESSAGE 'Retrieved attributes/properties of a topic.' TYPE 'I'.  
CATCH /aws1/cx_snsnotfoundexception.  
    MESSAGE 'Topic does not exist.' TYPE 'E'.  
ENDTRY.
```

- Para obter detalhes da API, consulte a [GetTopicAttributes](#) referência da API AWS SDK for SAP ABAP.

ListSubscriptions

O código de exemplo a seguir mostra como usar ListSubscriptions.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.  
    oo_result = lo_sns->listsubscriptions( ).           " oo_result is  
returned for testing purposes. "  
    DATA(lt_subscriptions) = oo_result->get_subscriptions( ).  
    MESSAGE 'Retrieved list of subscribers.' TYPE 'I'.  
    CATCH /aws1/cx_rt_generic.  
        MESSAGE 'Unable to list subscribers.' TYPE 'E'.  
ENDTRY.
```

- Para obter detalhes da API, consulte a [ListSubscriptions](#) referência da API AWS SDK for SAP ABAP.

ListTopics

O código de exemplo a seguir mostra como usar ListTopics.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.  
    oo_result = lo_sns->listtopics( ).           " oo_result is returned for  
testing purposes. "  
    DATA(lt_topics) = oo_result->get_topics( ).  
    MESSAGE 'Retrieved list of topics.' TYPE 'I'.  
    CATCH /aws1/cx_rt_generic.  
        MESSAGE 'Unable to list topics.' TYPE 'E'.  
ENDTRY.
```

- Para obter detalhes da API, consulte a [ListTopics](#) referência da API AWS SDK for SAP ABAP.

Publish

O código de exemplo a seguir mostra como usar Publish.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.  
    oo_result = lo_sns->publish(                 " oo_result is returned for  
testing purposes. "  
        iv_topicarn = iv_topic_arn  
        iv_message = iv_message  
    ).  
    MESSAGE 'Message published to SNS topic.' TYPE 'I'.  
    CATCH /aws1/cx_snsnotfoundexception.  
        MESSAGE 'Topic does not exist.' TYPE 'E'.  
ENDTRY.
```

- Para obter os detalhes da API, consulte [Publicar](#) na Referência da API AWS SDK para SAP ABAP.

SetTopicAttributes

O código de exemplo a seguir mostra como usar SetTopicAttributes.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.  
  lo_sns->settopicattributes(  
    iv_topicarn = iv_topic_arn  
    iv_attributename = iv_attribute_name  
    iv_attributevalue = iv_attribute_value  
  ).  
  MESSAGE 'Set/updated SNS topic attributes.' TYPE 'I'.  
CATCH /aws1/cx_snsnotfoundexception.  
  MESSAGE 'Topic does not exist.' TYPE 'E'.  
ENDTRY.
```

- Para obter detalhes da API, consulte a [SetTopicAttributes](#) referência da API AWS SDK for SAP ABAP.

Subscribe

O código de exemplo a seguir mostra como usar Subscribe.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

Inscreva um endereço de e-mail em um tópico.

```
TRY.
    oo_result = lo_sns->subscribe(
for testing purposes."
        iv_topicarn = iv_topic_arn
        iv_protocol = 'email'
        iv_endpoint = iv_email_address
        iv_returnsubscriptionarn = abap_true
    ).
    MESSAGE 'Email address subscribed to SNS topic.' TYPE 'I'.
CATCH /aws1/cx_snsnotfoundexception.
    MESSAGE 'Topic does not exist.' TYPE 'E'.
CATCH /aws1/cx_snssubscriptionlmte00.
    MESSAGE 'Unable to create subscriptions. You have reached the maximum number
of subscriptions allowed.' TYPE 'E'.
ENDTRY.
```

- Para obter detalhes da API, consulte [Assinar](#) na Referência da API AWS SDK para SAP ABAP.

Unsubscribe

O código de exemplo a seguir mostra como usar Unsubscribe.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.
    lo_sns->unsubscribe( iv_subscriptionarn = iv_subscription_arn ).
    MESSAGE 'Subscription deleted.' TYPE 'I'.
CATCH /aws1/cx_snsnotfoundexception.
    MESSAGE 'Subscription does not exist.' TYPE 'E'.
CATCH /aws1/cx_snsinvalidparameterex.
    MESSAGE 'Subscription with "PendingConfirmation" status cannot be deleted/
unsubscribed. Confirm subscription before performing unsubscribe operation.' TYPE
'E'.
ENDTRY.
```

- Para obter detalhes da API, consulte [Cancelar assinatura](#) na Referência da API AWS SDK para SAP ABAP.

Cenários

Criar e publicar em um tópico FIFO

O exemplo de código a seguir mostra como criar e publicar em um tópico FIFO do Amazon SNS.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

Crie um tópico FIFO, inscreva uma fila FIFO do Amazon SQS no tópico e publique uma mensagem em um tópico do Amazon SNS.

```

" Creates a FIFO topic. "
DATA lt_tpc_attributes TYPE /aws1/cl_snstopicattrsmmap_w=>tt_topicattributesmap.
DATA ls_tpc_attributes TYPE /aws1/
cl_snstopicattrsmmap_w=>ts_topicattributesmap_maprow.
ls_tpc_attributes-key = 'FifoTopic'.
ls_tpc_attributes-value = NEW /aws1/cl_snstopicattrsmmap_w( iv_value = 'true' ).
INSERT ls_tpc_attributes INTO TABLE lt_tpc_attributes.

TRY.
  DATA(lo_create_result) = lo_sns->createtopic(
    iv_name = iv_topic_name
    it_attributes = lt_tpc_attributes
  ).
  DATA(lv_topic_arn) = lo_create_result->get_topicarn( ).
  ov_topic_arn = lv_topic_arn.
ov_topic_arn is returned for testing purposes. "
  MESSAGE 'FIFO topic created' TYPE 'I'.
CATCH /aws1/cx_snstopiclimitexcdex.

```

```

    MESSAGE 'Unable to create more topics. You have reached the maximum number
of topics allowed.' TYPE 'E'.
    ENDRY.

```

```

    " Subscribes an endpoint to an Amazon Simple Notification Service (Amazon SNS)
topic. "

```

```

    " Only Amazon Simple Queue Service (Amazon SQS) FIFO queues can be subscribed to
an SNS FIFO topic. "

```

```

    TRY.

```

```

        DATA(lo_subscribe_result) = lo_sns->subscribe(
            iv_topicarn = lv_topic_arn
            iv_protocol = 'sqs'
            iv_endpoint = iv_queue_arn
        ).

```

```

        DATA(lv_subscription_arn) = lo_subscribe_result->get_subscriptionarn( ).

```

```

        ov_subscription_arn = lv_subscription_arn.

```

```

    ov_subscription_arn is returned for testing purposes. "

```

```

    MESSAGE 'SQS queue was subscribed to SNS topic.' TYPE 'I'.

```

```

    CATCH /aws1/cx_snsnotfoundexception.

```

```

    MESSAGE 'Topic does not exist.' TYPE 'E'.

```

```

    CATCH /aws1/cx_snssubscriptionlmt00.

```

```

    MESSAGE 'Unable to create subscriptions. You have reached the maximum number
of subscriptions allowed.' TYPE 'E'.

```

```

    ENDRY.

```

```

    " Publish message to SNS topic. "

```

```

    TRY.

```

```

        DATA lt_msg_attributes TYPE /aws1/
cl_snsmessageattrvalue=>tt_messageattributemap.

```

```

        DATA ls_msg_attributes TYPE /aws1/
cl_snsmessageattrvalue=>ts_messageattributemap_maprow.

```

```

        ls_msg_attributes-key = 'Importance'.

```

```

        ls_msg_attributes-value = NEW /aws1/cl_snsmessageattrvalue( iv_datatype =
'String' iv_stringvalue = 'High' ).

```

```

        INSERT ls_msg_attributes INTO TABLE lt_msg_attributes.

```

```

        DATA(lo_result) = lo_sns->publish(

```

```

            iv_topicarn = lv_topic_arn

```

```

            iv_message = 'The price of your mobile plan has been increased from $19
to $23'

```

```

            iv_subject = 'Changes to mobile plan'

```

```

            iv_messagegroupid = 'Update-2'

```

```

            iv_messagededuplicationid = 'Update-2.1'

```

```

            it_messageattributes = lt_msg_attributes

```

```
    ).  
    ov_message_id = lo_result->get_messageid( ).  
ov_message_id is returned for testing purposes. "  
    MESSAGE 'Message was published to SNS topic.' TYPE 'I'.  
    CATCH /aws1/cx_snsnotfoundexception.  
    MESSAGE 'Topic does not exist.' TYPE 'E'.  
ENDTRY.
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK para SAP ABAP.
 - [CreateTopic](#)
 - [Publicar](#)
 - [Assinar](#)

Exemplos do Amazon SQS usando o SDK para SAP ABAP

Os exemplos de código a seguir mostram como realizar ações e implementar cenários comuns usando o AWS SDK para SAP ABAP com o Amazon SQS.

Ações são trechos de código de programas maiores e devem ser executadas em contexto. Embora as ações mostrem como chamar funções de serviço específicas, é possível ver as ações contextualizadas em seus devidos cenários e exemplos entre serviços.

Cenários são exemplos de código que mostram como realizar uma tarefa específica chamando várias funções dentro do mesmo serviço.

Cada exemplo inclui um link para GitHub, onde você pode encontrar instruções sobre como configurar e executar o código no contexto.

Tópicos

- [Ações](#)
- [Cenários](#)

Ações

CreateQueue

O código de exemplo a seguir mostra como usar CreateQueue.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

Crie uma fila padrão do Amazon SQS.

```
TRY.
    oo_result = lo_sqs->createqueue( iv_queue_name = iv_queue_name ).
    oo_result is returned for testing purposes. "
    MESSAGE 'SQS queue created.' TYPE 'I'.
    CATCH /aws1/cx_sqsqueuedeletedrecently.
        MESSAGE 'After deleting a queue, wait 60 seconds before creating another
        queue with the same name.' TYPE 'E'.
    CATCH /aws1/cx_sqsqueueexists.
        MESSAGE 'A queue with this name already exists.' TYPE 'E'.
ENDTRY.
```

Crie uma fila do Amazon SQS que aguarda a chegada de uma mensagem.

```
TRY.
    DATA lt_attributes TYPE /aws1/cl_sqsqueueattrmap_w=>tt_queueattributemap.
    DATA ls_attribute TYPE /aws1/
    cl_sqsqueueattrmap_w=>ts_queueattributemap_maprow.
    ls_attribute-key = 'ReceiveMessageWaitTimeSeconds'.
    " Time in
    seconds for long polling, such as how long the call waits for a message to arrive
    in the queue before returning. "
    ls_attribute-value = NEW /aws1/cl_sqsqueueattrmap_w( iv_value =
    iv_wait_time ).
    INSERT ls_attribute INTO TABLE lt_attributes.
    oo_result = lo_sqs->createqueue(
    " oo_result is returned
    for testing purposes. "
```

```
        iv_queuename = iv_queue_name
        it_attributes = lt_attributes
    ).
    MESSAGE 'SQS queue created.' TYPE 'I'.
    CATCH /aws1/cx_sqsqueuecreatedrecently.
        MESSAGE 'After deleting a queue, wait 60 seconds before creating another
queue with the same name.' TYPE 'E'.
    CATCH /aws1/cx_sqsqueueexists.
        MESSAGE 'A queue with this name already exists.' TYPE 'E'.
    ENDTRY.
```

- Para obter detalhes da API, consulte a [CreateQueue](#) referência da API AWS SDK for SAP ABAP.

DeleteQueue

O código de exemplo a seguir mostra como usar DeleteQueue.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.
    lo_sqs->deletequeue( iv_queueurl = iv_queue_url ).
    MESSAGE 'SQS queue deleted' TYPE 'I'.
ENDTRY.
```

- Para obter detalhes da API, consulte a [DeleteQueue](#) referência da API AWS SDK for SAP ABAP.

GetQueueUrl

O código de exemplo a seguir mostra como usar GetQueueUrl.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.  
    oo_result = lo_sqs->getqueueurl( iv_queue_name = iv_queue_name ).      "  
oo_result is returned for testing purposes. "  
    MESSAGE 'Queue URL retrieved.' TYPE 'I'.  
    CATCH /aws1/cx_sqsqueue_does_not_exist.  
        MESSAGE 'The requested queue does not exist.' TYPE 'E'.  
ENDTRY.
```

- Para obter detalhes da API, consulte a [GetQueueUrl](#) referência da API AWS SDK for SAP ABAP.

ListQueues

O código de exemplo a seguir mostra como usar ListQueues.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.  
    oo_result = lo_sqs->listqueues( ).      " oo_result is returned for  
testing purposes. "  
    MESSAGE 'Retrieved list of queues.' TYPE 'I'.  
ENDTRY.
```

- Para obter detalhes da API, consulte a [ListQueues](#) referência da API AWS SDK for SAP ABAP.

ReceiveMessage

O código de exemplo a seguir mostra como usar ReceiveMessage.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

Receba uma mensagem de uma fila do Amazon SQS.

```
TRY.
    oo_result = lo_sqs->receivemessage( iv_queueurl = iv_queue_url ).    "
oo_result is returned for testing purposes. "
    DATA(lt_messages) = oo_result->get_messages( ).
    MESSAGE 'Message received from SQS queue.' TYPE 'I'.
CATCH /aws1/cx_sqsoverlimit.
    MESSAGE 'Maximum number of in-flight messages reached.' TYPE 'E'.
ENDTRY.
```

Receba uma mensagem de uma fila do Amazon SQS usando o suporte de sondagem longa.

```
TRY.
    oo_result = lo_sqs->receivemessage(                                " oo_result is returned for
testing purposes. "
        iv_queueurl = iv_queue_url
        iv_waittimeseconds = iv_wait_time    " Time in seconds for long
polling, such as how long the call waits for a message to arrive in the queue
before returning. "
    ).
    DATA(lt_messages) = oo_result->get_messages( ).
    MESSAGE 'Message received from SQS queue.' TYPE 'I'.
CATCH /aws1/cx_sqsoverlimit.
    MESSAGE 'Maximum number of in-flight messages reached.' TYPE 'E'.
ENDTRY.
```

- Para obter detalhes da API, consulte a [ReceiveMessage](#) referência da API AWS SDK for SAP ABAP.

SendMessage

O código de exemplo a seguir mostra como usar SendMessage.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
TRY.  
    oo_result = lo_sqs->sendmessage(           " oo_result is returned for  
testing purposes. "  
        iv_queueurl = iv_queue_url  
        iv_messagebody = iv_message  
    ).  
    MESSAGE 'Message sent to SQS queue.' TYPE 'I'.  
    CATCH /aws1/cx_sqsinvalidmsgconts.  
        MESSAGE 'Message contains non-valid characters.' TYPE 'E'.  
    CATCH /aws1/cx_sqsunsupportedop.  
        MESSAGE 'Operation not supported.' TYPE 'E'.  
ENDTRY.
```


- Para obter detalhes da API, consulte a [SendMessage](#) referência da API AWS SDK for SAP ABAP.

Cenários

Criar e publicar em um tópico FIFO

O exemplo de código a seguir mostra como criar e publicar em um tópico FIFO do Amazon SNS.

SDK para SAP ABAP

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

Crie um tópico FIFO, inscreva uma fila FIFO do Amazon SQS no tópico e publique uma mensagem em um tópico do Amazon SNS.

```

" Creates a FIFO topic. "
DATA lt_tpc_attributes TYPE /aws1/cl_snstopicattrsmw=>tt_topicattributesmap.
DATA ls_tpc_attributes TYPE /aws1/
cl_snstopicattrsmw=>ts_topicattributesmap_maprow.
ls_tpc_attributes-key = 'FifoTopic'.
ls_tpc_attributes-value = NEW /aws1/cl_snstopicattrsmw( iv_value = 'true' ).
INSERT ls_tpc_attributes INTO TABLE lt_tpc_attributes.

TRY.
  DATA(lo_create_result) = lo_sns->createtopic(
    iv_name = iv_topic_name
    it_attributes = lt_tpc_attributes
  ).
  DATA(lv_topic_arn) = lo_create_result->get_topicarn( ).
  ov_topic_arn = lv_topic_arn.
ov_topic_arn is returned for testing purposes. "
  MESSAGE 'FIFO topic created' TYPE 'I'.
CATCH /aws1/cx_snstopiclimitexcdex.
  MESSAGE 'Unable to create more topics. You have reached the maximum number
of topics allowed.' TYPE 'E'.
ENDTRY.

" Subscribes an endpoint to an Amazon Simple Notification Service (Amazon SNS)
topic. "
" Only Amazon Simple Queue Service (Amazon SQS) FIFO queues can be subscribed to
an SNS FIFO topic. "
TRY.
  DATA(lo_subscribe_result) = lo_sns->subscribe(
    iv_topicarn = lv_topic_arn
    iv_protocol = 'sqs'
  )

```

```

        iv_endpoint = iv_queue_arn
    ).
    DATA(lv_subscription_arn) = lo_subscribe_result->get_subscriptionarn( ).
    ov_subscription_arn = lv_subscription_arn.
ov_subscription_arn is returned for testing purposes. "
    MESSAGE 'SQS queue was subscribed to SNS topic.' TYPE 'I'.
    CATCH /aws1/cx_snsnotfoundexception.
    MESSAGE 'Topic does not exist.' TYPE 'E'.
    CATCH /aws1/cx_snssubscriptionlmt00.
    MESSAGE 'Unable to create subscriptions. You have reached the maximum number
of subscriptions allowed.' TYPE 'E'.
ENDTRY.

" Publish message to SNS topic. "
TRY.
    DATA lt_msg_attributes TYPE /aws1/
cl_snsmessageattrvalue=>tt_messageattributemap.
    DATA ls_msg_attributes TYPE /aws1/
cl_snsmessageattrvalue=>ts_messageattributemap_maprow.
    ls_msg_attributes-key = 'Importance'.
    ls_msg_attributes-value = NEW /aws1/cl_snsmessageattrvalue( iv_datatype =
'String' iv_stringvalue = 'High' ).
    INSERT ls_msg_attributes INTO TABLE lt_msg_attributes.

    DATA(lo_result) = lo_sns->publish(
        iv_topicarn = lv_topic_arn
        iv_message = 'The price of your mobile plan has been increased from $19
to $23'
        iv_subject = 'Changes to mobile plan'
        iv_messagegroupid = 'Update-2'
        iv_messagededuplicationid = 'Update-2.1'
        it_messageattributes = lt_msg_attributes
    ).
    ov_message_id = lo_result->get_messageid( ).
ov_message_id is returned for testing purposes. "
    MESSAGE 'Message was published to SNS topic.' TYPE 'I'.
    CATCH /aws1/cx_snsnotfoundexception.
    MESSAGE 'Topic does not exist.' TYPE 'E'.
ENDTRY.

```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK para SAP ABAP.
 - [CreateTopic](#)
 - [Publicar](#)
 - [Assinar](#)

Exemplos do Amazon Textract usando o SDK para SAP ABAP

Os exemplos de código a seguir mostram como realizar ações e implementar cenários comuns usando o AWS SDK para SAP ABAP com o Amazon Textract.

Ações são trechos de código de programas maiores e devem ser executadas em contexto. Embora as ações mostrem como chamar funções de serviço específicas, é possível ver as ações contextualizadas em seus devidos cenários e exemplos entre serviços.

Cenários são exemplos de código que mostram como realizar uma tarefa específica chamando várias funções dentro do mesmo serviço.

Cada exemplo inclui um link para GitHub, onde você pode encontrar instruções sobre como configurar e executar o código no contexto.

Tópicos

- [Ações](#)
- [Cenários](#)

Ações

AnalyzeDocument

O código de exemplo a seguir mostra como usar `AnalyzeDocument`.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
"Detects text and additional elements, such as forms or tables,"
"in a local image file or from in-memory byte data."
"The image must be in PNG or JPG format."
```

```
"Create ABAP objects for feature type."
"Add TABLES to return information about the tables."
"Add FORMS to return detected form data."
"To perform both types of analysis, add TABLES and FORMS to FeatureTypes."
```

```
DATA(lt_featuretypes) = VALUE /aws1/cl_texfeaturetypes_w=>tt_featuretypes(
  ( NEW /aws1/cl_texfeaturetypes_w( iv_value = 'FORMS' ) )
  ( NEW /aws1/cl_texfeaturetypes_w( iv_value = 'TABLES' ) ) ).
```

```
"Create an ABAP object for the Amazon Simple Storage Service (Amazon S3)
object."
```

```
DATA(lo_s3object) = NEW /aws1/cl_texs3object( iv_bucket = iv_s3bucket
  iv_name      = iv_s3object ).
```

```
"Create an ABAP object for the document."
```

```
DATA(lo_document) = NEW /aws1/cl_texdocument( io_s3object = lo_s3object ).
```

```
"Analyze document stored in Amazon S3."
```

```
TRY.
  oo_result = lo_tex->analyzedocument(      "oo_result is returned for testing
purposes."
  io_document      = lo_document
  it_featuretypes = lt_featuretypes ).
  LOOP AT oo_result->get_blocks( ) INTO DATA(lo_block).
  IF lo_block->get_text( ) = 'INGREDIENTS: POWDERED SUGAR* (CANE SUGAR, '.
    MESSAGE 'Found text in the doc: ' && lo_block->get_text( ) TYPE 'I'.
  ENDIF.
ENDLOOP.
MESSAGE 'Analyze document completed.' TYPE 'I'.
CATCH /aws1/cx_texaccessdeniedex.
  MESSAGE 'You do not have permission to perform this action.' TYPE 'E'.
CATCH /aws1/cx_texbaddocumentex.
  MESSAGE 'Amazon Textract is not able to read the document.' TYPE 'E'.
CATCH /aws1/cx_texdocumenttoolargeex.
  MESSAGE 'The document is too large.' TYPE 'E'.
CATCH /aws1/cx_texhlquotaexceededex.
  MESSAGE 'Human loop quota exceeded.' TYPE 'E'.
```

```
CATCH /aws1/cx_texinternalservererr.  
  MESSAGE 'Internal server error.' TYPE 'E'.  
CATCH /aws1/cx_texinvalidparameterex.  
  MESSAGE 'Request has non-valid parameters.' TYPE 'E'.  
  
CATCH /aws1/cx_texinvalids3objectex.  
  MESSAGE 'Amazon S3 object is not valid.' TYPE 'E'.  
CATCH /aws1/cx_texprovthruputexcdex.  
  MESSAGE 'Provisioned throughput exceeded limit.' TYPE 'E'.  
CATCH /aws1/cx_texthrottlingex.  
  MESSAGE 'The request processing exceeded the limit.' TYPE 'E'.  
CATCH /aws1/cx_texunsupporteddocex.  
  MESSAGE 'The document is not supported.' TYPE 'E'.  
ENDTRY.
```

- Para obter detalhes da API, consulte a [AnalyzeDocument](#) referência da API AWS SDK for SAP ABAP.

DetectDocumentText

O código de exemplo a seguir mostra como usar DetectDocumentText.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
"Detects text in the input document."  
"Amazon Textract can detect lines of text and the words that make up a line of  
text."  
"The input document must be in one of the following image formats: JPEG, PNG,  
PDF, or TIFF."  
  
"Create an ABAP object for the Amazon S3 object."  
DATA(lo_s3object) = NEW /aws1/cl_texs3object( iv_bucket = iv_s3bucket  
  iv_name      = iv_s3object ).
```

```

"Create an ABAP object for the document."
DATA(lo_document) = NEW /aws1/cl_texdocument( io_s3object = lo_s3object ).
"Analyze document stored in Amazon S3."
TRY.
    oo_result = lo_tex->detectdocumenttext( io_document = lo_document ).
"oo_result is returned for testing purposes."
    LOOP AT oo_result->get_blocks( ) INTO DATA(lo_block).
        IF lo_block->get_text( ) = 'INGREDIENTS: POWDERED SUGAR* (CANE SUGAR, '.
            MESSAGE 'Found text in the doc: ' && lo_block->get_text( ) TYPE 'I'.
        ENDIF.
    ENDLLOOP.
    DATA(lo_metadata) = oo_result->get_documentmetadata( ).
    MESSAGE 'The number of pages in the document is ' && lo_metadata->ask_pages( ) TYPE 'I'.
    MESSAGE 'Detect document text completed.' TYPE 'I'.
    CATCH /aws1/cx_texaccessdeniedex.
        MESSAGE 'You do not have permission to perform this action.' TYPE 'E'.
    CATCH /aws1/cx_texbaddocumentex.
        MESSAGE 'Amazon Textract is not able to read the document.' TYPE 'E'.
    CATCH /aws1/cx_texdocumenttoolargeex.
        MESSAGE 'The document is too large.' TYPE 'E'.
    CATCH /aws1/cx_texinternalservererr.
        MESSAGE 'Internal server error.' TYPE 'E'.
    CATCH /aws1/cx_texinvalidparameterex.
        MESSAGE 'Request has non-valid parameters.' TYPE 'E'.
    CATCH /aws1/cx_texinvalids3objectex.
        MESSAGE 'Amazon S3 object is not valid.' TYPE 'E'.
    CATCH /aws1/cx_texprovthruputexcdex.
        MESSAGE 'Provisioned throughput exceeded limit.' TYPE 'E'.
    CATCH /aws1/cx_texthrottlingex.
        MESSAGE 'The request processing exceeded the limit' TYPE 'E'.
    CATCH /aws1/cx_texunsupporteddocex.
        MESSAGE 'The document is not supported.' TYPE 'E'.
ENDTRY.


```

- Para obter detalhes da API, consulte a [DetectDocumentText](#) referência da API AWS SDK for SAP ABAP.

GetDocumentAnalysis

O código de exemplo a seguir mostra como usar GetDocumentAnalysis.

SDK para SAP ABAP

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```

"Gets the results for an Amazon Textract"
"asynchronous operation that analyzes text in a document."
TRY.
    oo_result = lo_tex->getdocumentanalysis( iv_jobid = iv_jobid ).
"oo_result is returned for testing purposes."
    WHILE oo_result->get_jobstatus( ) <> 'SUCCEEDED'.
        IF sy-index = 10.
            EXIT.                "Maximum 300 seconds.
        ENDIF.
        WAIT UP TO 30 SECONDS.
        oo_result = lo_tex->getdocumentanalysis( iv_jobid = iv_jobid ).
    ENDWHILE.

DATA(lt_blocks) = oo_result->get_blocks( ).
LOOP AT lt_blocks INTO DATA(lo_block).
    IF lo_block->get_text( ) = 'INGREDIENTS: POWDERED SUGAR* (CANE SUGAR, '.
        MESSAGE 'Found text in the doc: ' && lo_block->get_text( ) TYPE 'I'.
    ENDIF.
ENDLOOP.
MESSAGE 'Document analysis retrieved.' TYPE 'I'.
CATCH /aws1/cx_texaccessdeniedex.
    MESSAGE 'You do not have permission to perform this action.' TYPE 'E'.
CATCH /aws1/cx_texinternalservererr.
    MESSAGE 'Internal server error.' TYPE 'E'.
CATCH /aws1/cx_texinvalidjobidex.
    MESSAGE 'Job ID is not valid.' TYPE 'E'.
CATCH /aws1/cx_texinvalidkmskeyex.
    MESSAGE 'AWS KMS key is not valid.' TYPE 'E'.
CATCH /aws1/cx_texinvalidparameterex.
    MESSAGE 'Request has non-valid parameters.' TYPE 'E'.
CATCH /aws1/cx_texinvalids3objectex.
    MESSAGE 'Amazon S3 object is not valid.' TYPE 'E'.
CATCH /aws1/cx_texprovthruputexcdex.

```

```

    MESSAGE 'Provisioned throughput exceeded limit.' TYPE 'E'.
    CATCH /aws1/cx_textthrottlingex.
    MESSAGE 'The request processing exceeded the limit.' TYPE 'E'.
    ENDRTRY.

```

- Para obter detalhes da API, consulte a [GetDocumentAnalysis](#) referência da API AWS SDK for SAP ABAP.

StartDocumentAnalysis

O código de exemplo a seguir mostra como usar StartDocumentAnalysis.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

"Starts the asynchronous analysis of an input document for relationships
"between detected items such as key-value pairs, tables, and selection
elements."

"Create ABAP objects for feature type."

"Add TABLES to return information about the tables."

"Add FORMS to return detected form data."

"To perform both types of analysis, add TABLES and FORMS to FeatureTypes."

```

DATA(lt_featuretypes) = VALUE /aws1/cl_texfeaturetypes_w=>tt_featuretypes(
  ( NEW /aws1/cl_texfeaturetypes_w( iv_value = 'FORMS' ) )
  ( NEW /aws1/cl_texfeaturetypes_w( iv_value = 'TABLES' ) ) ).

```

"Create an ABAP object for the Amazon S3 object."

```

DATA(lo_s3object) = NEW /aws1/cl_texs3object( iv_bucket = iv_s3bucket
  iv_name = iv_s3object ).

```

"Create an ABAP object for the document."

```

DATA(lo_documentlocation) = NEW /aws1/cl_texdocumentlocation( io_s3object =
  lo_s3object ).

```

```
"Start async document analysis."
TRY.
    oo_result = lo_tex->startdocumentanalysis(      "oo_result is returned for
testing purposes."
        io_documentlocation      = lo_documentlocation
        it_featuretypes          = lt_featuretypes ).
    DATA(lv_jobid) = oo_result->get_jobid( ).

    MESSAGE 'Document analysis started.' TYPE 'I'.
CATCH /aws1/cx_texaccessdeniedex.
    MESSAGE 'You do not have permission to perform this action.' TYPE 'E'.
CATCH /aws1/cx_texbaddocumentex.
    MESSAGE 'Amazon Textract is not able to read the document.' TYPE 'E'.
CATCH /aws1/cx_texdocumenttoolargeex.
    MESSAGE 'The document is too large.' TYPE 'E'.
CATCH /aws1/cx_texidempotentprmmis00.
    MESSAGE 'Idempotent parameter mismatch exception.' TYPE 'E'.
CATCH /aws1/cx_texinternalservererr.
    MESSAGE 'Internal server error.' TYPE 'E'.
CATCH /aws1/cx_texinvalidkmskeyex.
    MESSAGE 'AWS KMS key is not valid.' TYPE 'E'.
CATCH /aws1/cx_texinvalidparameterex.
    MESSAGE 'Request has non-valid parameters.' TYPE 'E'.
CATCH /aws1/cx_texinvalids3objectex.
    MESSAGE 'Amazon S3 object is not valid.' TYPE 'E'.
CATCH /aws1/cx_texlimitexceeddex.
    MESSAGE 'An Amazon Textract service limit was exceeded.' TYPE 'E'.
CATCH /aws1/cx_texprovthruputexcdex.
    MESSAGE 'Provisioned throughput exceeded limit.' TYPE 'E'.
CATCH /aws1/cx_texthrottlingex.
    MESSAGE 'The request processing exceeded the limit.' TYPE 'E'.
CATCH /aws1/cx_texunsupporteddocex.
    MESSAGE 'The document is not supported.' TYPE 'E'.
ENDTRY.
```

- Para obter detalhes da API, consulte a [StartDocumentAnalysis](#) referência da API AWS SDK for SAP ABAP.

StartDocumentTextDetection

O código de exemplo a seguir mostra como usar StartDocumentTextDetection.

SDK para SAP ABAP

 Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```

"Starts the asynchronous detection of text in a document."
"Amazon Textract can detect lines of text and the words that make up a line of
text."

"Create an ABAP object for the Amazon S3 object."
DATA(lo_s3object) = NEW /aws1/cl_texs3object( iv_bucket = iv_s3bucket
  iv_name      = iv_s3object ).
"Create an ABAP object for the document."
DATA(lo_documentlocation) = NEW /aws1/cl_texdocumentlocation( io_s3object =
lo_s3object ).
"Start document analysis."
TRY.
  oo_result = lo_tex->startdocumenttextdetection( io_documentlocation =
lo_documentlocation ).
  DATA(lv_jobid) = oo_result->get_jobid( ).           "oo_result is returned
for testing purposes."
  MESSAGE 'Document analysis started.' TYPE 'I'.
  CATCH /aws1/cx_texaccessdeniedex.
  MESSAGE 'You do not have permission to perform this action.' TYPE 'E'.
  CATCH /aws1/cx_texbaddocumentex.
  MESSAGE 'Amazon Textract is not able to read the document.' TYPE 'E'.
  CATCH /aws1/cx_texdocumenttoolargeex.
  MESSAGE 'The document is too large.' TYPE 'E'.
  CATCH /aws1/cx_texidempotentprmmis00.
  MESSAGE 'Idempotent parameter mismatch exception.' TYPE 'E'.
  CATCH /aws1/cx_texinternalservererr.
  MESSAGE 'Internal server error.' TYPE 'E'.
  CATCH /aws1/cx_texinvalidkmskeyex.
  MESSAGE 'AWS KMS key is not valid.' TYPE 'E'.
  CATCH /aws1/cx_texinvalidparameterex.
  MESSAGE 'Request has non-valid parameters.' TYPE 'E'.
  CATCH /aws1/cx_texinvalids3objectex.
  MESSAGE 'Amazon S3 object is not valid.' TYPE 'E'.

```

```
CATCH /aws1/cx_texlimitexceeddex.  
  MESSAGE 'An Amazon Textract service limit was exceeded.' TYPE 'E'.  
CATCH /aws1/cx_texprovthruputexcdex.  
  MESSAGE 'Provisioned throughput exceeded limit.' TYPE 'E'.  
CATCH /aws1/cx_texthrottlingex.  
  MESSAGE 'The request processing exceeded the limit.' TYPE 'E'.  
CATCH /aws1/cx_texunsupporteddocex.  
  MESSAGE 'The document is not supported.' TYPE 'E'.  
ENDTRY.
```

- Para obter detalhes da API, consulte a [StartDocumentTextDetection](#) referência da API AWS SDK for SAP ABAP.

Cenários

Conceitos básicos de usar análise de documentos

O exemplo de código a seguir mostra como:

- Iniciar análise assíncrona.
- Obter análise de documento.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
"Create ABAP objects for feature type."  
"Add TABLES to return information about the tables."  
"Add FORMS to return detected form data."  
"To perform both types of analysis, add TABLES and FORMS to FeatureTypes."  
  
DATA(lt_featuretypes) = VALUE /aws1/cl_texfeaturetypes_w=>tt_featuretypes(  
  ( NEW /aws1/cl_texfeaturetypes_w( iv_value = 'FORMS' ) )  
  ( NEW /aws1/cl_texfeaturetypes_w( iv_value = 'TABLES' ) ) ).
```

```
"Create an ABAP object for the Amazon Simple Storage Service (Amazon S3)
object."
DATA(lo_s3object) = NEW /aws1/cl_texs3object( iv_bucket = iv_s3bucket
      iv_name   = iv_s3object ).

"Create an ABAP object for the document."
DATA(lo_documentlocation) = NEW /aws1/cl_texdocumentlocation( io_s3object =
lo_s3object ).

"Start document analysis."
TRY.
  DATA(lo_start_result) = lo_tex->startdocumentanalysis(
    io_documentlocation   = lo_documentlocation
    it_featuretypes       = lt_featuretypes ).
  MESSAGE 'Document analysis started.' TYPE 'I'.
  CATCH /aws1/cx_texaccessdeniedex.
    MESSAGE 'You do not have permission to perform this action.' TYPE 'E'.
  CATCH /aws1/cx_texbaddocumentex.
    MESSAGE 'Amazon Textract is not able to read the document.' TYPE 'E'.
  CATCH /aws1/cx_texdocumenttoolargeex.
    MESSAGE 'The document is too large.' TYPE 'E'.
  CATCH /aws1/cx_texidempotentprmmis00.
    MESSAGE 'Idempotent parameter mismatch exception.' TYPE 'E'.
  CATCH /aws1/cx_texinternalservererr.
    MESSAGE 'Internal server error.' TYPE 'E'.
  CATCH /aws1/cx_texinvalidkmskeyex.
    MESSAGE 'AWS KMS key is not valid.' TYPE 'E'.
  CATCH /aws1/cx_texinvalidparameterex.
    MESSAGE 'Request has non-valid parameters.' TYPE 'E'.
  CATCH /aws1/cx_texinvalids3objectex.
    MESSAGE 'Amazon S3 object is not valid.' TYPE 'E'.
  CATCH /aws1/cx_texlimitexceededex.
    MESSAGE 'An Amazon Textract service limit was exceeded.' TYPE 'E'.
  CATCH /aws1/cx_texpvthruputexcdex.
    MESSAGE 'Provisioned throughput exceeded limit.' TYPE 'E'.
  CATCH /aws1/cx_texthrottlingex.
    MESSAGE 'The request processing exceeded the limit.' TYPE 'E'.
  CATCH /aws1/cx_texunsupporteddocex.
    MESSAGE 'The document is not supported.' TYPE 'E'.
ENDTRY.

"Get job ID from the output."
DATA(lv_jobid) = lo_start_result->get_jobid( ).
```

```
"Wait for job to complete."
oo_result = lo_tex->getdocumentanalysis( iv_jobid = lv_jobid ).      " oo_result
is returned for testing purposes. "
WHILE oo_result->get_jobstatus( ) <> 'SUCCEEDED'.
  IF sy-index = 10.
    EXIT.                  "Maximum 300 seconds."
  ENDIF.
  WAIT UP TO 30 SECONDS.
  oo_result = lo_tex->getdocumentanalysis( iv_jobid = lv_jobid ).
ENDWHILE.

DATA(lt_blocks) = oo_result->get_blocks( ).
LOOP AT lt_blocks INTO DATA(lo_block).
  IF lo_block->get_text( ) = 'INGREDIENTS: POWDERED SUGAR* (CANE SUGAR, '.
    MESSAGE 'Found text in the doc: ' && lo_block->get_text( ) TYPE 'I'.
  ENDIF.
ENDLOOP.
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK para SAP ABAP.
 - [GetDocumentAnalysis](#)
 - [StartDocumentAnalysis](#)

Exemplos do Amazon Translate usando o SDK para SAP ABAP

Os exemplos de código a seguir mostram como realizar ações e implementar cenários comuns usando o AWS SDK para SAP ABAP com o Amazon Translate.

Ações são trechos de código de programas maiores e devem ser executadas em contexto. Embora as ações mostrem como chamar funções de serviço específicas, é possível ver as ações contextualizadas em seus devidos cenários e exemplos entre serviços.

Cenários são exemplos de código que mostram como realizar uma tarefa específica chamando várias funções dentro do mesmo serviço.

Cada exemplo inclui um link para GitHub, onde você pode encontrar instruções sobre como configurar e executar o código no contexto.

Tópicos

- [Ações](#)
- [Cenários](#)

Ações

DescribeTextTranslationJob

O código de exemplo a seguir mostra como usar `DescribeTextTranslationJob`.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
"Gets the properties associated with an asynchronous batch translation job."
"Includes properties such as name, ID, status, source and target languages, and
input/output Amazon Simple Storage Service (Amazon S3) buckets."
TRY.
    oo_result = lo_xl8->describetexttranslationjob(      "oo_result is returned
for testing purposes."
    EXPORTING
        iv_jobid      = iv_jobid
    ).
    MESSAGE 'Job description retrieved.' TYPE 'I'.
CATCH /aws1/cx_xl8internalserverex .
    MESSAGE 'An internal server error occurred. Retry your request.' TYPE 'E'.
CATCH /aws1/cx_xl8resourcenotfoundex .
    MESSAGE 'The resource you are looking for has not been found.' TYPE 'E'.
CATCH /aws1/cx_xl8toomanyrequestsex.
    MESSAGE 'You have made too many requests within a short period of time.'
TYPE 'E'.
ENDTRY.
```

- Para obter detalhes da API, consulte a [DescribeTextTranslationJob](#) referência da API AWS SDK for SAP ABAP.

ListTextTranslationJobs

O código de exemplo a seguir mostra como usar ListTextTranslationJobs.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
"Gets a list of the batch translation jobs that you have submitted."

DATA lo_filter TYPE REF TO /aws1/cl_xl8textxl8translationjobfilt.

"Create an ABAP object for filtering using jobname."
CREATE OBJECT lo_filter
  EXPORTING
    iv_jobname = iv_jobname.

TRY.
  oo_result = lo_xl8->listtexttranslationjobs(      "oo_result is returned for
testing purposes."
  EXPORTING
    io_filter      = lo_filter
  ).
  MESSAGE 'Jobs retrieved.' TYPE 'I'.
CATCH /aws1/cx_xl8internalserverex .
  MESSAGE 'An internal server error occurred. Retry your request.' TYPE 'E'.
CATCH /aws1/cx_xl8invalidfilterex .
  MESSAGE 'The filter specified for the operation is not valid. Specify a
different filter.' TYPE 'E'.
CATCH /aws1/cx_xl8invalidrequestex .
  MESSAGE 'The request that you made is not valid.' TYPE 'E'.
CATCH /aws1/cx_xl8toomanyrequestsex.
  MESSAGE 'You have made too many requests within a short period of time.'
TYPE 'E'.
ENDTRY.
```

- Para obter detalhes da API, consulte a [ListTextTranslationJobs](#) referência da API AWS SDK for SAP ABAP.

StartTextTranslationJob

O código de exemplo a seguir mostra como usar StartTextTranslationJob.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
"Starts an asynchronous batch translation job."
"Use batch translation jobs to translate large volumes of text across multiple
documents at once."

DATA lo_inputdataconfig TYPE REF TO /aws1/cl_xl8inputdataconfig.
DATA lo_outputdataconfig TYPE REF TO /aws1/cl_xl8outputdataconfig.
DATA lt_targetlanguagecodes TYPE /aws1/
cl_xl8tgtlanguagecodes00=>tt_targetlanguagecodestrlist.
DATA lo_targetlanguagecodes TYPE REF TO /aws1/cl_xl8tgtlanguagecodes00.

"Create an ABAP object for the input data config."
CREATE OBJECT lo_inputdataconfig
EXPORTING
    iv_s3uri      = iv_input_data_s3uri
    iv_contenttype = iv_input_data_contenttype.

"Create an ABAP object for the output data config."
CREATE OBJECT lo_outputdataconfig
EXPORTING
    iv_s3uri = iv_output_data_s3uri.

"Create an internal table for target languages."
CREATE OBJECT lo_targetlanguagecodes
EXPORTING
    iv_value = iv_targetlanguagecode.
INSERT lo_targetlanguagecodes INTO TABLE lt_targetlanguagecodes.
```

```

TRY.
    oo_result = lo_xl8->starttexttranslationjob(      "oo_result is returned for
testing purposes."
    EXPORTING
        io_inputdataconfig = lo_inputdataconfig
        io_outputdataconfig = lo_outputdataconfig
        it_targetlanguagecodes = lt_targetlanguagecodes
        iv_dataaccessrolelearn = iv_dataaccessrolelearn
        iv_jobname = iv_jobname
        iv_sourcelanguagecode = iv_sourcelanguagecode
    ).
    MESSAGE 'Translation job started.' TYPE 'I'.
    CATCH /aws1/cx_xl8internalserverex .
        MESSAGE 'An internal server error occurred. Retry your request.' TYPE 'E'.
    CATCH /aws1/cx_xl8invparamvalueex .
        MESSAGE 'The value of the parameter is not valid.' TYPE 'E'.
    CATCH /aws1/cx_xl8invalidrequestex .
        MESSAGE 'The request that you made is not valid.' TYPE 'E'.
    CATCH /aws1/cx_xl8resourcenotfoundex .
        MESSAGE 'The resource you are looking for has not been found.' TYPE 'E'.
    CATCH /aws1/cx_xl8toomanyrequestsex .
        MESSAGE 'You have made too many requests within a short period of time.'
TYPE 'E'.
    CATCH /aws1/cx_xl8unsuppdedlanguage00 .
        MESSAGE 'Amazon Translate does not support translation from the language of
the source text into the requested target language.' TYPE 'E'.
ENDTRY.

```

- Para obter detalhes da API, consulte a [StartTextTranslationJob](#) referência da API AWS SDK for SAP ABAP.

StopTextTranslationJob

O código de exemplo a seguir mostra como usar StopTextTranslationJob.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
"Stops an asynchronous batch translation job that is in progress."
```

```
TRY.  
    oo_result = lo_xl8->stoptexttranslationjob(      "oo_result is returned for  
testing purposes."  
    EXPORTING  
        iv_jobid      = iv_jobid  
    ).  
    MESSAGE 'Translation job stopped.' TYPE 'I'.  
CATCH /aws1/cx_xl8internalserverex .  
    MESSAGE 'An internal server error occurred.' TYPE 'E'.  
CATCH /aws1/cx_xl8resourcenotfoundex .  
    MESSAGE 'The resource you are looking for has not been found.' TYPE 'E'.  
CATCH /aws1/cx_xl8toomanyrequestsex.  
    MESSAGE 'You have made too many requests within a short period of time.'  
TYPE 'E'.  
ENDTRY.
```

- Para obter detalhes da API, consulte a [StopTextTranslationJob](#) referência da API AWS SDK for SAP ABAP.

TranslateText

O código de exemplo a seguir mostra como usar TranslateText.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```

"Translates input text from the source language to the target language."
TRY.
    oo_result = lo_xl8->translatetext(      "oo_result is returned for testing
purposes."
    EXPORTING
        iv_text      = iv_text
        iv_sourcelanguagecode = iv_sourcelanguagecode
        iv_targetlanguagecode = iv_targetlanguagecode
    ).
    MESSAGE 'Translation completed.' TYPE 'I'.
    CATCH /aws1/cx_xl8detectedlanguage00 .
        MESSAGE 'The confidence that Amazon Comprehend accurately detected the
source language is low.' TYPE 'E'.
    CATCH /aws1/cx_xl8internalserverex .
        MESSAGE 'An internal server error occurred.' TYPE 'E'.
    CATCH /aws1/cx_xl8invalidrequestex .
        MESSAGE 'The request that you made is not valid.' TYPE 'E'.
    CATCH /aws1/cx_xl8resourcenotfoundex .
        MESSAGE 'The resource you are looking for has not been found.' TYPE 'E'.
    CATCH /aws1/cx_xl8serviceunavailex .
        MESSAGE 'The Amazon Translate service is temporarily unavailable.' TYPE 'E'.
    CATCH /aws1/cx_xl8textsizegmtexcdex .
        MESSAGE 'The size of the text you submitted exceeds the size limit. ' TYPE
'E'.
    CATCH /aws1/cx_xl8toomanyrequestsex .
        MESSAGE 'You have made too many requests within a short period of time.'
TYPE 'E'.
    CATCH /aws1/cx_xl8unsuppdedlanguage00 .
        MESSAGE 'Amazon Translate does not support translation from the language of
the source text into the requested target language. ' TYPE 'E'.
    ENDTRY.

```

- Para obter detalhes da API, consulte a [TranslateText](#) referência da API AWS SDK for SAP ABAP.

Cenários

Conceitos básicos de usar trabalhos de tradução

O exemplo de código a seguir mostra como:

- Iniciar um trabalho assíncrono de tradução em lote.
- Aguardar a conclusão do trabalho assíncrono.
- Descrever o trabalho assíncrono.

SDK para SAP ABAP

Note

Tem mais sobre GitHub. Encontre o exemplo completo e veja como configurar e executar no [AWS Code Examples Repository](#).

```
DATA lo_inputdataconfig TYPE REF TO /aws1/cl_xl8inputdataconfig.
DATA lo_outputdataconfig TYPE REF TO /aws1/cl_xl8outputdataconfig.
DATA lt_targetlanguagecodes TYPE /aws1/
cl_xl8tgtlanguagecodes00=>tt_targetlanguagecodestrlist.
DATA lo_targetlanguagecodes TYPE REF TO /aws1/cl_xl8tgtlanguagecodes00.

"Create an ABAP object for the input data config."
CREATE OBJECT lo_inputdataconfig
  EXPORTING
    iv_s3uri          = iv_input_data_s3uri
    iv_contenttype    = iv_input_data_contenttype.

"Create an ABAP object for the output data config."
CREATE OBJECT lo_outputdataconfig
  EXPORTING
    iv_s3uri = iv_output_data_s3uri.

"Create an internal table for target languages."
CREATE OBJECT lo_targetlanguagecodes
  EXPORTING
    iv_value = iv_targetlanguagecode.
INSERT lo_targetlanguagecodes INTO TABLE lt_targetlanguagecodes.

TRY.
  DATA(lo_translationjob_result) = lo_xl8->starttexttranslationjob(
    EXPORTING
      io_inputdataconfig = lo_inputdataconfig
      io_outputdataconfig = lo_outputdataconfig
```

```

        it_targetlanguagecodes = lt_targetlanguagecodes
        iv_dataaccessrolelearn = iv_dataaccessrolelearn
        iv_jobname = iv_jobname
        iv_sourcelanguagecode = iv_sourcelanguagecode
    ).
    MESSAGE 'Translation job started.' TYPE 'I'.
    CATCH /aws1/cx_xl8internalserverex .
        MESSAGE 'An internal server error occurred. Retry your request.' TYPE 'E'.
    CATCH /aws1/cx_xl8invparamvalueex .
        MESSAGE 'The value of the parameter is not valid.' TYPE 'E'.
    CATCH /aws1/cx_xl8invalidrequestex.
        MESSAGE 'The request that you made is not valid.' TYPE 'E'.
    CATCH /aws1/cx_xl8resourcenotfoundex .
        MESSAGE 'The resource you are looking for has not been found.' TYPE 'E'.
    CATCH /aws1/cx_xl8toomanyrequestsex.
        MESSAGE 'You have made too many requests within a short period of time. '
TYPE 'E'.
    CATCH /aws1/cx_xl8unsuppedlanguage00 .
        MESSAGE 'Amazon Translate does not support translation from the language of
the source text into the requested target language.' TYPE 'E'.
    ENDTRY.

    "Get the job ID."
    DATA(lv_jobid) = lo_translationjob_result->get_jobid( ).

    "Wait for translate job to complete."
    DATA(lo_des_translation_result) = lo_xl8->describetexttranslationjob( iv_jobid =
lv_jobid ).
    WHILE lo_des_translation_result->get_textxlationjobproperties( )-
>get_jobstatus( ) <> 'COMPLETED'.
        IF sy-index = 30.
            EXIT.                "Maximum 900 seconds."
        ENDIF.
        WAIT UP TO 30 SECONDS.
        lo_des_translation_result = lo_xl8->describetexttranslationjob( iv_jobid =
lv_jobid ).
    ENDWHILE.

    TRY.
        oo_result = lo_xl8->describetexttranslationjob(      "oo_result is returned
for testing purposes."
            EXPORTING
                iv_jobid          = lv_jobid
        ).

```

```
    MESSAGE 'Job description retrieved.' TYPE 'I'.
  CATCH /aws1/cx_xl8internalserverex .
    MESSAGE 'An internal server error occurred. Retry your request.' TYPE 'E'.
  CATCH /aws1/cx_xl8resourcenotfoundex .
    MESSAGE 'The resource you are looking for has not been found.' TYPE 'E'.
  CATCH /aws1/cx_xl8toomanyrequestsex.
    MESSAGE 'You have made too many requests within a short period of time.'
TYPE 'E'.
  ENDTRY.
```

- Para obter detalhes da API, consulte os tópicos a seguir na Referência da API AWS SDK para SAP ABAP.
 - [DescribeTextTranslationJob](#)
 - [StartTextTranslationJob](#)

Segurança em SDK da AWS para SAP ABAP

A segurança na nuvem AWS é a maior prioridade. Como AWS cliente, você se beneficia de data centers e arquiteturas de rede criados para atender aos requisitos das organizações mais sensíveis à segurança.

A segurança é uma responsabilidade compartilhada entre você AWS e você. O [modelo de responsabilidade compartilhada](#) descreve isso como a segurança da nuvem e segurança na nuvem:

- **Segurança da nuvem** — AWS é responsável por proteger a infraestrutura que é executada Serviços da AWS no Nuvem AWS. AWS também fornece serviços que você pode usar com segurança. Auditores terceirizados testam e verificam regularmente a eficácia de nossa segurança como parte dos Programas de Conformidade Programas de [AWS](#) de . Para saber mais sobre os programas de conformidade aplicáveis SDK da AWS para SAP ABAP, consulte [Serviços da AWS Escopo por Programa de Conformidade Serviços da AWS em Escopo por Programa](#) .
- **Segurança na nuvem** — Sua responsabilidade é determinada pelo Serviço da AWS que você usa. Você também é responsável por outros fatores, incluindo a confidencialidade de seus dados, os requisitos da empresa e as leis e regulamentos aplicáveis.

Esta seção abrange os seguintes tópicos.

Tópicos

- [SAPautenticação do sistema ativada AWS](#)
- [Práticas recomendadas de segurança do IAM](#)
- [Autorizações da SAP](#)
- [Operações seguras](#)
- [Usar certificados com o IAM Roles Anywhere](#)
- [Usando o SAP Credential Store](#)

SAPautenticação do sistema ativada AWS

Antes que um SAP sistema possa fazer chamadas para AWS em nome dos SAP usuários, o SAP sistema deve se autenticar AWS em. SDK da AWS para SAP ABAP suporta os três métodos de autenticação a seguir, selecionados nas configurações do SDK perfil emIMG.

AWS SDK for SAP ABAP - BTP edition só pode ser autenticada com o [the section called “Autenticação da chave de acesso secreta”](#) método usando o SAP Credential Store.

Tópicos

- [Autenticação de metadados de EC2 instâncias da Amazon](#)
- [Autenticação da chave de acesso secreta](#)
- [Autenticação baseada em certificado usando Roles Anywhere IAM](#)
- [Próxima etapa](#)

Autenticação de metadados de EC2 instâncias da Amazon

SAPos sistemas executados na Amazon EC2 podem adquirir credenciais de curta duração e rotativas automáticas a partir dos metadados de instâncias da Amazon. EC2 Para obter mais informações, consulte [Uso de credenciais para metadados de EC2 instâncias da Amazon](#).

É altamente recomendável usar esse método de autenticação durante o uso do SDK for SAP ABAP. Para habilitar, o administrador do Basis deve habilitar a HTTP comunicação de saída. Nenhuma outra configuração é necessária.

Note

Esse método de autenticação se aplica somente se seus SAP sistemas estiverem em execução na Amazon EC2. SAP sistemas hospedados localmente ou em outros ambientes de nuvem não podem ser autenticados usando esse método.

Autenticação da chave de acesso secreta

Com esse método, você usa uma ID de chave de acesso e uma chave de acesso secreta para autenticar seu SAP sistema. AWS O SAP sistema faz login AWS usando um IAM usuário. Para obter mais informações, consulte [Gerenciando chaves de acesso para IAM usuários](#).

O administrador do Basis recebe uma ID de chave de acesso e uma chave de acesso secreta do AWS IAM administrador. Seu SAP sistema deve estar configurado para armazenar o ID da chave de acesso e a chave de acesso secreta.

- Proteja, armazene e encaminhe (SSF)

- Use a SSF funcionalidade para se autenticar AWS SDK. SAP ABAP Para obter mais informações, consulte [Assinaturas digitais e criptografia](#).
- Você também pode testar SSF envelope e desenvolve funcionalidades com o SSF02 relatório. Para obter mais informações, consulte [Testando a SSF instalação](#).
- As etapas de configuração do SSF SDK for SAP ABAP estão descritas na /AWS1/IMG transação. Vá para Pré-requisitos técnicos e selecione Configurações adicionais para sistemas on-premises.
- SAP Armazenamento de credenciais
 - Use o SAP Credential Store para se autenticar AWS SDK para a edição SAP ABAP. BTP Para obter mais informações, consulte [O que é o SAP Credential Store?](#)
 - Consulte [Usando o SAP Credential Store](#) para ver as etapas de configuração.

Autenticação baseada em certificado usando Roles Anywhere IAM

Um certificado X.509 emitido pela sua autoridade de certificação (CA) pode ser usado para autenticação com o AWS Identity and Access Management Roles Anywhere. O certificado deve ser configurado em STRUST. A CA deve ser registrada no IAM Roles Anywhere como uma âncora de confiança e um perfil deve ser criado para especificar as funções e políticas que o IAM Roles Anywhere assumiria. Para obter mais informações, consulte [Criação de uma âncora de confiança e um perfil no AWS Identity and Access Management Roles Anywhere](#).

Para obter etapas detalhadas sobre como usar o IAM Roles Anywhere com SDK for SAP ABAP, consulte [Usando certificados com o IAM Roles Anywhere](#).

Note

A revogação de certificado só é permitida por meio do uso de listas de revogação de certificados importadas. Consulte mais informações em [Revocation](#).

Próxima etapa

Depois de autenticar seu SAP sistema AWS, o SDK for executa SAP ABAP automaticamente e assume `sts:assumeRole` a IAM função apropriada para a função comercial do SAP usuário.

Práticas recomendadas de segurança do IAM

O administrador do IAM será responsável pelas três áreas principais a seguir.

- Garantir que o sistema SAP possa se autenticar com metadados do Amazon EC2 ou credenciais de chave secreta.
- Garantir que o sistema SAP tenha as permissões necessárias para se elevar com o `sts:assumeRole`.
- Para cada perfil lógico do IAM, criar um perfil do IAM para usuários do SAP com as permissões necessárias para realizar as funções de negócios (por exemplo, as permissões necessárias para Amazon S3, DynamoDB ou outros serviços). Essas são as funções que o usuário assumirá.

Para obter mais informações, consulte o capítulo [Segurança](#) no SAP Lens: AWS Well-Architected Framework.

Tópicos

- [Melhores práticas para perfil de instância do Amazon EC2](#)
- [Perfil do IAM para usuários do SAP](#)

Melhores práticas para perfil de instância do Amazon EC2

A instância do Amazon EC2 na qual seu sistema SAP é executado tem um conjunto de autorizações com base em seu perfil de instância. Geralmente, o perfil da instância só precisa ter permissões para chamar `sts:assumeRole`, para permitir que o sistema SAP assuma perfis do IAM específicos da empresa, conforme o necessário. Essa elevação para outras funções garante que um programa ABAP possa assumir uma função que dê ao usuário o menor privilégio necessário para realizar o trabalho. Por exemplo, um perfil de instância pode conter a seguinte declaração.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "VisualEditor0",
      "Effect": "Allow",
      "Action": "sts:AssumeRole",
      "Resource": [
```

```
        "arn:aws:iam::0123456789:role/finance-cfo",  
        "arn:aws:iam::0123456789:role/finance-auditor",  
        "arn:aws:iam::0123456789:role/finance-reporting"  
    ]  
}  
]  
}
```

Esse exemplo anterior permite que o sistema SAP assumira os perfis do IAM para o usuário CFO, AUDITOR ou REPORTING. AWS SDK escolherá o perfil correto do IAM para o usuário com base na função PFCG do usuário no SAP.

O perfil de instância do Amazon EC2 também pode ser usado para outras funções.

- [AWS Backint Agent para SAP HANA](#)
- [SAP em alta disponibilidade da AWS com roteamento de endereço IP de sobreposição](#)

Essas soluções também podem exigir permissões `sts:assumeRole` para funções específicas de backup ou failover ou podem exigir que as permissões sejam atribuídas diretamente ao perfil de instância.

Perfil do IAM para usuários do SAP

O programa ABAP precisa de permissões para realizar o trabalho do usuário: ler uma tabela do DynamoDB, invocar o Amazon Textract em um objeto PDF no Amazon S3, executar uma função AWS Lambda. O mesmo modelo de segurança é usado em todos os AWS SDKs. Você pode usar um perfil do IAM existente que foi usado para outro AWS SDK.

O analista de negócios da SAP solicitará ao administrador do IAM a função `arn:aws:` de um perfil do IAM para cada função lógica necessária. Por exemplo, em um cenário financeiro, o analista de negócios pode definir os seguintes perfis lógicos do IAM.

- CFO
- AUDITOR
- REPORTING

O administrador do IAM definirá os perfis do IAM para cada perfil lógico do IAM.

CFO

- `arn:aws:iam::0123456789:role/finance-cfo`
- permissões de gravação de dados em um bucket do Amazon S3.
- permissões de leitura e gravação em um banco de dados do DynamoDB

AUDITOR

- `arn:aws:iam::0123456789:role/finance-auditor`
- permissões de leitura em um bucket do Amazon S3
- permissões de leitura em um banco de dados do DynamoDB

REPORTING

- `arn:aws:iam::0123456789:role/finance-reporting`
- permissões de leitura em um banco de dados do DynamoDB
- nenhuma permissão para o bucket do Amazon S3

O analista de negócios inserirá os perfis do IAM em uma tabela de mapeamento para mapear os perfis lógicos do IAM com os perfis físicos do IAM.

Os perfis do IAM para usuários do SAP precisam permitir a ação `sts:assumeRole` de entidades principais confiáveis. As entidades principais confiáveis podem variar de acordo com a forma como o sistema SAP é autenticado na AWS. Para obter mais informações, consulte [Especificar uma entidade principal](#).

Veja alguns dos cenários de exemplo comuns do SAP:

- Sistema SAP em execução no Amazon EC2 com um perfil de instância atribuído: aqui, um perfil de instância do Amazon EC2 é anexado a um perfil do IAM.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "sts:AssumeRole"
      ],
      "Principal": {
        "AWS": "arn:aws:iam::123456789012:role/SapInstanceProfile"
      }
    }
  ]
}
```

```

    }
  }
]
}

```

- Sistemas SAP em execução no Amazon EC2 sem um perfil de instância; aqui o Amazon EC2 assume funções para usuários SAP.

```

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

```

- Sistemas SAP executados on-premises: os sistemas SAP executados on-premises só podem ser autenticados usando a chave de acesso secreta. Para obter mais informações, consulte [Autenticação do sistema do SAP em AWS](#).

Aqui qualquer perfil do IAM assumido por um usuário do SAP deve ter uma relação de confiança que confie no usuário do SAP.

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "sts:AssumeRole"
      ],
      "Principal": {
        "AWS": "arn:aws:iam::123456789012:user/SAP_SYSTEM_S4H"
      }
    }
  ]
}

```

```
}
```

Autorizações da SAP

A autorização necessária para configurar o SDK depende da edição do SDK.

Tópicos

- [Autorizações para configuração](#)
- [Autorizações SAP para usuários finais](#)

Autorizações para configuração

Consulte as guias a seguir para obter mais detalhes.

SDK for SAP ABAP

As seguintes autorizações são necessárias para configurar o SDK para SAP ABAP.

- S_TCODE
 - TCD = /AWS1/IMG
- S_TABU_DIS
 - ACTVT = 02, 03
 - DICBERCLS

Escolha um dos seguintes grupos de autorização.

- /AWS1/CFG- SDK da AWS para SAP ABAP v1 - Config
- /AWS1/MOD- SDK da AWS para SAP ABAP v1 - Tempo de execução
- /AWS1/PFL- SDK da AWS para SAP ABAP v1 - Perfil do SDK
- /AWS1/RES- SDK da AWS para SAP ABAP v1 - Recursos lógicos
- /AWS1/TRC- SDK da AWS para SAP ABAP v1 - Rastreamento

SDK for SAP ABAP - BTP edition

Use as etapas a seguir para permitir que o SDK for SAP ABAP - edição BTP acesse a configuração.

1. Crie uma nova função comercial usando o modelo SAP_BR_BPC_EXPERT de função comercial. Esse modelo fornece acesso ao aplicativo Custom Business Configuration.
2. Em Detalhes gerais da função, acesse Categorias de acesso e escolha Irrestrito para ajuda de leitura, gravação e valor.
3. Vá até a guia Catálogo comercial e atribua o catálogo /AWS1/RTBTP_BCAT comercial para fornecer acesso à configuração do SDK.
4. Vá até a guia Usuários corporativos e designe usuários corporativos para conceder acesso à configuração do SDK.

Autorizações SAP para usuários finais

Pré-requisito: definir perfis do SDK

Antes que o administrador de segurança da SAP possa definir suas funções, o analista de negócios definirá os perfis do SDK em transação /AWS1/IMG para o AWS SDK for SAP ABAP ou o aplicativo Custom Business Configuration for SDK for SAP ABAP - edição BTP. Normalmente, um perfil SDK será nomeado de acordo com sua função comercial: ZFINANCE, ZBILLING, ZMFG, ZPAYROLL etc. Para cada perfil do SDK, o analista de negócios definirá perfis lógicos do IAM com nomes curtos, como CFO, AUDITOR, REPORTING. Eles serão mapeados para os perfis reais do IAM pelo administrador de segurança do IAM.

Defina PFCG ou funções comerciais

Note

As funções do PFCG são chamadas de funções de negócios no ambiente SAP, BTP e ABAP.

O administrador de segurança do SAP então adicionará um objeto de autorização da /AWS1/SESS para conceder acesso a um perfil do SDK.

Objeto de autenticação da /AWS1/SESS

- Campo /AWS1/PROF = ZFINANCE

Os usuários também devem ser mapeados para perfis lógicos do IAM para cada perfil do SDK, dependendo da função do trabalho. Por exemplo, um auditor financeiro com acesso a relatórios pode ser autorizado para um perfil lógico do IAM chamado AUDITOR.

Objeto de autenticação da /AWS1/LROL

- Campo /AWS1/PROF = ZFINANCE
- Campo /AWS1/LROL = AUDITOR

Enquanto isso, o CFO, com autorizações de leitura/gravação, pode ter um perfil de PFCG autorizando-o o perfil lógico de CFO.

Objeto de autenticação da /AWS1/LROL

- Campo /AWS1/PROF = ZFINANCE
- Campo /AWS1/LROL = CFO

Em geral, um usuário deve ser autorizado para apenas um perfil lógico do IAM por perfil do SDK. Se um usuário for autorizado para mais de uma função do IAM (por exemplo, se o CFO estiver autorizado para ambas as funções CFO e funções AUDITOR lógicas do IAM), o AWS SDK rompe o empate garantindo que a função de maior prioridade (menor número de sequência) entre em vigor.

Como em todos os cenários de segurança, os usuários devem ter privilégios mínimos para realizar suas funções de trabalho. Uma estratégia simples para gerenciar funções de PFCG seria nomear funções únicas de PFCG de acordo com o perfil do SDK e a função lógica que elas autorizam. Por exemplo, o perfil Z_AWS_PFL_ZFINANCE_CFO concede acesso ao perfil ZFINANCE e ao perfil lógico do IAM do CFO. Esses perfis únicos podem então ser atribuídas a perfis compostos que definem as funções do trabalho. Cada empresa tem sua própria estratégia para gerenciamento de perfis, e nós encorajamos você a definir uma estratégia de PFCG que funcione para a sua empresa.

Operações seguras

Criptografia de dados em repouso

AWS As chaves de acesso secretas são usadas para autenticar o SDK. Eles são criptografados usando a funcionalidade SSF ou Credential Store da SAP.

Criptografia de dados em trânsito

Todas as chamadas para Serviços da AWS são criptografadas com HTTPS. O SAP ICM gerencia a conexão HTTPS. AWS os certificados devem ser confiáveis no STRUST.

Uso da API

Quando um usuário fABAP assume um perfil usando `sts:assumeRole`, o nome da sessão é intitulado `USERID-SID-MANDT`, onde:

- `USERID` é o usuário ABAP da variável `SY-UNAME`.
- `SID` é o ID do sistema ABAP da variável `SY-SYSID`.
- `MANDT` é o cliente ABAP da variável `SY-MANDT`.

O nome da sessão aparece CloudTrail como nome de usuário. Isso garante que as chamadas de API de um usuário ABAP possam ser rastreadas até o sistema, o cliente e o usuário que iniciou a chamada. Para ter mais informações, consulte [O que é o AWS CloudTrail?](#)

Usar certificados com o IAM Roles Anywhere

O sistema SAP pode ser autenticado AWS usando a autenticação baseada em certificados com AWS Identity and Access Management o Roles Anywhere. Você deve configurar o certificado no STRUST e configurar o perfil do SDK em `/AWS1/IMG`.

Pré-requisitos

Os pré-requisitos a seguir devem ser atendidos antes de começar a configuração para certificação.

- O certificado X.509 emitido pela autoridade de certificação (CA) deve atender aos requisitos a seguir.
 - O certificado de assinatura deve ser um certificado v3.
 - A cadeia não deve exceder cinco certificados.
 - O certificado deve permitir algoritmos RSA ou ECDSA.
- Registre a CA no IAM Roles Anywhere como uma âncora de confiança e crie um perfil para especificar os perfis/as políticas para o IAM Roles Anywhere. Para obter mais informações, consulte [Criação de uma âncora de confiança e um perfil no AWS Identity and Access Management Roles Anywhere](#).

- Os perfis do IAM para usuários do SAP devem ser criados pelo administrador do IAM. As funções devem ter permissões para chamar o necessário Serviços da AWS. Para obter mais informações, consulte [Práticas recomendadas de segurança no IAM](#).
- Crie autorização para executar a transação /AWS1/IMG. Consulte mais informações em [Autorizações para configuração](#).

Procedimento

Siga estas instruções para configurar a autenticação baseada em certificado.

Etapas

- [Etapa 1: definir uma aplicação de SSF usando o Secure Store and Forward \(SSF\) de SAP](#)
- [Etapa 2: definir os parâmetros de SSF](#)
- [Etapa 3: criar o PSE e a solicitação de certificado](#)
- [Etapa 4: importar a resposta do certificado para o PSE relevante](#)
- [Etapa 5: configurar o perfil do SDK para usar o IAM Roles Anywhere](#)

Etapa 1: definir uma aplicação de SSF usando o Secure Store and Forward (SSF) de SAP

1. Execute o código da transação SE16 para definir uma aplicação de SSF.
2. Insira o nome da tabela SSFAPPLIC e selecione Novas entradas.
3. Insira um nome para a aplicação de SSF no campo APPLIC, uma descrição no campo DESCRIPT e selecione a opção Selected (X) para os campos restantes.

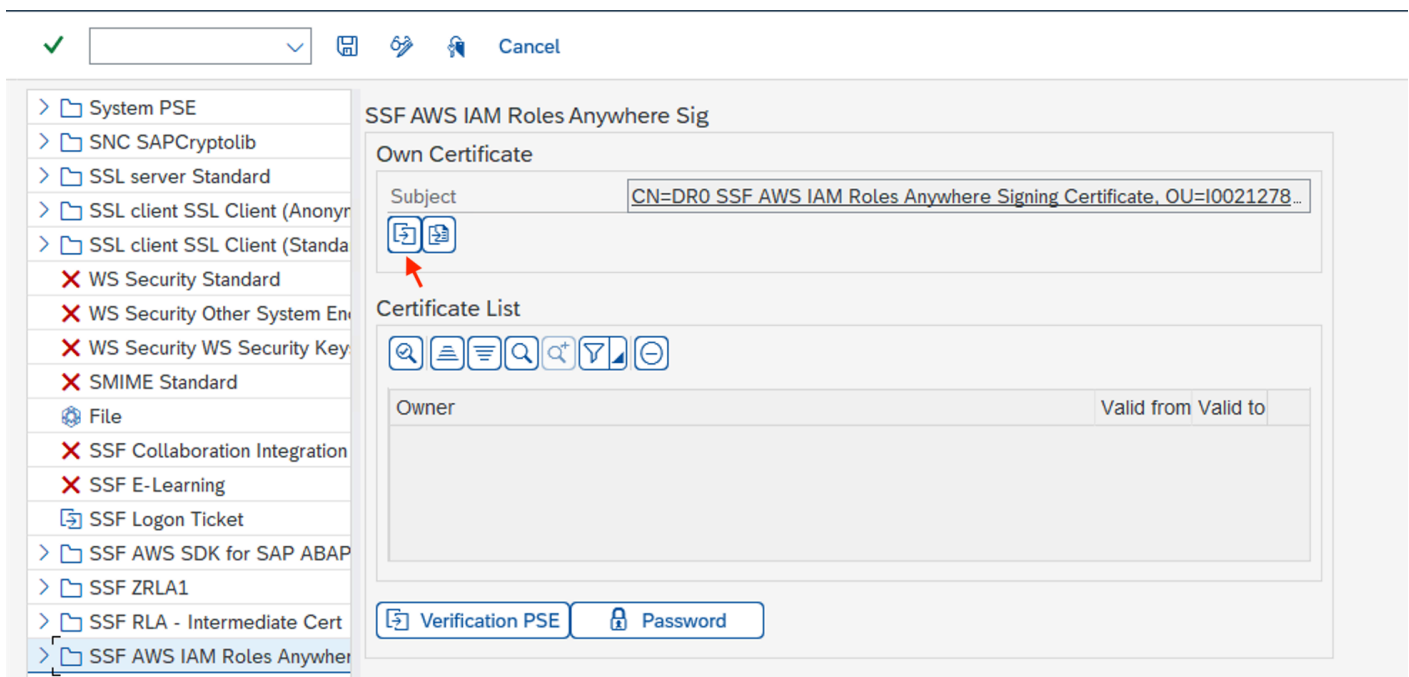
Etapa 2: definir os parâmetros de SSF

1. Execute o Guia /n/AWS1/IMG de SDK da AWS para SAP ABAP Implementação (IMG) para iniciar.
2. Selecione Configurações do SDK da AWS para SAP ABAP > Pré-requisitos técnicos > Configurações adicionais para sistemas on-premises.
3. Execute a atividade do IMG Definir parâmetros de SSF.
4. Selecione Novas entradas e escolha a aplicação de SSF criada na etapa anterior. Selecione Save (Salvar).

5. Modifique o algoritmo de hash para SHA256 e o algoritmo de criptografia para AES256-CBC. Mantenha as outras configurações como padrão e selecione Salvar.

Etapa 3: criar o PSE e a solicitação de certificado

1. Execute a transação /n/AWS1/IMG e selecione Configurações do SDK da AWS para SAP ABAP > Pré-requisitos técnicos > Configurações adicionais para sistemas on-premises.
2. Execute a atividade Create PSE for SSF Application do IMG.
3. Selecione Editar para a transação STRUST.
4. Selecione com o botão direito a aplicação de SSF criado na [the section called "Etapa 1"](#) e escolha Criar. Mantenha todas as outras configurações padrão e selecione Continuar.
5. Selecione Criar solicitação de certificado. Veja a imagem a seguir. Mantenha as opções padrão e selecione Continuar. Copie ou exporte a solicitação de certificado gerada e forneça-a à CA. A CA verifica a solicitação e responde com um certificado de chave pública assinado.



O processo de assinatura varia de acordo com a CA e a tecnologia usada por ela. Consulte [Emissão de certificados de entidade final privada](#) com a Autoridade de Certificação AWS Privada para obter um exemplo.

Etapa 4: importar a resposta do certificado para o PSE relevante

1. Execute a transação /n/AWS1/IMG e selecione Configurações do SDK da AWS para SAP ABAP > Pré-requisitos técnicos > Configurações adicionais para sistemas on-premises.
2. Execute a atividade Create PSE for SSF Application do IMG.
3. Selecione Editar para a transação STRUST.
4. Escolha a aplicação de SSF e selecione Importar resposta do certificado, localizada na seção de PSE abaixo do assunto. Copie e cole a resposta do certificado na caixa de texto ou importe o arquivo do sistema de arquivos. Selecione Continuar > Salvar.
5. Os detalhes do certificado podem ser visualizados selecionando o assunto duas vezes. As informações são exibidas na seção de certificados.

Etapa 5: configurar o perfil do SDK para usar o IAM Roles Anywhere

1. Execute a transação /n/AWS1/IMG e selecione Configurações do SDK da AWS para SAP ABAP > Pré-requisitos técnicos > Configurações da aplicação.
2. Crie um perfil do SDK e dê um nome a ele.
3. Escolha o IAM Roles Anywhere como método de autenticação.
 - No painel esquerdo, escolha Autenticação e configurações.
 - Crie uma entrada e insira as informações do sistema de SAP e da Região da AWS.
 - Selecione IAM Roles Anywhere para o método de autenticação e selecione Salvar.
 - Selecione Inserir detalhes e, na janela pop-up, escolha a aplicação de SSF criada na [the section called “Etapa 1”](#). Insira o ARN da âncora de confiança e o ARN do perfil que foram criados em [the section called “Pré-requisitos”](#). Veja a imagem a seguir. Selecione Continuar.

☰ SAP ✕

Select Signing Certificate issued by your certificate authority (CA) from SSF

Certificate (SSF Application)

Enter your IAM Roles Anywhere details

Trust Anchor ARN

Profile ARN

✓ ✕

4. No painel esquerdo, selecione Mapeamento de perfis do IAM. Insira um nome e forneça o ARN do perfil do IAM fornecido pelo administrador do IAM.

Consulte mais informações em [Configuração da aplicação](#).

Usando o SAP Credential Store

O SAP Credential Store é usado na SAP Business Technology Platform para armazenar com segurança as credenciais para autenticação de chave de acesso secreta. AWS Você deve ter uma assinatura para usar o serviço.

As instruções a seguir pressupõem que você já tenha configurado um perfil do SDK. Para obter mais informações, consulte [Configurando. SDK da AWS para SAP ABAP](#)

Antes de iniciar a configuração, verifique se você atende aos pré-requisitos. Para obter mais informações, consulte [SAP Credential Store](#).

Tópicos

- [Etapas de configuração](#)
- [Usando o SAP Credential Store com o SDK](#)

Etapas de configuração

Etapas

- [Etapa 1: definir as configurações para autenticação](#)
- [Etapa 2: criar uma chave de serviço](#)
- [Etapa 3: converter a chave de serviço em .p12 formato](#)
- [Etapa 4: Conecte-se ao SAP BTP, ambiente ABAP](#)

Etapa 1: definir as configurações para autenticação

Use as etapas a seguir para definir as configurações do Credential Store para autenticação.

1. Navegue até a guia Configurações da instância do SAP Credential Store.
2. Selecione Editar configurações:
 - Escolha TLS mútuo como o tipo de autenticação padrão.
 - Selecione Desativado para o status de criptografia de carga útil. A carga é criptografada em trânsito com HTTPS. No entanto, a carga útil atualmente não pode ser criptografada duas vezes.
3. Selecione Save (Salvar).

Etapa 2: criar uma chave de serviço

Use as etapas a seguir para criar uma chave de serviço para o Credential Store.

1. No painel esquerdo do aplicativo SAP Credential Store, navegue até Service Keys.
2. Selecione Criar chave de serviço.
3. Insira um nome para a chave de serviço e selecione Criar.

A chave de serviço é criada com base no tipo de autenticação escolhido. Faça o download da chave de serviço e mantenha-a segura para uso posterior.

Etapa 3: converter a chave de serviço em **.p12** formato

É necessário um certificado de cliente no .p12 formato para criar um usuário de saída para o sistema de comunicação. Use as etapas a seguir para gerar um .p12 certificado a partir dos detalhes do certificado fornecidos na chave do Serviço de Armazenamento de Credenciais.

1. Baixe o certificado SAP Cloud Root CA (exigido pela SAP) do [SAP Trust Center Services](#).

2. Abra o certificado SAP Cloud Root CA em qualquer formato de arquivo de texto. No final do arquivo, pressione Enter e copie e cole o certificado do campo certificado da chave de serviço. Substitua os caracteres \n da nova linha pela nova linha real (Enter) e salve o certificado inteiro em formato de .cer arquivo.
3. Copie a chave do campo chave da chave de serviço. Essa chave privada deve ser tratada como dados confidenciais. Cole-o em um arquivo de texto e substitua os caracteres \n da nova linha pela nova linha real (Enter). Salve a chave privada em um arquivo de texto.
4. Com o certificado e a chave privada gerados nas etapas anteriores, execute o comando a seguir para gerar um .p12 certificado.

```
openssl pkcs12 -export -out <.p12_<filename>> -inkey <private_key.key> -in  
<certificate.cer>
```

O comando exigia a verificação da senha de exportação. Guarde a senha para uso posterior.

Exclua o arquivo de .key texto salvo em sua chave privada.

Etapa 4: Conecte-se ao SAP BTP, ambiente ABAP

Configure o ambiente SAP BTP, ABAP para se conectar ao SAP Credential Store.

Tópicos

- [Sistema de comunicação](#)
- [Arranjo de comunicação](#)

Sistema de comunicação

Use as etapas a seguir para criar um sistema de comunicação que permita a comunicação do ambiente SAP BTP e ABAP com o SAP Credential Store.

1. Abra a barra de lançamento Fiori do sistema de ambiente ABAP.
2. Selecione o quadro Sistemas de comunicação para abrir o aplicativo.
3. Selecione Novo.
4. Insira um nome e uma ID para o sistema de comunicação e selecione Criar. Por exemplo, você pode nomear o sistema ZSAP_CREDSTORE.
5. Insira outras informações necessárias:

- Nome do host: copie o nome do host do URL da chave de serviço. Por exemplo, se o URL for `https://credstore.mesh.cf.us10.hana.ondemand.com/api/v1/credentials`, o nome do host será `credstore.mesh.cf.us10.hana.ondemand.com`.
 - Usuários para comunicação externa: selecione + para adicionar um novo usuário.
 - a. Selecione Certificado de cliente SSL como mecanismo de autenticação.
 - b. Selecione Carregar novo certificado:
 - Procure o .p12 certificado gerado na etapa anterior.
 - Insira uma descrição.
 - Insira a senha de exportação usada para gerar o .p12 certificado.
 - Selecione Carregar.
 - c. Selecione Criar para criar um usuário externo.
6. Selecione Save (Salvar).
 7. Exclua a chave de serviço baixada na etapa anterior.

Arranjo de comunicação

Use as etapas a seguir para criar um acordo de comunicação para fornecer um cenário de comunicação para comunicação externa.

1. Abra a barra de lançamento Fiori do sistema de ambiente ABAP.
2. Selecione o quadro Arranjos de comunicação para abrir o aplicativo.
3. Selecione Novo.
4. Selecione o cenário `/AWS1/CRED_COMM_SCENARIO0` de comunicação e insira um nome para o arranjo de comunicação. Por exemplo, `Z_AWS_SDK_TO_SAP_CREDSTORE`.
5. Escolha Criar.
6. No campo Sistema de Comunicação, procure o Sistema de Comunicação criado na etapa anterior. Outras informações são preenchidas automaticamente após a seleção do sistema.
7. Selecione Save (Salvar).
8. Selecione Verificar conexão para testar sua conexão.

Quando essa configuração estiver concluída, o ambiente ABAP poderá usar o arranjo de comunicação para usar o serviço SAP Credential Store via serviço de saída (HTTP).

Usando o SAP Credential Store com o SDK

Etapas

- [Etapa 1: criar um namespace e credencial \(s\)](#)
- [Etapa 2: Configurar o aplicativo Custom Business Configuration](#)

Etapa 1: criar um namespace e credencial (s)

Crie um namespace e uma credencial no SAP Credential Store com a ajuda da SAP — [Crie, edite e exclua uma credencial](#).

Insira os detalhes a seguir para criar uma credencial do tipo Key.

- Namespace — insira um nome para o namespace e agrupe as credenciais relacionadas.
- Nome — Insira um nome para a chave. Recomendamos `saws-0123456789012-username`, onde:
 - `0123456789012` é o Conta da AWS ID ao qual a credencial concede acesso
 - `username` é o nome de usuário do IAM ao qual a credencial pertence
- Valor — Insira uma chave de acesso secreta codificada em base 64. Use o comando a seguir para codificar em base 64 sua chave de acesso secreta.

```
xargs echo -n | base64 # just press enter, do not enter arguments on the command line
MySecretAccessKey
Ctrl-D
```

O comando lê a chave de acesso secreta da entrada padrão e a passa para base64 sem uma nova linha à direita. Ele envia a chave de acesso secreta codificada em base 64 para a tela. Limpe ou feche seu terminal depois de copiar o valor no SAP Credential Store.

- Nome de usuário — Insira o ID da sua chave de acesso.
- Escolha Criar.

Um novo namespace com uma credencial é criado e as credenciais podem ser adicionadas, excluídas ou modificadas nesse namespace.

Siga o princípio do privilégio mínimo para gerenciar o acesso às credenciais armazenadas no namespace.

Etapa 2: Configurar o aplicativo Custom Business Configuration

Use as etapas a seguir para configurar o aplicativo Custom Business Configuration para definir a credencial a ser usada para autenticação pelo SDK.

1. Abra a barra de lançamento Fiori do sistema de ambiente ABAP.
2. Navegue pelo quadro Configuração Empresarial Personalizada para abrir o aplicativo.
3. Abra a configuração comercial do SDK Profile.
4. Selecione o perfil do SDK para o qual as configurações de autenticação devem ser definidas para o SAP Credential Store.
5. Na guia Autenticação e configurações do perfil selecionado, selecione Editar e insira os seguintes detalhes:
 - Método de autenticação — Selecione as credenciais do SAP Credential Store.
 - Namespace — Insira o namespace criado no SAP Credential Store. Para ter mais informações, consulte [the section called “Etapa 1: criar um namespace e credencial \(s\)”](#).
 - Nome da chave — Insira o nome da chave de serviço criada. Para ter mais informações, consulte [the section called “Etapa 2: criar uma chave de serviço”](#).
 - Arranjo de comunicação — insira o nome do arranjo de comunicação criado. Para ter mais informações, consulte [the section called “Arranjo de comunicação”](#).
6. Selecione Aplicar para acessar a tela Perfil do AWS SDK.
7. Selecione Selecionar transporte para selecionar o transporte usando a ajuda de valores.
8. Selecione Save (Salvar).

Solução de problemas do SDK da AWS para SAP ABAP

Esta seção fornece etapas de solução de problemas para possíveis cenários de erro.

Tópicos

- [Falha de importação](#)
- [Restrição de localização não especificada](#)
- [Erros SSL](#)
- [Configuração do perfil](#)
- [Autorização do IAM](#)
- [Autorização para realizar as ações necessárias](#)
- [Cenário ativo](#)
- [Caracteres especiais no código](#)
- [Conectividade](#)

Falha de importação

Problema: a classe 'CL_SYSTEM_UUID' não contém uma interface 'IF_SYSTEM_UUID_RFC4122_STATIC'

Causa: a nota SAP 0002619546 está ausente em seu sistema.

Resolução: verifique se o [SAP Note 0002619546](#) foi aplicado ao seu sistema.

Restrição de localização não especificada

Problema: a restrição de localização não especificada é incompatível com o endpoint de region específico para o qual a solicitação foi enviada

Causa: seu bucket do Amazon S3 não tem a região da AWS no parâmetro `io_createbucketconfiguration`.

Resolução: ao criar um bucket em qualquer região, exceto `us-east-1`, especifique a região do bucket do Amazon S3 usando o parâmetro `io_createbucketconfiguration` em `createbucket()`. Não é necessário especificar uma restrição para `us-east-1`.

Veja a seguir um exemplo de um parâmetro `io_createbucketconfiguration` configurado corretamente:

```
createbucket(  
  iv_bucket = 'test-bucket'  
  io_createbucketconfiguration = NEW /aws1/cl_s3_createbucketconf( 'us-west-1' )  
).
```

Erros SSL

Problema: falha na incompatibilidade do nome de host do certificado do servidor SSL ou no handshake SSL com docs.aws.amazon.com:443: SSSLERR_NO_SSL_RESPONSE

Causa: o parâmetro `icm/HTTPS/client_sni_enabled` não está definido como TRUE no perfil DEFAULT.

Resolução: use as etapas a seguir para solucionar os problemas em questão ou qualquer outro problema relacionado ao SSL.

1. Abra a SAPGUI e vá até a barra de comando.
2. Execute a transação RZ10.
3. Vá para Perfil e escolha o perfil DEFAULT. A versão é preenchida automaticamente.
4. Na seção Editar perfil, selecione Manutenção estendida e, em seguida, selecione Alterar.
5. Pesquise o parâmetro `icm/HTTPS/client_sni_enabled`.
 - Se o parâmetro existir, edite o valor do parâmetro e defina-o como TRUE.
 - Se o parâmetro não existir, crie um parâmetro usando as etapas a seguir.

1. Selecione Parâmetro.

Note

Selecione o parâmetro para criação e não edição (ícone de lápis).

2. Insira `icm/HTTPS/client_sni_enabled` no campo Nome do parâmetro.
 3. Insira TRUE no campo Valor do parâmetro.
 4. Selecione Save.
6. Salve essas alterações no perfil DEFAULT e saia.

Configuração do perfil

Problema: não foi possível encontrar a configuração no perfil <profile_name> com o cenário DEFAULT para <sid>:<client>

Causas: <profile_name> está incorreto ou não foi configurado.

Resolução: use as etapas a seguir para configurar o perfil.

1. Abra a SAPGUI e execute a transação /n/AWS1/IMG.
2. Vá para Configuração do aplicativo > Perfil do SDK.
 - Se seu perfil estiver configurado, verifique se o nome do perfil está correto.
 - Se seu perfil não estiver configurado, siga as etapas para configurar um perfil.
3. Selecione Novas entradas.
 - a. Insira um Nome e uma Descrição para o perfil.
 - b. Selecione Save.
4. Escolha a entrada criada na etapa anterior e selecione Autenticação e configurações.
5. Selecione Novas entradas, insira os detalhes a seguir e selecione Salvar.
 - SID
 - Cliente
 - ID do cenário
 - AWSRegião da
 - Método de autenticação
 - Selecione a Função da instância via metadados para sistemas SAP em execução na AWS.
 - Selecione Credenciais do armazenamento SSF para sistemas SAP executados on-premises ou em outra nuvem.
6. Selecione Mapeamento de perfis do IAM > Novas entradas, insira os detalhes a seguir e selecione Salvar.
 - Número de sequência
 - Perfil lógico do IAM
 - ARN do perfil do IAM

Autorização do IAM

Problema: não foi possível assumir a função <iam_role_arn> ou Usuário: <user_arn> não está autorizado a executar: sts:AssumeRole no recurso: <iam_role_arn>

Causas: os seguintes podem ser os possíveis motivos desse erro.

- O ARN incorreto do perfil do IAM foi especificado
- O usuário do IAM não tem permissão para acessar o perfil do IAM
- Relação de falta de confiança entre o perfil do IAM assumido e o perfil do IAM assumido ou usuário do IAM

Resolução: use as etapas a seguir para garantir que o ARN do perfil do IAM esteja correto.

1. Abra a SAPGUI e execute a transação /n/AWS1/IMG.
2. Vá para Configuração do aplicativo > Perfil do SDK e escolha o perfil que foi configurado com seu perfil do IAM.
3. Selecione Mapeamento de perfis do IAM e verifique ou corrija o ARN do perfil do IAM.
 - Se o ARN do seu perfil do IAM estiver correto, verifique se seu perfil do IAM foi configurado corretamente. Para obter mais informações, consulte [Solução de problemas dos perfis do IAM](#).

Autorização para realizar as ações necessárias

Problema: o usuário <user_arn> não está autorizado a executar: <action> no recurso: <resource_arn>

Causa: o usuário não tem permissões para realizar uma ação.

Resolução: user_arn deve ser configurado com as permissões necessárias do resource_arn para realizar uma determinada action. Para obter mais informações, consulte [Permissões necessárias para acessar recursos do IAM](#).

Cenário ativo

Problema: nenhum cenário ativo configurado

Causa: a configuração do cenário ativo foi perdida.

Resolução: consulte [Configurações de runtime](#) para configurar um cenário ativo.

Caracteres especiais no código

Aviso: o caractere 0x00A0 não pode fazer parte de uma palavra ABAP

Note

Esse aviso pode ser precedido por várias mensagens de erro.

Causa: copiar e colar código de fontes diferentes pode inserir caracteres especiais em seu código.

Resolução: quando você cola qualquer código no editor de código-fonte ABAP, você vê o seguinte pop-up.

Caracteres de espaço ininterruptos foram detectados. Converter em espaços?

Escolha Sim para responder a essa pergunta. Além disso, recomendamos selecionar o código para copiá-lo, em vez de usar o botão copiar nas caixas de código.

Conectividade

Problema: SCLNT_HTTP (411): Falha na conexão direta com tla.region.amazonaws.com:443: NIECONN_REFUSED (-10)

Causa: o sistema SAP não tem conectividade com a Internet e não pode estabelecer uma conexão TCP/IP com a porta 443 de tla.region.amazonaws.com.

Resolução: o sistema SAP deve ser capaz de estabelecer conexão com endpoints da AWS na porta HTTPS 443, diretamente ou por meio de um servidor proxy. Você pode estabelecer/verificar a conectividade com a Internet com uma das seguintes opções:

- Conexão de saída direta à Internet por meio de um NAT ou gateway da Internet
- Conexão por meio de um servidor proxy

Para obter mais informações, consulte [Conexão por meio de um servidor de proxy](#).

Tópicos adicionais

Esta seção abrange os seguintes tópicos.

Tópicos

- [SDK da AWS para SAP ABAP lançamentos](#)
- [Licenciamento SAP](#)

SDK da AWS para SAP ABAP lançamentos

AWS O SDK para SAP ABAP é entregue em transportes, e o AWS SDK para SAP ABAP - edição BTP é entregue como complementos. O mecanismo para importar transportes e complementos é diferente, mas a funcionalidade técnica é a mesma. Para obter mais informações, consulte [Configurar](#).

Tópicos

- [Estratégia de lançamento](#)
- [Práticas recomendadas](#)
- [Patches de SDK para SAP ABAP](#)
- [Instalação de um módulo adicional](#)
- [Desinstalando o SDK para SAP ABAP](#)

Estratégia de lançamento

A versão 1 do SDK da AWS para SAP ABAP é atualizada com frequência. Novos patches são lançados semanalmente ou diariamente com base nos lançamentos e atualizações do Serviços da AWS. Os patches para Serviços da AWS podem incluir correções de bugs e outras alterações que atualizam o nível de patch do SDK. Para obter mais informações, consulte a política de manutenção de [AWS SDKs e ferramentas](#).

Práticas recomendadas

Recomendamos reter o mesmo nível de patch do SDK para SAP ABAP para todos os sistemas SAP (desenvolvimento, controle de qualidade e produção).

Ao corrigir o SDK, importe a versão mais recente em sua sandbox. Em seguida, você pode importá-lo para os sistemas de desenvolvimento, controle de qualidade e produção, seguindo seus procedimentos normais de controle de alterações.

Patches de SDK para SAP ABAP

Cada versão do SDK para SAP ABAP é entregue como um conjunto de transportes cumulativos, incluindo todas as correções de erros, atributos e atualizações. Não há diferença entre um patch e um transporte de instalação. Você deve importar os transportes mais recentes para corrigir o SDK para SAP ABAP.

Devido às dependências dos módulos `core` Runtime e API, você deve corrigir o módulo `core` e todos os outros módulos que você instalou, mesmo que não esteja mais usando esses módulos. Por exemplo, se você importou os transportes `core`, `ec2` e `lmd` ao instalar o SDK, deverá importar os transportes mais recentes para `core`, `ec2` e `lmd` durante a aplicação de patches.

Instalação de um módulo adicional

Importe o transporte para o novo módulo no mesmo nível de patch que o existente `core` e os módulos para instalar um módulo de API adicional em seu sistema SAP. Siga as diretrizes em [the section called “Patches de SDK para SAP ABAP”](#) se quiser importar uma versão mais recente do módulo. Isso garante que os níveis de patch sejam compatíveis em todos os módulos do SDK.

Desinstalando o SDK para SAP ABAP

Para desinstalar o SDK para SAP ABAP, você deve baixar um kit de transporte de exclusão em <https://.aws.amazon.com/ V1/release/uninstall-abapsdk-LATEST.zip>. `awsSdkSapabap`

```
curl "https://sdk-for-sapabap.aws.amazon.com/awsSdkSapabapV1/release/uninstall-abapsdk-LATEST.zip" -o "uninstall-abapsdk-LATEST.zip"
```

Você pode baixar um arquivo de assinatura em <https://sdk-for-sapabap.aws.amazon.com/v1/release/uninstall-abapsdk-latest.sig>. `awsSdkSapabap` Para validar o arquivo, consulte [Verificar SDK para SAP ABAP](#).

Para cada módulo SDK instalado em seu sistema SAP, o transporte de exclusão correspondente deve ser importado do arquivo ZIP anterior. Você pode remover um único módulo sem desinstalar o SDK inteiro. Você pode fazer isso importando somente o transporte de exclusão do módulo que

você deseja remover. Se você estiver desinstalando o SDK inteiro com todos os seus módulos, o transporte de exclusão principal deverá ser importado por último.

Recomendamos que você teste a desinstalação em um sandbox antes de tentar nos sistemas de desenvolvimento, controle de qualidade ou produção.

Considerações

Antes de desinstalar o SDK, consulte as considerações a seguir.

- As configurações do SDK de serão perdidas. O IMG deve ser reconfigurado na instalação.
- Se você tiver programas Z que dependem do SDK, eles gerarão erros de sintaxe após a remoção do SDK.
- As funções PFCG ou Business que contêm referências de autorização do SDK terão autorizações inválidas após a remoção do SDK. Remova as referências de autorização do SDK dos perfis do PFCG antes de desinstalar o SDK.

Note

AWS O SDK para SAP ABAP - edição BTP não pode ser desinstalado durante a versão prévia para desenvolvedores.

Licenciamento SAP

O uso do software da SAP está sujeito aos termos da SAP. Você é responsável por cumprir os termos de licenciamento da SAP, incluindo distribuição de software e condições de licenciamento indireto. Qualquer informação fornecida não é aconselhamento jurídico e não deve ser usada para fins de conformidade com o licenciamento. Se tiver dúvidas sobre o licenciamento ou os direitos ao software da Microsoft, consulte sua equipe jurídica, a Microsoft ou seu revendedor da Microsoft.

Pergunta: o uso do SDK para SAP ABAP afetará minha licença SAP?

Resposta: o SDK da AWS para SAP ABAP permite que você consuma os Serviços da AWS com seu próprio código ABAP. É usado em cenários de integração entre um sistema SAP e os Serviços da AWS. Qualquer cenário em que os dados do sistema SAP sejam enviados para um sistema terceirizado (não SAP) ou criados por esse sistema pode ter implicações no licenciamento indireto. A SAP tem várias abordagens para definir o acesso indireto, como cálculos baseados no usuário

e cálculos baseados em resultados. A metodologia para definir o acesso indireto depende do seu contrato com a SAP. Você deve estar ciente das orientações fornecidas em seu contrato com a SAP e pode discutir mais sobre isso com a SAP ou seu revendedor.

Em 2018, a SAP lançou dois documentos, Guia de acesso indireto para clientes da base instalada da SAP e Preços do SAP ERP para a era digital, abordando o acesso indireto/digital. Esses documentos podem ser encontrados nos sites da SAP e são exemplos de abordagens de licenciamento indireto. No entanto, esses documentos não refletem seu contrato específico com a SAP.

Histórico de documentos do SDK da AWS para SAP ABAP Developer Guide

A tabela a seguir descreve as versões de documentação do SDK da AWS para SAP ABAP.

Alteração	Descrição	Data
Novo conteúdo	Prévia para desenvolvedores do SDK para SAP ABAP - edição BTP.	31 de maio de 2024
Novo conteúdo	Adição de Using certificates with IAM Roles Anywhere .	1.º de dezembro de 2023
Novo conteúdo	Adição de Building products with SDK .	1.º de dezembro de 2023
Novo conteúdo	Adição de Retry behavior .	1.º de dezembro de 2023
Novo conteúdo	Adição de Licenciamento SAP .	22 de setembro de 2023
Versão pública	Relançamento inicial do SDK da AWS para SAP ABAP Developer Guide.	30 de junho de 2023
Novo conteúdo	Atributos adicionados do SDK da AWS para SAP ABAP .	30 de maio de 2023
Novo conteúdo	Solução de problemas adicionada do SDK da AWS para SAP ABAP .	17 de fevereiro de 2023
Pré-visualização para desenvolvedores	Prévia do SDK da AWS para SAP ABAP Developer Guide para desenvolvedores.	17 de novembro de 2022

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.