

Panduan Developerr

AWS SDKuntuk Ruby



AWS SDK untuk Ruby: Panduan Developer

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

Merek dagang dan tampilan dagang Amazon tidak boleh digunakan sehubungan dengan produk atau layanan apa pun yang bukan milik Amazon, dengan cara apa pun yang dapat menyebabkan kebingungan antara para pelanggan, atau dengan cara apa pun yang menghina atau mendiskreditkan Amazon. Semua merek dagang lain yang tidak dimiliki oleh Amazon merupakan kekayaan masing-masing pemiliknya, yang mungkin atau mungkin tidak berafiliasi, terkait dengan, atau disponsori oleh Amazon.

Table of Contents

Apa AWS SDK for Ruby?	1
Dokumentasi dan sumber daya tambahan	1
Menerapkan ke Cloud AWS	2
Pemeliharaan dan dukungan untuk versi utama SDK	2
Memulai	3
Otentikasi SDK dengan AWS	3
Memulai sesi portal AWS akses	4
Informasi autentikasi lebih lanjut	5
Instal SDK	5
Prasyarat	6
Instalasi SDK	6
Halo tutorial	7
Tulis kode	7
Menjalankan program	8
Catatan untuk pengguna Windows	9
Langkah selanjutnya	9
Gunakan AWS Cloud9 dengan SDK	9
Langkah 1: Siapkan Akun AWS untuk Anda gunakan AWS Cloud9	10
Langkah 2: Siapkan lingkungan AWS Cloud9 pengembangan Anda	10
Langkah 3: Siapkan AWS SDK Ruby	11
Langkah 4: Unduh kode contoh	12
Langkah 5: Jalankan kode contoh	13
Konfigurasi SDK	15
Rantai penyedia kredensi	15
Membuat token AWS STS akses	17
Mengatur Wilayah	17
Mengatur Wilayah menggunakan config file bersama	18
Mengatur Wilayah menggunakan variabel lingkungan	18
Mengatur Wilayah dengan <code>Aws.config</code>	18
Mengatur Wilayah di klien atau objek sumber daya	19
Menetapkan titik akhir yang tidak standar	19
Gunakan SDK	20
Gunakan utilitas REPL	20
Prasyarat	20

Pengaturan bundler	21
Menjalankan REPL	21
Gunakan SDK dengan Ruby on Rails	22
Tip debugging: Dapatkan informasi pelacakan kawat dari klien	22
Respons dan kesalahan klien rintisan	23
Tanggapan klien yang mematahkan	23
Kesalahan klien yang mematahkan	25
Paginasi	25
Respons paged dapat dihitung	25
Menangani tanggapan paged secara manual	26
Kelas data paged	26
Pelayan	26
Memanggil pelayan	27
Kegagalan tunggu kegagalan	27
Mengkonfigurasi pelayan	28
Memperluas pelayan	28
Tentukan perilaku coba lagi klien	29
Bermigrasi dari versi 1 atau 2 ke AWS SDK versi 3 untuk Ruby	30
ide-by-side Penggunaan S	30
Perbedaan umum	30
Perbedaan klien	31
Perbedaan sumber daya	32
Contoh kode	34
Aurora	35
Auto Scaling	36
CloudTrail	38
Tindakan	38
CloudWatch	42
Tindakan	38
Penyedia Identitas Amazon Cognito	54
Amazon Comprehend	56
Skenario	56
Amazon DocumentDB	57
Contoh nirserver	58
DynamoDB	59
Hal-hal mendasar	61

Tindakan	38
Skenario	56
Contoh nirserver	58
Amazon EC2	87
Tindakan	38
Elastic Beanstalk	123
Tindakan	38
EventBridge	129
Skenario	56
AWS Glue	150
Hal-hal mendasar	61
Tindakan	38
IAM	180
Tindakan	38
Skenario	56
Kinesis	236
Contoh nirserver	58
AWS KMS	239
Tindakan	38
Lambda	243
Tindakan	38
Skenario	56
Contoh nirserver	58
Amazon MSK	272
Contoh nirserver	58
Amazon Polly	273
Tindakan	38
Skenario	56
Amazon RDS	278
Tindakan	38
Contoh nirserver	58
Amazon S3	285
Hal-hal mendasar	61
Tindakan	38
Skenario	56
Contoh nirserver	58

Amazon SES	317
Tindakan	38
Amazon SES API v2	323
Tindakan	38
Amazon SNS	324
Tindakan	38
Contoh nirserver	58
Amazon SQS	334
Tindakan	38
Contoh nirserver	58
AWS STS	347
Tindakan	38
Amazon Textract	349
Skenario	56
Amazon Translate	350
Skenario	56
Amazon WorkDocs	351
Tindakan	38
Keamanan	355
Perlindungan Data	355
Identity and Access Management	356
Validasi Kepatuhan	357
Ketangguhan	358
Keamanan Infrastruktur	359
Menegakkan versi minimum TLS	359
Memeriksa SSL versi Terbuka	359
Meningkatkan dukungan TLS	360
Migrasi Klien Enkripsi S3	360
Ikhtisar Migrasi	360
Perbarui Klien yang Ada untuk Membaca Format Baru	360
Migrasi Klien Enkripsi dan Dekripsi ke V2	362
Riwayat Dokumen	366
.....	ccclxviii

Apa AWS SDK for Ruby?

Selamat datang di AWS SDK for Ruby Developer Guide. AWS SDK for Ruby, Amazon Elastic Compute Cloud (Amazon S3), Amazon Elastic Compute Cloud (Amazon EC2) AWS layanan, Amazon Elastic Compute Cloud (Amazon EC2), Amazon Elastic Compute Cloud (Amazon EC2), Amazon Elastic Compute Cloud (Amazon EC2), Amazon DynamoDB.

AWS SDK for Ruby Developer Guide menyediakan informasi tentang cara menginstal, mengatur, dan menggunakan AWS SDK for Ruby untuk membuat aplikasi Ruby yang menggunakan AWS layanan

[Memulai AWS SDK for Ruby](#)

Dokumentasi dan sumber daya tambahan

Untuk AWS sumber daya selengkapnya, lihat berikut:

- [AWS Panduan Referensi SDK dan Alat](#) - Berisi pengaturan, fitur, dan konsep dasar lainnya yang umum di antara SDK AWS
- [AWS SDK for Ruby Referensi API - Versi 3](#)
- [AWS Contoh Kode Repositori](#) pada GitHub
- [RubyGems.org](#) - Versi terbaru SDK dimodulasi menjadi permata khusus layanan yang tersedia di sini
 - [Layanan yang Didukung](#) - Mencantumkan semua permata yang didukung AWS SDK for Ruby
- AWS SDK untuk sumber Ruby pada: GitHub
 - [Sumber](#) dan [README](#)
 - [Ubah log di bawah setiap permata](#)
 - [Pindah dari v2 ke v3](#)
 - [Masalah](#)
 - [Catatan peningkatan inti](#)
- [Blog pengembang](#)
- [Saluran Gitter](#)
- [@awsforruby](#) di Twitter

Menerapkan ke Cloud AWS

Anda dapat menggunakan AWS layanan seperti AWS Elastic Beanstalk, AWS OpsWorks, dan AWS CodeDeploy untuk menyebarkan aplikasi Anda ke AWS Cloud. Untuk menerapkan aplikasi Ruby dengan Elastic Beanstalk, lihat [Deploying Elastic Beanstalk Applications di Ruby Using EB CLI dan Git di dalam Developer Guide](#). AWS Elastic Beanstalk Untuk men-deploy aplikasi Ruby on Rails AWS OpsWorks, lihat [Deploying Ruby on Rails Applications ke AWS OpsWorks](#) Untuk ikhtisar layanan AWS penyebaran, lihat [Ikhtisar Opsi Penyebaran aktif](#). AWS

Pemeliharaan dan dukungan untuk versi utama SDK

Untuk informasi tentang pemeliharaan dan dukungan untuk versi utama SDK dan dependensi yang mendasarinya, lihat berikut di [Panduan Referensi SDK dan Alat AWS](#):

- [AWS Kebijakan Pemeliharaan SDK dan Alat](#)
- [AWS Matriks Support Versi SDK dan Alat](#)

Memulai AWS SDK for Ruby

Pelajari cara menginstal, mengatur, dan menggunakan SDK untuk membuat aplikasi Ruby untuk mengakses AWS sumber daya secara terprogram.

Topik

- [Otentikasi SDK dengan AWS](#)
- [Instal AWS SDK for Ruby](#)
- [Halo tutorial untuk AWS SDK for Ruby](#)
- [Gunakan AWS Cloud9 dengan AWS SDK untuk Ruby](#)

Otentikasi SDK dengan AWS

Anda harus menetapkan bagaimana kode Anda mengautentikasi dengan AWS ketika mengembangkan dengan AWS layanan. Anda dapat mengonfigurasi akses terprogram ke AWS sumber daya dengan cara yang berbeda tergantung pada lingkungan dan AWS akses yang tersedia untuk Anda.

Untuk memilih metode autentikasi dan mengkonfigurasinya untuk SDK, lihat [Autentikasi dan akses](#) di Panduan Referensi AWS SDK dan Alat.

Kami menyarankan agar pengguna baru yang berkembang secara lokal dan tidak diberi metode otentikasi oleh majikan mereka harus menyiapkan AWS IAM Identity Center. Metode ini termasuk menginstal AWS CLI untuk kemudahan konfigurasi dan untuk masuk secara teratur ke portal AWS akses. Jika Anda memilih metode ini, lingkungan Anda harus berisi elemen-elemen berikut setelah Anda menyelesaikan prosedur untuk [otentikasi IAM Identity Center](#) di AWSSDK dan Tools Reference Guide:

- Itu AWS CLI, yang Anda gunakan untuk memulai sesi AWS akses portal sebelum Anda menjalankan aplikasi Anda.
- [AWSconfigFile bersama](#) yang memiliki [default] profil dengan serangkaian nilai konfigurasi yang dapat direferensikan dari SDK. Untuk menemukan lokasi file ini, lihat [Lokasi file bersama di Panduan Referensi AWS SDK dan Alat](#).
- `configFile` bersama menetapkan [region](#) pengaturan. Ini menetapkan default Wilayah AWS yang digunakan SDK untuk AWS permintaan. Wilayah ini digunakan untuk permintaan layanan SDK yang tidak ditentukan dengan Wilayah untuk digunakan.

- SDK menggunakan [konfigurasi penyedia token SSO](#) profil untuk memperoleh kredensi sebelum mengirim permintaan ke AWS. `sso_role_name` Nilai, yang merupakan peran IAM yang terhubung ke set izin IAM Identity Center, memungkinkan akses ke yang AWS layanan digunakan dalam aplikasi Anda.

configFile contoh berikut menunjukkan profil default yang diatur dengan konfigurasi penyedia token SSO. `sso_session` Pengaturan profil mengacu pada [sso-sessionbagian](#) bernama. `sso-session` Bagian ini berisi pengaturan untuk memulai sesi portal AWS akses.

```
[default]
sso_session = my-sso
sso_account_id = 111122223333
sso_role_name = SampleRole
region = us-east-1
output = json

[sso-session my-sso]
sso_region = us-east-1
sso_start_url = https://provided-domain.awsapps.com/start
sso_registration_scopes = sso:account:access
```

AWSSDK for Ruby tidak memerlukan paket tambahan (seperti `SSO` dan `SSO0IDC`) untuk ditambahkan ke aplikasi Anda untuk menggunakan otentikasi IAM Identity Center.

Memulai sesi portal AWS akses

Sebelum menjalankan aplikasi yang mengakses AWS layanan, Anda memerlukan sesi portal AWS akses aktif agar SDK menggunakan autentikasi IAM Identity Center untuk menyelesaikan kredensi. Bergantung pada panjang sesi yang dikonfigurasi, akses Anda pada akhirnya akan kedaluwarsa dan SDK akan mengalami kesalahan autentikasi. Untuk masuk ke portal AWS akses, jalankan perintah berikut di AWS CLI.

```
aws sso login
```

Jika Anda mengikuti panduan dan memiliki pengaturan profil default, Anda tidak perlu memanggil perintah dengan `--profile` opsi. Jika konfigurasi penyedia token SSO Anda menggunakan profil bernama, perintahnya adalah `aws sso login --profile named-profile`.

Untuk menguji secara opsional jika Anda sudah memiliki sesi aktif, jalankan AWS CLI perintah berikut.

```
aws sts get-caller-identity
```

Jika sesi Anda aktif, respons terhadap perintah ini melaporkan akun IAM Identity Center dan set izin yang dikonfigurasi dalam `config` file bersama.

Note

Jika Anda sudah memiliki sesi portal AWS akses aktif dan menjalankannya `aws sso login`, Anda tidak akan diminta untuk memberikan kredensi.

Proses login mungkin meminta Anda untuk mengizinkan AWS CLI akses ke data Anda. Karena AWS CLI dibangun di atas SDK untuk Python, pesan izin mungkin berisi variasi `botocore` nama.

Informasi autentikasi lebih lanjut

Pengguna manusia, juga dikenal sebagai identitas manusia, adalah orang, administrator, pengembang, operator, dan konsumen aplikasi Anda. Mereka harus memiliki identitas untuk mengakses AWS lingkungan dan aplikasi Anda. Pengguna manusia yang merupakan anggota organisasi Anda - itu berarti Anda, pengembang - dikenal sebagai identitas tenaga kerja.

Gunakan kredensi sementara saat mengakses AWS. Anda dapat menggunakan penyedia identitas bagi pengguna manusia untuk menyediakan akses federasi ke AWS akun dengan mengasumsikan peran, yang memberikan kredensi sementara. Untuk manajemen akses terpusat, kami menyarankan Anda menggunakan AWS IAM Identity Center (IAM Identity Center) untuk mengelola akses ke akun dan izin Anda dalam akun tersebut. Untuk alternatif lebih lanjut, lihat hal berikut:

- Untuk mempelajari lebih lanjut tentang praktik terbaik, lihat [Praktik terbaik keamanan di IAM](#) dalam Panduan Pengguna IAM.
- Untuk membuat AWS kredensi jangka pendek, lihat [Kredensi Keamanan Sementara](#) di Panduan Pengguna IAM.
- Untuk mempelajari tentang penyedia kredensi AWS SDK for Ruby lainnya, lihat penyedia [kredensi standar](#) di Panduan Referensi AWS SDK dan Alat.

Instal AWS SDK for Ruby

Bagian ini mencakup prasyarat dan petunjuk instalasi untuk SDK for Ruby. AWS

Prasyarat

Sebelum Anda menggunakan AWS SDK for Ruby, Anda harus mengotentikasi dengan AWS. Untuk informasi tentang menyiapkan autentikasi, lihat [Otentikasi SDK dengan AWS](#).

Instalasi SDK

Anda dapat menginstal AWS SDK for Ruby seperti yang Anda lakukan setiap Ruby gem. Permata yang tersedia di [RubyGems](#). AWS SDK for Ruby dirancang untuk menjadi modular dan dipisahkan oleh layanan. Instalasi seluruh `aws-sdk` permata besar dan mungkin memakan waktu lebih dari satu jam.

Kami sarankan hanya menginstal permata untuk yang AWS layanan Anda gunakan. Ini dinamai like `aws-sdk-service_abbreviation` dan daftar lengkapnya ditemukan di tabel [Layanan yang Didukung](#) dari file AWS SDK for Ruby README. Misalnya, permata untuk berinteraksi dengan layanan Amazon S3 tersedia langsung di [aws-sdk-s3](#)

Manajer versi Ruby

Alih-alih menggunakan sistem Ruby, sebaiknya gunakan manajer versi Ruby seperti berikut ini:

- [RVM](#)
- [chruby](#)
- [rbenv](#)

Misalnya, jika Anda menggunakan sistem operasi Amazon Linux 2, perintah berikut dapat digunakan untuk memperbarui RVM, mencantumkan versi Ruby yang tersedia, lalu pilih versi yang ingin Anda gunakan untuk pengembangan dengan AWS SDK for Ruby. Versi Ruby minimum yang dibutuhkan adalah 2.3.

```
$ rvm get head
$ rvm list known
$ rvm install ruby-3.1.3
$ rvm --default use 3.1.3
```

Bundler

Jika Anda menggunakan [Bundler](#), perintah berikut menginstal AWS SDK for Ruby gem untuk Amazon S3:

1. Instal Bundler dan buat: Gemfile

```
$ gem install bundler
$ bundle init
```

2. Buka yang dibuat Gemfile dan tambahkan gem baris untuk setiap permata AWS layanan yang akan digunakan kode Anda. Untuk mengikuti contoh Amazon S3, tambahkan baris berikut ke bagian bawah file:

```
gem "aws-sdk-s3"
```

3. Simpan Gemfile.

4. Instal dependensi yang ditentukan dalam: Gemfile

```
$ bundle install
```

Halo tutorial untuk AWS SDK for Ruby

Sapa Amazon S3 menggunakan AWS SDK for Ruby. Contoh berikut menampilkan daftar bucket Amazon S3 Anda.

Tulis kode

Salin dan tempel kode berikut ke file sumber baru. Beri nama file `hello-s3.rb`.

```
require "aws-sdk-s3"

# Wraps Amazon S3 resource actions.
class BucketListWrapper
  attr_reader :s3_resource

  # @param s3_resource [Aws::S3::Resource] An Amazon S3 resource.
  def initialize(s3_resource)
    @s3_resource = s3_resource
  end

  # Lists buckets for the current account.
  #
  # @param count [Integer] The maximum number of buckets to list.
```

```
def list_buckets(count)
  puts "Found these buckets:"
  @s3_resource.buckets.each do |bucket|
    puts "\t#{bucket.name}"
    count -= 1
    break if count.zero?
  end
  true
rescue Aws::Errors::ServiceError => e
  puts "Couldn't list buckets. Here's why: #{e.message}"
  false
end
end

# Example usage:
def run_demo
  wrapper = BucketListWrapper.new(Aws::S3::Resource.new)
  wrapper.list_buckets(25)
end

run_demo if $PROGRAM_NAME == __FILE__
```

AWSSDK for Ruby dirancang untuk modular dan dipisahkan oleh. AWS layanan Setelah permata diinstal, `require` pernyataan di bagian atas file sumber Ruby Anda mengimpor kelas dan metode AWS SDK untuk layanan Amazon S3. Untuk daftar lengkap AWS service gems yang tersedia, lihat tabel [Layanan yang Didukung](#) dari file AWS SDK for Ruby README.

```
require 'aws-sdk-s3'
```

Menjalankan program

Buka prompt perintah untuk menjalankan program Ruby Anda. Sintaks perintah khas untuk menjalankan program Ruby adalah:

```
ruby [source filename] [arguments...]
```

Kode contoh ini tidak menggunakan argumen. Untuk menjalankan kode ini, masukkan yang berikut ini ke command prompt:

```
$ ruby hello-s3.rb
```

Catatan untuk pengguna Windows

Saat Anda menggunakan sertifikat SSL di Windows dan menjalankan kode Ruby Anda, Anda mungkin melihat kesalahan yang mirip dengan yang berikut ini.

```
C:\Ruby>ruby buckets.rb
C:/Ruby200-x64/lib/ruby/2.0.0/net/http.rb:921:in `connect': SSL_connect returned=1
  errno=0 state=SSLv3 read server certificate B: certificate verify failed
  (Seahorse::Client::NetworkingError)
    from C:/Ruby200-x64/lib/ruby/2.0.0/net/http.rb:921:in `block in connect'

    from C:/Ruby200-x64/lib/ruby/2.0.0/timeout.rb:66:in `timeout'
    from C:/Ruby200-x64/lib/ruby/2.0.0/net/http.rb:921:in `connect'
    from C:/Ruby200-x64/lib/ruby/2.0.0/net/http.rb:862:in `do_start'
    from C:/Ruby200-x64/lib/ruby/2.0.0/net/http.rb:857:in `start'
...

```

Untuk memperbaiki masalah ini, tambahkan baris berikut ke file sumber Ruby Anda, di suatu tempat sebelum AWS panggilan pertama Anda.

```
Aws.use_bundled_cert!
```

Jika Anda hanya menggunakan `aws-sdk-s3` permata dalam program Ruby Anda dan Anda ingin menggunakan sertifikat yang dibundel, Anda juga perlu menambahkan permata `aws-sdk-core`

Langkah selanjutnya

Untuk menguji banyak operasi Amazon S3 lainnya, lihat [Repositori Contoh AWS Kode](#) di GitHub

Gunakan AWS Cloud9 dengan AWS SDK untuk Ruby

AWS Cloud9 adalah lingkungan pengembangan terintegrasi berbasis web (IDE) yang berisi kumpulan alat yang Anda gunakan untuk membuat kode, membangun, menjalankan, menguji, men-debug, dan merilis perangkat lunak di cloud. Anda dapat menggunakan AWS Cloud9 dengan AWS SDK for Ruby untuk menulis dan menjalankan kode Ruby Anda dengan menggunakan browser. AWS Cloud9 termasuk alat-alat seperti editor kode dan terminal. Karena berbasis cloud, Anda dapat mengerjakan proyek Anda dari kantor, rumah, atau di mana saja dengan menggunakan mesin yang terhubung ke internet. AWS Cloud9 IDE Untuk informasi umum tentang AWS Cloud9, lihat [Panduan AWS Cloud9 Pengguna](#).

Ikuti petunjuk ini untuk mengatur AWS Cloud9 dengan Ruby: AWS SDK

- [Langkah 1: Siapkan Akun AWS untuk Anda gunakan AWS Cloud9](#)
- [Langkah 2: Siapkan lingkungan AWS Cloud9 pengembangan Anda](#)
- [Langkah 3: Siapkan AWS SDK Ruby](#)
- [Langkah 4: Unduh kode contoh](#)
- [Langkah 5: Jalankan kode contoh](#)

Langkah 1: Siapkan Akun AWS untuk Anda gunakan AWS Cloud9

Untuk menggunakan AWS Cloud9, masuk ke AWS Cloud9 konsol dari file AWS Management Console.

Note

Jika Anda menggunakan AWS IAM Identity Center untuk mengautentikasi, Anda mungkin perlu menambahkan izin yang diperlukan `iam:ListInstanceProfilesForRole` ke kebijakan yang dilampirkan pengguna di konsol. IAM

Untuk menyiapkan IAM entitas di AWS akun Anda agar dapat mengakses AWS Cloud9 dan masuk ke AWS Cloud9 konsol, lihat [Pengaturan Tim AWS Cloud9](#) di Panduan AWS Cloud9 Pengguna.

Langkah 2: Siapkan lingkungan AWS Cloud9 pengembangan Anda

Setelah Anda masuk ke AWS Cloud9 konsol, gunakan konsol untuk membuat lingkungan AWS Cloud9 pengembangan. Setelah Anda menciptakan lingkungan, AWS Cloud9 buka IDE untuk lingkungan itu.

Untuk detailnya, lihat [Membuat Lingkungan AWS Cloud9 di](#) Panduan AWS Cloud9 Pengguna.

Note

Saat Anda membuat lingkungan di konsol untuk pertama kalinya, kami sarankan Anda memilih opsi untuk Membuat instance baru untuk environment (EC2). Opsi ini memberi tahu AWS Cloud9 untuk membuat lingkungan, meluncurkan EC2 instance Amazon, dan kemudian

menghubungkan instance baru ke lingkungan baru. Ini adalah cara tercepat untuk mulai menggunakan AWS Cloud9.

Jika terminal belum terbuka di IDE, buka. Pada bilah menu di IDE, pilih Jendela, Terminal Baru. Anda dapat menggunakan jendela terminal untuk menginstal alat dan membangun aplikasi Anda.

Langkah 3: Siapkan AWS SDK Ruby

Setelah AWS Cloud9 membuka IDE untuk lingkungan pengembangan Anda, gunakan jendela terminal untuk mengatur Ruby di lingkungan Anda. AWS SDK

Anda dapat menginstal AWS SDK untuk Ruby seperti halnya permata Ruby. Permata tersedia di [RubyGems](#). The AWS SDK for Ruby dirancang untuk menjadi modular dan dipisahkan oleh AWS layanan. Memasang seluruh `aws-sdk` permata itu besar dan mungkin memakan waktu lebih dari satu jam.

Kami sarankan hanya menginstal permata untuk AWS layanan Anda gunakan. Ini diberi nama seperti `aws-sdk-service_abbreviation` dan daftar lengkapnya ditemukan di tabel [Layanan yang Didukung](#) dari README file AWS SDK untuk Ruby. Misalnya, permata untuk berinteraksi dengan layanan Amazon S3 tersedia langsung di [aws-sdk-s3](#)

Manajer versi Ruby

Alih-alih menggunakan sistem Ruby, sebaiknya gunakan pengelola versi Ruby seperti berikut ini:

- [RVM](#)
- [chruby](#)
- [rbenv](#)

Misalnya, jika Anda menggunakan sistem operasi Amazon Linux 2, perintah berikut dapat digunakan untuk memperbarui RVM, daftar versi Ruby yang tersedia, lalu pilih versi yang ingin Anda gunakan untuk pengembangan dengan AWS SDK untuk Ruby. Versi Ruby minimum yang diperlukan adalah 2.3.

```
$ rvm get head
$ rvm list known
$ rvm install ruby-3.1.3
```

```
$ rvm --default use 3.1.3
```

Bundler

Jika Anda menggunakan [Bundler](#), perintah berikut menginstal permata AWS SDK untuk Ruby untuk Amazon S3:

1. Instal Bundler dan buat: Gemfile

```
$ gem install bundler
$ bundle init
```

2. Buka yang dibuat Gemfile dan tambahkan gem baris untuk setiap permata AWS layanan yang akan digunakan kode Anda. Untuk mengikuti contoh Amazon S3, tambahkan baris berikut ke bagian bawah file:

```
gem "aws-sdk-s3"
```

3. Simpan Gemfile.
4. Instal dependensi yang ditentukan dalam: Gemfile

```
$ bundle install
```

Langkah 4: Unduh kode contoh

Gunakan jendela terminal untuk mengunduh kode contoh AWS SDK untuk Ruby ke dalam lingkungan AWS Cloud9 pengembangan.

Untuk mengunduh salinan semua contoh kode yang digunakan dalam AWS SDK dokumentasi resmi ke direktori root lingkungan Anda, jalankan perintah berikut:

```
$ git clone https://github.com/awsdocs/aws-doc-sdk-examples.git
```

Contoh kode AWS SDK untuk Ruby terletak di `ENVIRONMENT_NAME/aws-doc-sdk-examples/ruby` direktori, di `ENVIRONMENT_NAME` mana nama lingkungan pengembangan Anda.

Untuk mengikuti menggunakan contoh Amazon S3, sebaiknya mulai dengan contoh kode. `ENVIRONMENT_NAME/aws-doc-sdk-examples/ruby/example_code/s3/bucket_list.rb` Gunakan jendela terminal untuk menavigasi ke `s3` direktori dan daftar file.

```
$ cd aws-doc-sdk-examples/ruby/example_code/s3
$ ls
```

Untuk membuka file di AWS Cloud9, Anda dapat mengklik `bucket_list.rb` langsung di jendela terminal.

Langkah 5: Jalankan kode contoh

Untuk menjalankan kode di lingkungan AWS Cloud9 pengembangan Anda, pilih tombol Run di bilah menu atas. AWS Cloud9 akan secara otomatis mendeteksi ekstensi `.rb` file dan menggunakan runner Ruby untuk menjalankan kode. Untuk informasi selengkapnya tentang menjalankan kode AWS Cloud9, lihat [Menjalankan Kode Anda](#) di Panduan AWS Cloud9 Pengguna.

Pada tangkapan layar berikut, perhatikan area dasar ini:

- 1: Lari. Tombol Run terletak di bilah menu atas. Ini membuka tab baru untuk hasil Anda.

Note

Anda juga dapat membuat konfigurasi run baru secara manual. Pada bilah menu, pilih Jalankan, Jalankan Konfigurasi, Jalankan Konfigurasi Baru.

- 2: Perintah. AWS Cloud9 mengisi kotak teks Command dengan path dan nama file ke file yang Anda jalankan. Jika kode Anda mengharapkan parameter baris perintah apa pun yang akan diteruskan, ini dapat ditambahkan ke baris perintah dengan cara yang sama seperti yang Anda lakukan saat menjalankan kode melalui jendela terminal.
- 3: Pelari. AWS Cloud9 mendeteksi bahwa ekstensi file Anda `.rb` dan memilih Ruby Runner untuk menjalankan kode Anda.

```
Go Run Tools Window Support Preview Run A Share
bucket_list.rb x +
9 require "aws-sdk-s3"
10
11 # Wraps Amazon S3 resource actions.
12 class BucketListWrapper
13   attr_reader :s3_resource
14
15   # @param s3_resource [Aws::S3::Resource] An Amazon S3 resource.
16   def initialize(s3_resource)
17     @s3_resource = s3_resource
18   end
19
20   # Lists buckets for the current account.
21   #
```

1

2

3

bash - "ip-172-31-35-38.ec x aws-doc-sdk-examples/ru x

Run Command: aws-doc-sdk-examples/ruby/example_code/s3/bucket_list.rb Runner: Ruby CWD ENV

Found these buckets:

Setiap output yang dihasilkan dari kode yang sedang berjalan ditampilkan di tab.

Untuk menguji banyak operasi Amazon S3 lainnya, lihat [Repositori Contoh AWS Kode](#) di GitHub

Konfigurasi AWS SDK untuk Ruby

Pelajari cara mengkonfigurasi AWS SDK untuk Ruby. Anda harus menetapkan bagaimana kode Anda mengotentikasi dengan AWS ketika Anda mengembangkan dengan AWS layanan. Anda juga harus mengatur yang ingin Wilayah AWS Anda gunakan.

Rantai penyedia kredensi

Semua SDKs memiliki serangkaian tempat (atau sumber) yang mereka periksa untuk mendapatkan kredensial yang valid untuk digunakan untuk membuat permintaan ke sebuah AWS layanan. Setelah kredensial yang valid ditemukan, pencarian dihentikan. Pencarian sistematis ini disebut rantai penyedia kredensial default.

Untuk setiap langkah dalam rantai, ada berbagai cara untuk mengatur nilai. Menetapkan nilai secara langsung dalam kode selalu diutamakan, diikuti dengan pengaturan sebagai variabel lingkungan, dan kemudian di file bersama AWS config. Untuk informasi selengkapnya, lihat [Prioritas pengaturan di Panduan Referensi](#) Alat AWS SDKs dan Alat.

Panduan Referensi AWS SDKs and Tools memiliki informasi tentang pengaturan SDK konfigurasi yang digunakan oleh semua AWS SDKs dan AWS CLI. Untuk mempelajari lebih lanjut tentang cara mengonfigurasi SDK melalui AWS config file bersama, lihat File [konfigurasi dan kredensial bersama](#). Untuk mempelajari lebih lanjut tentang cara mengonfigurasi variabel lingkungan SDK melalui pengaturan, lihat [Dukungan variabel lingkungan](#).

Untuk mengotentikasi AWS, AWS SDK for Ruby memeriksa penyedia kredensial dalam urutan yang tercantum dalam tabel berikut.

Penyedia kredensial berdasarkan prioritas	AWS SDK dan Panduan Referensi Alat	AWS SDK for Ruby API Referensi
AWS kunci akses (kredensial sementara dan jangka panjang)	AWS kunci akses	Aws::Credentials Aws::SharedCredentials
Token identitas web dari AWS Security Token Service (AWS STS)	Asumsikan penyedia kredensial peran	Aws::AssumeRoleWebIdentityCredentials

Penyedia kredensi berdasarkan prioritas	AWS SDK dan Panduan Referensi Alat	AWS SDK for Ruby API Referensi
	Menggunakan <code>anrole_arn</code> , <code>role_session_name</code> , dan <code>web_identity_token_file</code>	
AWS IAM Identity Center. Dalam panduan ini, lihat Otentikasi SDK dengan AWS .	IAM Penyedia kredensi Pusat Identitas	Aws::SSOCredentials
Penyedia entitas tepercaya (seperti <code>AWS_ROLE_ARN</code>). Dalam panduan ini, lihat Membuat token AWS STS akses .	Asumsikan penyedia kredensi peran Menggunakan <code>role_arn</code> dan <code>role_session_name</code>	Aws::AssumeRoleCredentials
Penyedia kredensi proses	Penyedia kredensi proses	Aws::ProcessCredentials
Kredensi Amazon Elastic Container Service (Amazon ECS)	Penyedia kredensi kontainer	Aws::ECSCredentials
Kredensi profil instans Amazon Elastic Compute Cloud (Amazon EC2) (IMDS penyedia kredensi)	IMDS penyedia kredensi	Aws::InstanceProfileCredentials

Jika variabel lingkungan AWS SDK for Ruby `AWS_SDK_CONFIG_OPT_OUT` disetel, AWS config file bersama, biasanya di `~/.aws/config`, tidak akan diurai untuk kredensialnya.

Jika Anda mengikuti pendekatan yang disarankan bagi pengguna baru untuk memulai, Anda menyiapkan AWS IAM Identity Center [Otentikasi SDK dengan AWS](#) autentikasi selama topik Memulai. Metode otentikasi lainnya berguna untuk situasi yang berbeda. Untuk menghindari risiko

keamanan, kami sarankan untuk selalu menggunakan kredensi jangka pendek. Untuk prosedur metode otentikasi lainnya, lihat [Otentikasi dan akses di Panduan Referensi Alat AWS SDKs dan Alat](#).

Membuat token AWS STS akses

Dengan asumsi peran melibatkan penggunaan seperangkat kredensial keamanan sementara yang dapat Anda gunakan untuk mengakses AWS sumber daya yang biasanya tidak dapat Anda akses. Kredensial sementara ini terdiri dari access key ID, secret access key, dan token keamanan. Anda dapat menggunakan [Aws::AssumeRoleCredentials](#) metode ini untuk membuat token akses AWS Security Token Service (AWS STS).

Contoh berikut menggunakan token akses untuk membuat objek klien Amazon S3, di mana `linked::account::arn` adalah Amazon Resource Name (ARN) dari peran yang akan diambil dan `session-name` merupakan pengidentifikasi untuk sesi peran yang diasumsikan.

```
role_credentials = Aws::AssumeRoleCredentials.new(
  client: Aws::STS::Client.new,
  role_arn: "linked::account::arn",
  role_session_name: "session-name"
)

s3 = Aws::S3::Client.new(credentials: role_credentials)
```

Untuk informasi selengkapnya tentang pengaturan `role_arn` atau `role_session_name`, atau tentang menyetelnya menggunakan AWS config file bersama sebagai gantinya, lihat [Mengasumsikan penyedia kredensi peran](#) di Panduan Referensi Alat AWS SDKs dan.

Mengatur Wilayah

Anda perlu mengatur Wilayah saat menggunakan sebagian besar AWS layanan. The AWS SDK for Ruby mencari Wilayah dengan urutan sebagai berikut:

1. [Mengatur Wilayah di klien atau objek sumber daya](#)
2. [Mengatur Wilayah dengan menggunakan `Aws.config`](#)
3. [Mengatur Wilayah dengan menggunakan variabel lingkungan](#)
4. [Mengatur Wilayah dengan menggunakan config file bersama](#)

Untuk informasi selengkapnya tentang region pengaturan, lihat [Wilayah AWS](#) di Panduan Referensi Alat AWS SDKs dan Alat. Sisa bagian ini menjelaskan cara mengatur Wilayah, dimulai dengan pendekatan yang paling umum.

Mengatur Wilayah menggunakan **config** file bersama

Atur wilayah dengan mengatur region variabel dalam AWS config file bersama. Untuk informasi selengkapnya tentang config file bersama, lihat File [konfigurasi dan kredensial bersama](#) di Panduan Referensi Alat AWS SDKs dan Alat.

Contoh pengaturan nilai ini dalam config file:

```
[default]
region = us-west-2
```

configFile bersama tidak diperiksa jika variabel lingkungan `AWS_SDK_CONFIG_OPT_OUT` disetel.

Mengatur Wilayah menggunakan variabel lingkungan

Mengatur Region dengan mengatur variabel `AWS_REGION` lingkungan.

Gunakan `export` perintah untuk mengatur variabel ini pada sistem berbasis Unix, seperti Linux atau macOS. Contoh berikut menetapkan Region ke `us-west-2`.

```
export AWS_REGION=us-west-2
```

Untuk mengatur variabel ini pada Windows, gunakan `set` perintah. Contoh berikut menetapkan Region ke `us-west-2`.

```
set AWS_REGION=us-west-2
```

Mengatur Wilayah dengan **Aws.config**

Atur Region dengan menambahkan `region` nilai ke `Aws.config` hash. Contoh berikut memperbarui `Aws.config` hash untuk menggunakan `us-west-1` Wilayah.

```
Aws.config.update({region: 'us-west-1'})
```

Setiap klien atau sumber daya yang Anda buat nanti terikat ke Wilayah ini.

Mengatur Wilayah di klien atau objek sumber daya

Atur Wilayah saat Anda membuat AWS klien atau sumber daya. Contoh berikut membuat objek sumber daya Amazon S3 di Wilayah. `us-west-1` Pilih Wilayah yang tepat untuk AWS sumber daya Anda. Objek klien layanan tidak dapat diubah, jadi Anda harus membuat klien baru untuk setiap layanan yang Anda minta dan untuk membuat permintaan ke layanan yang sama menggunakan konfigurasi yang berbeda.

```
s3 = Aws::S3::Resource.new(region: 'us-west-1')
```

Menetapkan titik akhir yang tidak standar

Wilayah ini digunakan untuk membangun SSL titik akhir yang akan digunakan untuk AWS permintaan. Jika Anda perlu menggunakan titik akhir yang tidak standar di Wilayah yang Anda pilih, tambahkan endpoint entri ke `Aws.config` Atau, atur `endpoint:` saat membuat klien layanan atau objek sumber daya. Contoh berikut membuat objek sumber daya Amazon S3 di titik akhir. `other_endpoint`

```
s3 = Aws::S3::Resource.new(endpoint: other_endpoint)
```

Untuk menggunakan titik akhir yang Anda pilih untuk API permintaan dan agar pilihan itu tetap ada, lihat opsi konfigurasi [titik akhir khusus layanan](#) di Panduan Referensi Alat dan AWS SDKs.

Gunakan AWS SDK untuk Ruby

Bagian ini memberikan informasi tentang pengembangan perangkat lunak dengan AWS SDK for Ruby, termasuk cara menggunakan beberapa fitur canggih. SDK

[Panduan Referensi AWS SDKs and Tools](#) juga berisi pengaturan, fitur, dan konsep dasar lainnya yang umum di antara banyak. AWS SDKs

Topik

- [Gunakan utilitas AWS SDK for Ruby REPL](#)
- [Gunakan SDK dengan Ruby on Rails](#)
- [Tip debugging: Dapatkan informasi pelacakan kawat dari klien](#)
- [Respons dan kesalahan klien rintisan](#)
- [Paginasi](#)
- [Pelayan](#)
- [Tentukan perilaku coba lagi klien](#)
- [Bermigrasi dari versi 1 atau 2 ke AWS SDK versi 3 untuk Ruby](#)

Gunakan utilitas AWS SDK for Ruby REPL

`aws-sdk` mencakup antarmuka baris perintah interaktif Read-Eval-Print-Loop (REPL) di mana Anda dapat menguji SDK for Ruby dan segera melihat hasilnya. [SDK for Ruby gems tersedia RubyGems di .org.](#)

Prasyarat

- [Instal AWS SDK for Ruby.](#)
- `aws-v3.rb` terletak di `aws-sdk-resources` permata. `aws-sdk-resources` permata itu juga disertakan oleh `aws-sdk` permata utama.
- Anda akan membutuhkan perpustakaan xml. seperti `rexml` permata.
- Meskipun program ini bekerja dengan Interactive Ruby Shell (`irb`), kami menyarankan Anda menginstal `pry` permata, yang menyediakan lingkungan REPL yang lebih kuat.

Pengaturan bundler

Jika Anda menggunakan [Bundler](#), pembaruan berikut untuk Anda Gemfile akan membahas permata prasyarat:

1. Buka Gemfile yang Anda buat saat menginstal AWS SDK for Ruby. Tambahkan baris berikut ke file:

```
gem "aws-sdk"  
gem "rexml"  
gem "pry"
```

2. Simpan Gemfile.
3. Instal dependensi yang ditentukan dalam: Gemfile

```
$ bundle install
```

Menjalankan REPL

Anda dapat mengakses REPL dengan menjalankan `aws-v3.rb` dari baris perintah.

```
aws-v3.rb
```

Atau, Anda dapat mengaktifkan pencatatan kawat HTTP dengan menyetel flag verbose. HTTP wire logging memberikan informasi tentang komunikasi antara AWS SDK for AWS Ruby dan. Catatan, flag verbose juga menambahkan overhead yang dapat membuat kode Anda berjalan lebih lambat.

```
aws-v3.rb -v
```

SDK for Ruby mencakup kelas klien yang menyediakan antarmuka ke file. AWS layanan Setiap kelas klien mendukung yang tertentu AWS layanan. Dalam REPL, setiap kelas layanan memiliki helper yang mengembalikan objek klien baru untuk berinteraksi dengan layanan itu. Nama pembantu akan menjadi nama layanan yang dikonversi ke huruf kecil. Misalnya, nama-nama objek s3 pembantu Amazon S3 dan Amazon EC2 adalah dan, masing-masing. `ec2` Untuk membuat daftar bucket Amazon S3 di akun Anda, Anda dapat masuk `s3.list_buckets` ke prompt.

Anda dapat mengetik `quit` ke prompt REPL untuk keluar.

Gunakan SDK dengan Ruby on Rails

[Ruby on Rails](#) menyediakan kerangka pengembangan web yang membuatnya mudah untuk membuat situs web dengan Ruby.

AWS menyediakan `aws-sdk-rails` permata untuk memungkinkan integrasi yang mudah dengan Rails. Anda dapat menggunakan AWS Elastic Beanstalk, AWS OpsWorks, AWS CodeDeploy, atau [Penyedia AWS Rails untuk menyebarkan](#) dan menjalankan aplikasi Rails Anda di Cloud. AWS

Untuk informasi tentang menginstal dan menggunakan `aws-sdk-rails` permata, lihat GitHub <https://github.com/aws/aws-sdk-rails> repositori.

Tip debugging: Dapatkan informasi pelacakan kawat dari klien

Anda bisa mendapatkan informasi wire trace dari AWS klien dengan menetapkan `http_wire_trace` Boolean. Informasi pelacakan kawat membantu membedakan perubahan klien, masalah layanan, dan kesalahan pengguna. `Kapantrue`, pengaturan menunjukkan apa yang dikirim pada kawat. Contoh berikut membuat klien Amazon S3 dengan pelacakan kawat diaktifkan pada saat pembuatan klien.

```
s3 = Aws::S3::Client.new(http_wire_trace: true)
```

Mengingat kode berikut dan argumen `bucket_name`, output menampilkan pesan yang mengatakan apakah bucket dengan nama yang ada.

```
require 'aws-sdk-s3'

s3 = Aws::S3::Resource.new(client: Aws::S3::Client.new(http_wire_trace: true))

if s3.bucket(ARGV[0]).exists?
  puts "Bucket #{ARGV[0]} exists"
else
  puts "Bucket #{ARGV[0]} does not exist"
end
```

Jika ember ada, output serupa dengan yang berikut ini. (Pengembalian ditambahkan ke HEAD baris untuk keterbacaan.)

```
opening connection to bucket_name.s3-us-west-1.amazonaws.com:443...
opened
```

```
starting SSL for bucket_name.s3-us-west-1.amazonaws.com:443...
SSL established, protocol: TLSv1.2, cipher: ECDHE-RSA-AES128-GCM-SHA256
-> "HEAD / HTTP/1.1
    Accept-Encoding:
    User-Agent: aws-sdk-ruby3/3.171.0 ruby/3.2.2 x86_64-linux aws-sdk-s3/1.120.0
    Host: bucket_name.s3-us-west-1.amazonaws.com
    X-Amz-Date: 20230427T143146Z
/* omitted */
Accept: */*\r\n\r\n"
-> "HTTP/1.1 200 OK\r\n"
-> "x-amz-id-2: XxB2J+kpHgTjmMUwpkUI1EjaFSPxAjWRgkn/+z7YwWc/
iAX5E30XRBzJ37cfc8T4D7ELC1KFELM=\r\n"
-> "x-amz-request-id: 5MD4APQOS815QVBR\r\n"
-> "Date: Thu, 27 Apr 2023 14:31:47 GMT\r\n"
-> "x-amz-bucket-region: us-east-1\r\n"
-> "x-amz-access-point-alias: false\r\n"
-> "Content-Type: application/xml\r\n"
-> "Server: AmazonS3\r\n"
-> "\r\n"
Conn keep-alive
Bucket bucket_name exists
```

Anda juga dapat mengaktifkan pelacakan kawat setelah pembuatan klien.

```
s3 = Aws::S3::Client.new
s3.config.http_wire_trace = true
```

Untuk informasi selengkapnya tentang bidang dalam informasi pelacakan kawat yang dilaporkan, lihat [Header permintaan wajib Transfer Family](#).

Respons dan kesalahan klien rintisan

Pelajari cara menghentikan respons klien dan kesalahan klien dalam aplikasi AWS SDK for Ruby.

Tanggapan klien yang mematahkan

Ketika Anda menghentikan respons, AWS SDK for Ruby menonaktifkan lalu lintas jaringan dan klien mengembalikan data stubbed (atau palsu). Jika Anda tidak menyediakan data stubbed, klien akan mengembalikan:

- Daftar sebagai array kosong

- Maps sebagai hash kosong
- Nilai numerik sebagai nol
- Tanggal sebagai now

Contoh berikut menampilkan nama stubbed untuk daftar bucket Amazon S3.

```
require 'aws-sdk'

s3 = Aws::S3::Client.new(stub_responses: true)

bucket_data = s3.stub_data(:list_buckets, :buckets => [{name:'aws-sdk'}, {name:'aws-
sdk2'}])
s3.stub_responses(:list_buckets, bucket_data)
bucket_names = s3.list_buckets.buckets.map(&:name)

# List each bucket by name
bucket_names.each do |name|
  puts name
end
```

Menjalankan kode ini menampilkan berikut ini.

```
aws-sdk
aws-sdk2
```

Note

Setelah Anda menyediakan data stubbed, nilai default tidak lagi berlaku untuk atribut instans yang tersisa. Ini berarti bahwa dalam contoh sebelumnya, atribut instance yang tersisa `creation_date`, tidak `now` tapinil.

AWSSDK for Ruby memvalidasi data stubbed Anda. Jika Anda meneruskan data dari jenis yang salah, itu menimbulkan `ArgumentError` pengecualian. Misalnya, jika alih-alih tugas sebelumnya `bucket_data`, Anda menggunakan yang berikut:

```
bucket_data = s3.stub_data(:list_buckets, buckets:['aws-sdk', 'aws-sdk2'])
```

AWSSDK for Ruby memunculkan dua pengecualian. `ArgumentError`

```
expected params[:buckets][0] to be a hash
expected params[:buckets][1] to be a hash
```

Kesalahan klien yang mematahkan

Anda juga dapat melakukan rintisan error yang diajukan AWS SDK for Ruby untuk metode tertentu. Contoh berikut menampilkan `Caught Timeout::Error error calling head_bucket on aws-sdk`.

```
require 'aws-sdk'

s3 = Aws::S3::Client.new(stub_responses: true)
s3.stub_responses(:head_bucket, Timeout::Error)

begin
  s3.head_bucket({bucket: 'aws-sdk'})
rescue Exception => ex
  puts "Caught #{ex.class} error calling 'head_bucket' on 'aws-sdk'"
end
```

Paginasi

Beberapa AWS panggilan memberikan respons paginated untuk membatasi jumlah data yang dikembalikan dengan setiap respons. Halaman data mewakili hingga 1.000 item.

Respons paginated dapat dihitung

Cara paling sederhana untuk menangani data respons paginated adalah dengan menggunakan pencacah built-in dalam objek respons, seperti yang ditunjukkan pada contoh berikut.

```
s3 = Aws::S3::Client.new

s3.list_objects(bucket: 'aws-sdk').each do |response|
  puts response.contents.map(&:key)
end
```

Ini menghasilkan satu objek respons per panggilan API yang dibuat, dan menghitung objek dalam bucket bernama. SDK mengambil halaman data tambahan untuk menyelesaikan permintaan.

Menangani tanggapan paged secara manual

Untuk menangani paging sendiri, gunakan `next_page?` metode respons untuk memverifikasi ada lebih banyak halaman untuk diambil, atau gunakan `last_page?` metode untuk memverifikasi bahwa tidak ada lagi halaman yang akan diambil.

Jika ada lebih banyak halaman, gunakan metode `next_page` (pemberitahuan tidak ada?) untuk mengambil halaman hasil berikutnya, seperti yang ditunjukkan pada contoh berikut.

```
s3 = Aws::S3::Client.new

# Get the first page of data
response = s3.list_objects(bucket:'aws-sdk')

# Get additional pages
while response.next_page? do
  response = response.next_page
  # Use the response data here...
end
```

Note

Jika Anda memanggil `next_page` metode dan tidak ada lagi halaman yang akan diambil, SDK akan memunculkan pengecualian [Aws::PageableResponse::LastPageError](#)

Kelas data paged

Data paged dalam AWS SDK for Ruby ditangani oleh `PageableResponse` kelas [Aws::](#), yang [disertakan dengan Seahorse](#): `:Client`: `:Response` untuk menyediakan akses ke data paged.

Pelayan

Pelayan adalah metode utilitas yang polling untuk negara tertentu terjadi pada klien. Pelayan dapat gagal setelah sejumlah upaya pada interval pemungutan suara yang ditentukan untuk klien layanan. Untuk contoh bagaimana pelayan digunakan, lihat metode [create_table](#) dari Klien Enkripsi Amazon DynamoDB di Repositori Contoh Kode. AWS

Memanggil pelayan

Untuk memanggil pelayan, hubungi klien `wait_until` layanan. Pada contoh berikut, pelayan menunggu sampai instance `i-12345678` berjalan sebelum melanjutkan.

```
ec2 = Aws::EC2::Client.new

begin
  ec2.wait_until(:instance_running, instance_ids:['i-12345678'])
  puts "instance running"
rescue Aws::Writers::Errors::WaiterFailed => error
  puts "failed waiting for instance running: #{error.message}"
end
```

Parameter pertama adalah nama pelayan, yang khusus untuk klien layanan dan menunjukkan operasi mana yang sedang menunggu. Parameter kedua adalah hash parameter yang diteruskan ke metode klien yang disebut oleh pelayan, yang bervariasi sesuai dengan nama pelayan.

Untuk daftar operasi yang dapat ditunggu dan metode klien dipanggil untuk setiap operasi, lihat dokumentasi `waiter_names` dan `wait_until` bidang untuk klien yang Anda gunakan.

Kegagalan tunggu kegagalan

Pelayan dapat gagal dengan salah satu pengecualian berikut.

[Aws::Writers::Errors::FailureStateError](#)

Keadaan kegagalan ditemui saat menunggu.

[Aws::Writers::Errors::NoSuchWaiterError](#)

Nama pelayan yang ditentukan tidak didefinisikan untuk klien yang digunakan.

[Aws::Writers::Errors::TooManyAttemptsError](#)

Jumlah upaya melebihi nilai pelayan. `max_attempts`

[Aws::Writers::Errors::UnexpectedError](#)

Terjadi galat tak terduga saat menunggu.

[Aws::Writers::Errors::WaiterFailed](#)

Salah satu negara menunggu terlampaui atau kegagalan lain terjadi saat menunggu.

Semua kesalahan ini — kecuali - didasarkan `NoSuchWaiterError` pada `WaiterFailed`. Untuk menangkap kesalahan dalam pelayan, gunakan `WaiterFailed`, seperti yang ditunjukkan dalam contoh berikut.

```
rescue Aws::Waiters::Errors::WaiterFailed => error
  puts "failed waiting for instance running: #{error.message}"
end
```

Mengkonfigurasi pelayan

Setiap pelayan memiliki interval polling default dan jumlah upaya maksimum yang akan dilakukan sebelum mengembalikan kontrol ke program Anda. Untuk mengatur nilai-nilai ini, gunakan `max_attempts` dan `delay`: parameter dalam `wait_until` panggilan Anda. Contoh berikut menunggu hingga 25 detik, polling setiap lima detik.

```
# Poll for ~25 seconds
client.wait_until(...) do |w|
  w.max_attempts = 5
  w.delay = 5
end
```

Untuk menonaktifkan kegagalan tunggu, tetapkan nilai salah satu parameter ini ke `nil`.

Memperluas pelayan

Untuk mengubah perilaku pelayan, Anda dapat mendaftarkan callback yang dipicu sebelum setiap upaya polling dan sebelum menunggu.

Contoh berikut mengimplementasikan cadangan eksponensial dalam pelayan dengan menggandakan jumlah waktu untuk menunggu setiap upaya.

```
ec2 = Aws::EC2::Client.new

ec2.wait_until(:instance_running, instance_ids:['i-12345678']) do |w|
  w.interval = 0 # disable normal sleep
  w.before_wait do |n, resp|
    sleep(n ** 2)
  end
end
```

Contoh berikut menonaktifkan jumlah maksimum upaya, dan sebaliknya menunggu selama satu jam (3600 detik) sebelum gagal.

```
started_at = Time.now
client.wait_until(...) do |w|
  # Disable max attempts
  w.max_attempts = nil

  # Poll for one hour, instead of a number of attempts
  w.before_wait do |attempts, response|
    throw :failure if Time.now - started_at > 3600
  end
end
```

Tentukan perilaku coba lagi klien

Secara default, AWS SDK for Ruby melakukan hingga tiga percobaan ulang, dengan 15 detik antara percobaan ulang, dengan total hingga empat upaya. Oleh karena itu, operasi bisa memakan waktu hingga 60 detik untuk time out.

Contoh berikut membuat klien Amazon S3 di wilayah tersebutus-west-2, dan menetapkan untuk menunggu lima detik antara dua percobaan ulang pada setiap operasi klien. Oleh karena itu, operasi klien Amazon S3 dapat memakan waktu hingga 15 detik untuk waktu habis.

```
s3 = Aws::S3::Client.new(
  region: region,
  retry_limit: 2,
  retry_backoff: lambda { |c| sleep(5) }
)
```

Contoh ini menunjukkan cara mengubah parameter coba lagi secara langsung di dalam kode. Namun, Anda juga dapat menggunakan variabel lingkungan atau AWS config file bersama untuk mengatur ini untuk aplikasi Anda. Untuk informasi selengkapnya tentang setelan ini, lihat [Coba lagi perilaku](#) di Panduan Referensi AWS SDK dan Alat. Pengaturan eksplisit apa pun yang disetel dalam kode atau pada klien layanan itu sendiri lebih diutamakan daripada yang ditetapkan dalam variabel lingkungan atau file bersama. config

Bermigrasi dari versi 1 atau 2 ke AWS SDK versi 3 untuk Ruby

Tujuan dari topik ini adalah untuk membantu Anda bermigrasi dari AWS SDK versi 1 atau 2 untuk Ruby ke versi 3.

Side-by-side Penggunaan

Tidak perlu mengganti AWS SDK untuk Ruby versi 1 atau 2 dengan versi 3. Anda dapat menggunakannya bersama-sama dalam aplikasi yang sama. Lihat [posting blog ini](#) untuk informasi lebih lanjut.

Contoh cepat berikut.

```
require 'aws-sdk-v1' # version 1
require 'aws-sdk'    # version 2
require 'aws-sdk-s3' # version 3

s3 = AWS::S3::Client.new # version 1
s3 = Aws::S3::Client.new # version 2 or 3
```

Anda tidak perlu menulis ulang kode versi 1 atau 2 yang sudah ada untuk mulai menggunakan SDK versi 3. Strategi migrasi yang valid adalah hanya menulis kode baru terhadap SDK versi 3.

Perbedaan umum

Versi 3 berbeda dari versi 2 dengan satu cara penting.

- Setiap layanan tersedia sebagai permata terpisah.

Versi 2 berbeda dari versi 1 dalam beberapa cara penting.

- Namespace root yang berbeda - `Aws` versus `AWS`. Hal ini memungkinkan side-by-side penggunaan.
- `Aws.config`- Sekarang vanilla Ruby hash, bukan metode.
- Opsi konstruktor yang ketat - Saat membangun objek klien atau sumber daya dalam SDK versi 1, opsi konstruktor yang tidak dikenal diabaikan. Dalam versi 2, pilihan konstruktor yang tidak diketahui memicu `ArgumentError`. Misalnya:

```
# version 1
AWS::S3::Client.new(http_read_timeout: 10)
```

```
# oops, typo'd option is ignored

# version 2
Aws::S3::Client.new(http_reed_timeout: 10)
# => raises ArgumentError
```

Perbedaan klien

Tidak ada perbedaan antara kelas klien di versi 2 dan versi 3.

Antara versi 1 dan versi 2, kelas klien memiliki perbedaan eksternal paling sedikit. Banyak klien layanan akan memiliki antarmuka yang kompatibel setelah konstruksi klien. Beberapa perbedaan penting:

- `Aws::S3::Client`- Versi 1 kelas klien Amazon S3 adalah tangan-kode. Versi 2 dihasilkan dari model layanan. Nama metode dan input sangat berbeda dalam versi 2.
- `Aws::EC2::Client`- Versi 2 menggunakan nama jamak untuk daftar output, versi 1 menggunakan `akhiran_set`. Misalnya:

```
# version 1
resp = AWS::EC2::Client.new.describe_security_groups
resp.security_group_set
#=> [...]

# version 2
resp = Aws::EC2::Client.new.describe_security_groups
resp.security_groups
#=> [...]
```

- `Aws::SWF::Client`- Versi 2 menggunakan respons terstruktur, di mana versi 1 menggunakan vanilla Ruby hash.
- Layanan kelas mengubah nama - Versi 2 menggunakan nama yang berbeda untuk beberapa layanan:
 - `AWS::SimpleWorkflow` telah menjadi `Aws::SWF`
 - `AWS::ELB` telah menjadi `Aws::ElasticLoadBalancing`
 - `AWS::SimpleEmailService` telah menjadi `Aws::SES`
- Opsi konfigurasi klien - Beberapa opsi konfigurasi versi 1 diganti namanya dalam versi 2. Lainnya dihapus atau diganti. Berikut adalah perubahan yang mungkin:

- `:use_ssl` telah dihapus. Versi 2 menggunakan SSL di mana-mana. Untuk menonaktifkan SSL Anda harus mengkonfigurasi `:endpoint` yang menggunakan `http://`.
- `:ssl_ca_file` sekarang `:ssl_ca_bundle`
- `:ssl_ca_path` sekarang `:ssl_ca_directory`
- Ditambahkan `:ssl_ca_store`.
- `:endpoint` sekarang harus menjadi HTTP atau HTTPS URI yang memenuhi syarat, bukan nama host.
- Dihapus `:_port` pilihan untuk setiap layanan, sekarang digantikan oleh `:endpoint`.
- `:user_agent_prefix` sekarang `:user_agent_suffix`

Perbedaan sumber daya

Tidak ada perbedaan antara antarmuka sumber daya di versi 2 dan versi 3.

Ada perbedaan yang signifikan antara antarmuka sumber daya dalam versi 1 dan versi 2. Versi 1 sepenuhnya tangan-kode, di mana sebagai versi 2 antarmuka sumber daya yang dihasilkan dari model. Antarmuka sumber daya versi 2 secara signifikan lebih konsisten. Beberapa perbedaan sistemik meliputi:

- Kelas sumber daya terpisah - Dalam versi 2, nama layanan adalah modul, bukan kelas. Dalam modul ini, itu adalah antarmuka sumber daya:

```
# version 1
s3 = AWS::S3.new

# version 2
s3 = Aws::S3::Resource.new
```

- Sumber daya referensi - SDK versi 2 memisahkan koleksi dan pengambil sumber daya individual menjadi dua metode berbeda:

```
# version 1
s3.buckets['bucket-name'].objects['key'].delete

# version 2
s3.bucket('bucket-name').object('key').delete
```

- Operasi Batch - Dalam versi 1, semua operasi batch adalah utilitas dengan kode tangan. Dalam versi 2, banyak operasi batch yang autogenerated operasi batching atas API. Antarmuka batching versi 2 sangat berbeda dari versi 1.

SDK untuk contoh kode Ruby

Contoh kode dalam topik ini menunjukkan cara menggunakan AWS SDK for Ruby with AWS.

Dasar-dasar adalah contoh kode yang menunjukkan kepada Anda bagaimana melakukan operasi penting dalam suatu layanan.

Tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Skenario adalah contoh kode yang menunjukkan kepada Anda bagaimana menyelesaikan tugas tertentu dengan memanggil beberapa fungsi dalam layanan atau dikombinasikan dengan yang lain AWS layanan.

Layanan

- [Contoh Aurora menggunakan Ruby SDK](#)
- [Contoh Auto Scaling menggunakan Ruby SDK](#)
- [CloudTrail contoh menggunakan SDK untuk Ruby](#)
- [CloudWatch contoh menggunakan SDK untuk Ruby](#)
- [Contoh Penyedia Identitas Amazon Cognito menggunakan Ruby SDK](#)
- [Amazon SDK Comprehend contoh yang digunakan untuk Ruby](#)
- [Amazon DocumentDB contoh menggunakan untuk Ruby SDK](#)
- [Contoh DynamoDB menggunakan untuk Ruby SDK](#)
- [EC2Contoh Amazon menggunakan SDK untuk Ruby](#)
- [Contoh Elastic SDK Beanstalk yang digunakan untuk Ruby](#)
- [EventBridge contoh menggunakan SDK untuk Ruby](#)
- [AWS Glue contoh menggunakan SDK untuk Ruby](#)
- [IAMcontoh menggunakan SDK untuk Ruby](#)
- [Contoh Kinesis yang digunakan SDK untuk Ruby](#)
- [AWS KMS contoh menggunakan SDK untuk Ruby](#)
- [Contoh Lambda menggunakan SDK untuk Ruby](#)
- [MSKContoh Amazon menggunakan SDK untuk Ruby](#)
- [Contoh Amazon Polly menggunakan Ruby SDK](#)

- [RDS Contoh Amazon menggunakan SDK untuk Ruby](#)
- [Contoh Amazon S3 menggunakan SDK untuk Ruby](#)
- [SESContoh Amazon menggunakan SDK untuk Ruby](#)
- [Amazon SES API v2 contoh menggunakan SDK untuk Ruby](#)
- [SNSContoh Amazon menggunakan SDK untuk Ruby](#)
- [SQSContoh Amazon menggunakan SDK untuk Ruby](#)
- [AWS STS contoh menggunakan SDK untuk Ruby](#)
- [Contoh Amazon Texttract yang digunakan SDK untuk Ruby](#)
- [Contoh Amazon Translate menggunakan SDK Ruby](#)
- [WorkDocs Contoh Amazon menggunakan SDK untuk Ruby](#)

Contoh Aurora menggunakan Ruby SDK

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan AWS SDK for Ruby with Aurora.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Memulai

Halo Aurora

Contoh kode berikut ini menunjukkan cara mulai menggunakan Aurora.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
require 'aws-sdk-rds'  
  
# Creates an Amazon RDS client for the AWS Region
```

```
rds = Aws::RDS::Client.new

puts 'Listing clusters in this AWS account...'

# Calls the describe_db_clusters method to get information about clusters
resp = rds.describe_db_clusters(max_records: 20)

# Checks if any clusters are found and prints the appropriate message
if resp.db_clusters.empty?
  puts 'No clusters found!'
else
  # Loops through the array of cluster objects and prints the cluster identifier
  resp.db_clusters.each do |cluster|
    puts "Cluster identifier: #{cluster.db_cluster_identifier}"
  end
end
```

- Untuk API detailnya, lihat [DescribeDBClusters](#) di AWS SDK for Ruby API Referensi.

Contoh Auto Scaling menggunakan Ruby SDK

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan Auto Scaling AWS SDK for Ruby with.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Memulai

Halo Auto Scaling

Contoh kode berikut menunjukkan cara memulai menggunakan Auto Scaling.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
require 'aws-sdk-autoscaling'
require 'logger'

# AutoScalingManager is a class responsible for managing AWS Auto Scaling operations
# such as listing all Auto Scaling groups in the current AWS account.
class AutoScalingManager
  def initialize(client)
    @client = client
    @logger = Logger.new($stdout)
  end

  # Gets and prints a list of Auto Scaling groups for the account.
  def list_auto_scaling_groups
    paginator = @client.describe_auto_scaling_groups
    auto_scaling_groups = []
    paginator.each_page do |page|
      auto_scaling_groups.concat(page.auto_scaling_groups)
    end

    if auto_scaling_groups.empty?
      @logger.info('No Auto Scaling groups found for this account.')
    else
      auto_scaling_groups.each do |group|
        @logger.info("Auto Scaling group name: #{group.auto_scaling_group_name}")
        @logger.info("  Group ARN:           #{group.auto_scaling_group_arn}")
        @logger.info("  Min/max/desired:      #{group.min_size}/#{group.max_size}/
#{group.desired_capacity}")
        @logger.info("\n")
      end
    end
  end

  end

  end

  if $PROGRAM_NAME == __FILE__
    autoscaling_client = Aws::AutoScaling::Client.new
    manager = AutoScalingManager.new(autoscaling_client)
    manager.list_auto_scaling_groups
  end
end
```

- Untuk API detailnya, lihat [DescribeAutoScalingGroups](#) di AWS SDK for Ruby API Referensi.

CloudTrail contoh menggunakan SDK untuk Ruby

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan AWS SDK for Ruby with CloudTrail.

Tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

CreateTrail

Contoh kode berikut menunjukkan cara menggunakan `CreateTrail`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
require "aws-sdk-cloudtrail" # v2: require 'aws-sdk'
require "aws-sdk-s3"
require "aws-sdk-sts"

def create_trail_example(s3_client, sts_client, cloudtrail_client, trail_name,
  bucket_name)

  resp = sts_client.get_caller_identity({})
  account_id = resp.account

  # Attach policy to an Amazon Simple Storage Service (S3) bucket.
```

```
s3_client.create_bucket(bucket: bucket_name)
begin
  policy = {
    "Version" => "2012-10-17",
    "Statement" => [
      {
        "Sid" => "AWSCloudTrailAclCheck20150319",
        "Effect" => "Allow",
        "Principal" => {
          "Service" => "cloudtrail.amazonaws.com"
        },
        "Action" => "s3:GetBucketAcl",
        "Resource" => "arn:aws:s3:::#{bucket_name}"
      },
      {
        "Sid" => "AWSCloudTrailWrite20150319",
        "Effect" => "Allow",
        "Principal" => {
          "Service" => "cloudtrail.amazonaws.com"
        },
        "Action" => "s3:PutObject",
        "Resource" => "arn:aws:s3:::#{bucket_name}/AWSLogs/#{account_id}/*",
        "Condition" => {
          "StringEquals" => {
            "s3:x-amz-acl" => "bucket-owner-full-control"
          }
        }
      }
    ]
  }.to_json

  s3_client.put_bucket_policy(
    bucket: bucket_name,
    policy: policy
  )
  puts "Successfully added policy to bucket #{bucket_name}"
end

begin
  cloudtrail_client.create_trail({
    name: trail_name, # required
    s3_bucket_name: bucket_name # required
  })
end
```

```
puts "Successfully created trail: #{trail_name}."
rescue StandardError => e
  puts "Got error trying to create trail #{trail_name}:\n #{e}"
  puts e
  exit 1
end
```

- Untuk API detailnya, lihat [CreateTrail](#) di AWS SDK for Ruby API Referensi.

DeleteTrail

Contoh kode berikut menunjukkan cara menggunakan `DeleteTrail`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
client.delete_trail({
  name: trail_name # required
})
puts "Successfully deleted trail: " + trail_name
rescue StandardError => err
  puts "Got error trying to delete trail: " + trail_name + ":"
  puts err
  exit 1
end
```

- Untuk API detailnya, lihat [DeleteTrail](#) di AWS SDK for Ruby API Referensi.

ListTrails

Contoh kode berikut menunjukkan cara menggunakan `ListTrails`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
require "aws-sdk-cloudtrail" # v2: require 'aws-sdk'

def describe_trails_example(client)
  resp = client.describe_trails({})
  puts "Found #{resp.trail_list.count} trail(s)."
  resp.trail_list.each do |trail|
    puts "Name:          " + trail.name
    puts "S3 bucket name: " + trail.s3_bucket_name
    puts
  end
end
```

- Untuk API detailnya, lihat [List Trails](#) di AWS SDK for Ruby API Referensi.

Lookup Events

Contoh kode berikut menunjukkan cara menggunakan `LookupEvents`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
require "aws-sdk-cloudtrail" # v2: require 'aws-sdk'

# @param [Object] client
def lookup_events_example(client)
```

```
resp = client.lookup_events
puts "Found #{resp.events.count} events:"
resp.events.each do |e|
  puts "Event name:   #{e.event_name}"
  puts "Event ID:     #{e.event_id}"
  puts "Event time:   #{e.event_time}"
  puts "Resources:"

  e.resources.each do |r|
    puts "  Name:       #{r.resource_name}"
    puts "  Type:       #{r.resource_type}"
    puts ""
  end
end
end
```

- Untuk API detailnya, lihat [LookupEvents](#) di AWS SDK for Ruby API Referensi.

CloudWatch contoh menggunakan SDK untuk Ruby

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan AWS SDK for Ruby with CloudWatch.

Tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

DescribeAlarms

Contoh kode berikut menunjukkan cara menggunakan `DescribeAlarms`.

SDK untuk Ruby

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
require "aws-sdk-cloudwatch"

# Lists the names of available Amazon CloudWatch alarms.
#
# @param cloudwatch_client [Aws::CloudWatch::Client]
#   An initialized CloudWatch client.
# @example
#   list_alarms(Aws::CloudWatch::Client.new(region: 'us-east-1'))
def list_alarms(cloudwatch_client)
  response = cloudwatch_client.describe_alarms
  if response.metric_alarms.count.positive?
    response.metric_alarms.each do |alarm|
      puts alarm.alarm_name
    end
  else
    puts "No alarms found."
  end
rescue StandardError => e
  puts "Error getting information about alarms: #{e.message}"
end
```

- Untuk API detailnya, lihat [DescribeAlarms](#) di AWS SDK for Ruby API Referensi.

DescribeAlarmsForMetric

Contoh kode berikut menunjukkan cara menggunakan `DescribeAlarmsForMetric`.

SDK untuk Ruby

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
#
# @param cloudwatch_client [Aws::CloudWatch::Client]
#   An initialized CloudWatch client.
# @example
#   describe_metric_alarms(Aws::CloudWatch::Client.new(region: 'us-east-1'))
def describe_metric_alarms(cloudwatch_client)
  response = cloudwatch_client.describe_alarms

  if response.metric_alarms.count.positive?
    response.metric_alarms.each do |alarm|
      puts "-" * 16
      puts "Name:           " + alarm.alarm_name
      puts "State value:      " + alarm.state_value
      puts "State reason:     " + alarm.state_reason
      puts "Metric:           " + alarm.metric_name
      puts "Namespace:        " + alarm.namespace
      puts "Statistic:         " + alarm.statistic
      puts "Period:            " + alarm.period.to_s
      puts "Unit:              " + alarm.unit.to_s
      puts "Eval. periods:    " + alarm.evaluation_periods.to_s
      puts "Threshold:         " + alarm.threshold.to_s
      puts "Comp. operator:   " + alarm.comparison_operator

      if alarm.key?(:ok_actions) && alarm.ok_actions.count.positive?
        puts "OK actions:"
        alarm.ok_actions.each do |a|
          puts "  " + a
        end
      end

      if alarm.key?(:alarm_actions) && alarm.alarm_actions.count.positive?
        puts "Alarm actions:"
        alarm.alarm_actions.each do |a|
```

```
        puts " " + a
      end
    end

    if alarm.key?(:insufficient_data_actions) &&
      alarm.insufficient_data_actions.count.positive?
      puts "Insufficient data actions:"
      alarm.insufficient_data_actions.each do |a|
        puts " " + a
      end
    end

    puts "Dimensions:"
    if alarm.key?(:dimensions) && alarm.dimensions.count.positive?
      alarm.dimensions.each do |d|
        puts " Name: " + d.name + ", Value: " + d.value
      end
    else
      puts " None for this alarm."
    end
  end
else
  puts "No alarms found."
end
rescue StandardError => e
  puts "Error getting information about alarms: #{e.message}"
end

# Example usage:
def run_me
  region = ""

  # Print usage information and then stop.
  if ARGV[0] == "--help" || ARGV[0] == "-h"
    puts "Usage:  ruby cw-ruby-example-show-alarms.rb REGION"
    puts "Example: ruby cw-ruby-example-show-alarms.rb us-east-1"
    exit 1
  # If no values are specified at the command prompt, use these default values.
  elsif ARGV.count.zero?
    region = "us-east-1"
  # Otherwise, use the values as specified at the command prompt.
  else
    region = ARGV[0]
  end
end
```

```

cloudwatch_client = Aws::CloudWatch::Client.new(region: region)
puts "Available alarms:"
describe_metric_alarms(cloudwatch_client)
end

run_me if $PROGRAM_NAME == __FILE__

```

- Untuk API detailnya, lihat [DescribeAlarmsForMetric](#) di AWS SDK for Ruby API Referensi.

DisableAlarmActions

Contoh kode berikut menunjukkan cara menggunakan `DisableAlarmActions`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```

# Disables an alarm in Amazon CloudWatch.
#
# Prerequisites.
#
# - The alarm to disable.
#
# @param cloudwatch_client [Aws::CloudWatch::Client]
#   An initialized CloudWatch client.
# @param alarm_name [String] The name of the alarm to disable.
# @return [Boolean] true if the alarm was disabled; otherwise, false.
# @example
#   exit 1 unless alarm_actions_disabled?(
#     Aws::CloudWatch::Client.new(region: 'us-east-1'),
#     'ObjectsInBucket'
#   )
def alarm_actions_disabled?(cloudwatch_client, alarm_name)
  cloudwatch_client.disable_alarm_actions(alarm_names: [alarm_name])
  return true
end

```

```
rescue StandardError => e
  puts "Error disabling alarm actions: #{e.message}"
  return false
end

# Example usage:
def run_me
  alarm_name = "ObjectsInBucket"
  alarm_description = "Objects exist in this bucket for more than 1 day."
  metric_name = "NumberOfObjects"
  # Notify this Amazon Simple Notification Service (Amazon SNS) topic when
  # the alarm transitions to the ALARM state.
  alarm_actions = ["arn:aws:sns:us-
east-1:111111111111:Default_CloudWatch_Alarms_Topic"]
  namespace = "AWS/S3"
  statistic = "Average"
  dimensions = [
    {
      name: "BucketName",
      value: "doc-example-bucket"
    },
    {
      name: "StorageType",
      value: "AllStorageTypes"
    }
  ]
  period = 86_400 # Daily (24 hours * 60 minutes * 60 seconds = 86400 seconds).
  unit = "Count"
  evaluation_periods = 1 # More than one day.
  threshold = 1 # One object.
  comparison_operator = "GreaterThanThreshold" # More than one object.
  # Replace us-west-2 with the AWS Region you're using for Amazon CloudWatch.
  region = "us-east-1"

  cloudwatch_client = Aws::CloudWatch::Client.new(region: region)

  if alarm_created_or_updated?(
    cloudwatch_client,
    alarm_name,
    alarm_description,
    metric_name,
    alarm_actions,
    namespace,
    statistic,
```

```

    dimensions,
    period,
    unit,
    evaluation_periods,
    threshold,
    comparison_operator
  )
  puts "Alarm '#{alarm_name}' created or updated."
else
  puts "Could not create or update alarm '#{alarm_name}'."
end

if alarm_actions_disabled?(cloudwatch_client, alarm_name)
  puts "Alarm '#{alarm_name}' disabled."
else
  puts "Could not disable alarm '#{alarm_name}'."
end
end
end

run_me if $PROGRAM_NAME == __FILE__

```

- Untuk API detailnya, lihat [DisableAlarmActions](#) di AWS SDK for Ruby API Referensi.

ListMetrics

Contoh kode berikut menunjukkan cara menggunakan `ListMetrics`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```

# Lists available metrics for a metric namespace in Amazon CloudWatch.
#
# @param cloudwatch_client [Aws::CloudWatch::Client]
#   An initialized CloudWatch client.
# @param metric_namespace [String] The namespace of the metric.
# @example

```

```
# list_metrics_for_namespace(
#   Aws::CloudWatch::Client.new(region: 'us-east-1'),
#   'SITE/TRAFFIC'
# )
def list_metrics_for_namespace(cloudwatch_client, metric_namespace)
  response = cloudwatch_client.list_metrics(namespace: metric_namespace)

  if response.metrics.count.positive?
    response.metrics.each do |metric|
      puts " Metric name: #{metric.metric_name}"
      if metric.dimensions.count.positive?
        puts "   Dimensions:"
        metric.dimensions.each do |dimension|
          puts "     Name: #{dimension.name}, Value: #{dimension.value}"
        end
      else
        puts "No dimensions found."
      end
    end
  else
    puts "No metrics found for namespace '#{metric_namespace}'. " \
      "Note that it could take up to 15 minutes for recently-added metrics " \
      "to become available."
  end
end

# Example usage:
def run_me
  metric_namespace = "SITE/TRAFFIC"
  # Replace us-west-2 with the AWS Region you're using for Amazon CloudWatch.
  region = "us-east-1"

  cloudwatch_client = Aws::CloudWatch::Client.new(region: region)

  # Add three datapoints.
  puts "Continuing..." unless datapoint_added_to_metric?(
    cloudwatch_client,
    metric_namespace,
    "UniqueVisitors",
    "SiteName",
    "example.com",
    5_885.0,
    "Count"
  )
end
```

```
puts "Continuing..." unless datapoint_added_to_metric?(
  cloudwatch_client,
  metric_namespace,
  "UniqueVisits",
  "SiteName",
  "example.com",
  8_628.0,
  "Count"
)

puts "Continuing..." unless datapoint_added_to_metric?(
  cloudwatch_client,
  metric_namespace,
  "PageViews",
  "PageURL",
  "example.html",
  18_057.0,
  "Count"
)

puts "Metrics for namespace '#{metric_namespace}':"
list_metrics_for_namespace(cloudwatch_client, metric_namespace)
end

run_me if $PROGRAM_NAME == __FILE__
```

- Untuk API detailnya, lihat [ListMetrics](#) di AWS SDK for Ruby API Referensi.

PutMetricAlarm

Contoh kode berikut menunjukkan cara menggunakan `PutMetricAlarm`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).


```
# Creates or updates an alarm in Amazon CloudWatch.
#
# @param cloudwatch_client [Aws::CloudWatch::Client]
#   An initialized CloudWatch client.
# @param alarm_name [String] The name of the alarm.
# @param alarm_description [String] A description about the alarm.
# @param metric_name [String] The name of the metric associated with the alarm.
# @param alarm_actions [Array] A list of Strings representing the
#   Amazon Resource Names (ARNs) to execute when the alarm transitions to the
#   ALARM state.
# @param namespace [String] The namespace for the metric to alarm on.
# @param statistic [String] The statistic for the metric.
# @param dimensions [Array] A list of dimensions for the metric, specified as
#   Aws::CloudWatch::Types::Dimension.
# @param period [Integer] The number of seconds before re-evaluating the metric.
# @param unit [String] The unit of measure for the statistic.
# @param evaluation_periods [Integer] The number of periods over which data is
#   compared to the specified threshold.
# @param threshold [Float] The value against which the specified statistic is
#   compared.
# @param comparison_operator [String] The arithmetic operation to use when
#   comparing the specified statistic and threshold.
# @return [Boolean] true if the alarm was created or updated; otherwise, false.
# @example
#   exit 1 unless alarm_created_or_updated?(
#     Aws::CloudWatch::Client.new(region: 'us-east-1'),
#     'ObjectsInBucket',
#     'Objects exist in this bucket for more than 1 day.',
#     'NumberOfObjects',
#     ['arn:aws:sns:us-east-1:111111111111:Default_CloudWatch_Alarms_Topic'],
#     'AWS/S3',
#     'Average',
#     [
#       {
#         name: 'BucketName',
#         value: 'doc-example-bucket'
#       },
#       {
#         name: 'StorageType',
#         value: 'AllStorageTypes'
#       }
#     ],
#     86_400,
```

```
# 'Count',
# 1,
# 1,
# 'GreaterThanThreshold'
# )
def alarm_created_or_updated?(
  cloudwatch_client,
  alarm_name,
  alarm_description,
  metric_name,
  alarm_actions,
  namespace,
  statistic,
  dimensions,
  period,
  unit,
  evaluation_periods,
  threshold,
  comparison_operator
)
  cloudwatch_client.put_metric_alarm(
    alarm_name: alarm_name,
    alarm_description: alarm_description,
    metric_name: metric_name,
    alarm_actions: alarm_actions,
    namespace: namespace,
    statistic: statistic,
    dimensions: dimensions,
    period: period,
    unit: unit,
    evaluation_periods: evaluation_periods,
    threshold: threshold,
    comparison_operator: comparison_operator
  )
  return true
rescue StandardError => e
  puts "Error creating alarm: #{e.message}"
  return false
end
```

- Untuk API detailnya, lihat [PutMetricAlarm](#) di AWS SDK for Ruby API Referensi.

PutMetricData

Contoh kode berikut menunjukkan cara menggunakan `PutMetricData`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
require "aws-sdk-cloudwatch"

# Adds a datapoint to a metric in Amazon CloudWatch.
#
# @param cloudwatch_client [Aws::CloudWatch::Client]
#   An initialized CloudWatch client.
# @param metric_namespace [String] The namespace of the metric to add the
#   datapoint to.
# @param metric_name [String] The name of the metric to add the datapoint to.
# @param dimension_name [String] The name of the dimension to add the
#   datapoint to.
# @param dimension_value [String] The value of the dimension to add the
#   datapoint to.
# @param metric_value [Float] The value of the datapoint.
# @param metric_unit [String] The unit of measurement for the datapoint.
# @return [Boolean]
# @example
#   exit 1 unless datapoint_added_to_metric?(
#     Aws::CloudWatch::Client.new(region: 'us-east-1'),
#     'SITE/TRAFFIC',
#     'UniqueVisitors',
#     'SiteName',
#     'example.com',
#     5_885.0,
#     'Count'
#   )
def datapoint_added_to_metric?(
  cloudwatch_client,
  metric_namespace,
  metric_name,
  dimension_name,
```

```
    dimension_value,  
    metric_value,  
    metric_unit  
  )  
  cloudwatch_client.put_metric_data(  
    namespace: metric_namespace,  
    metric_data: [  
      {  
        metric_name: metric_name,  
        dimensions: [  
          {  
            name: dimension_name,  
            value: dimension_value  
          }  
        ],  
        value: metric_value,  
        unit: metric_unit  
      }  
    ]  
  )  
  puts "Added data about '#{metric_name}' to namespace " \  
    "'#{metric_namespace}'."  
  return true  
rescue StandardError => e  
  puts "Error adding data about '#{metric_name}' to namespace " \  
    "'#{metric_namespace}': #{e.message}"  
  return false  
end
```

- Untuk API detailnya, lihat [PutMetricData](#) di AWS SDK for Ruby API Referensi.

Contoh Penyedia Identitas Amazon Cognito menggunakan Ruby SDK

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan Penyedia Identitas Amazon Cognito AWS SDK for Ruby dengan.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Memulai

Halo Amazon Cognito

Contoh kode berikut menunjukkan cara memulai menggunakan Amazon Cognito.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
require 'aws-sdk-cognitoidentityprovider'
require 'logger'

# CognitoManager is a class responsible for managing AWS Cognito operations
# such as listing all user pools in the current AWS account.
class CognitoManager
  def initialize(client)
    @client = client
    @logger = Logger.new($stdout)
  end

  # Lists and prints all user pools associated with the AWS account.
  def list_user_pools
    paginator = @client.list_user_pools(max_results: 10)
    user_pools = []
    paginator.each_page do |page|
      user_pools.concat(page.user_pools)
    end

    if user_pools.empty?
      @logger.info('No Cognito user pools found.')
    else
      user_pools.each do |user_pool|
        @logger.info("User pool ID: #{user_pool.id}")
        @logger.info("User pool name: #{user_pool.name}")
        @logger.info("User pool status: #{user_pool.status}")
        @logger.info('---')
      end
    end
  end
end
```

```
        end
      end
    end
  end

  if $PROGRAM_NAME == __FILE__
    cognito_client = Aws::CognitoIdentityProvider::Client.new
    manager = CognitoManager.new(cognito_client)
    manager.list_user_pools
  end
```

- Untuk API detailnya, lihat [ListUserPools](#) di AWS SDK for Ruby API Referensi.

Amazon SDK Comprehend contoh yang digunakan untuk Ruby

Contoh kode berikut menunjukkan cara melakukan tindakan dan menerapkan skenario umum dengan menggunakan Amazon AWS SDK for Ruby Comprehend.

Skenario adalah contoh kode yang menunjukkan kepada Anda bagaimana menyelesaikan tugas tertentu dengan memanggil beberapa fungsi dalam layanan atau dikombinasikan dengan yang lain AWS layanan.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Skenario](#)

Skenario

Buat aplikasi untuk menganalisis umpan balik pelanggan

Contoh kode berikut menunjukkan cara membuat aplikasi yang menganalisis kartu komentar pelanggan, menerjemahkannya dari bahasa aslinya, menentukan sentimen mereka, dan menghasilkan file audio dari teks yang diterjemahkan.

SDK untuk Ruby

Aplikasi contoh ini menganalisis dan menyimpan kartu umpan balik pelanggan. Secara khusus, ini memenuhi kebutuhan hotel fiktif di New York City. Hotel menerima umpan balik dari para tamu dalam berbagai bahasa dalam bentuk kartu komentar fisik. Umpan balik itu diunggah ke aplikasi melalui klien web. Setelah gambar kartu komentar diunggah, langkah-langkah berikut terjadi:

- Teks diekstraksi dari gambar menggunakan Amazon Textract.
- Amazon Comprehend menentukan sentimen teks yang diekstraksi dan bahasanya.
- Teks yang diekstraksi diterjemahkan ke bahasa Inggris menggunakan Amazon Translate.
- Amazon Polly mensintesis file audio dari teks yang diekstraksi.

Aplikasi lengkap dapat digunakan dengan AWS CDK Untuk kode sumber dan petunjuk penerapan, lihat proyek di [GitHub](#).

Layanan yang digunakan dalam contoh ini

- Amazon Comprehend
- Lambda
- Amazon Polly
- Amazon Textract
- Amazon Translate

Amazon DocumentDB contoh menggunakan untuk Ruby SDK

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan Amazon DocumentDB. AWS SDK for Ruby

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Contoh nirserver](#)

Contoh nirserver

Memanggil fungsi Lambda dari pemicu Amazon DocumentDB

Contoh kode berikut menunjukkan cara menerapkan fungsi Lambda yang menerima peristiwa yang dipicu dengan menerima catatan dari aliran perubahan DocumentDB. Fungsi mengambil payload DocumentDB dan mencatat isi catatan.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di repositori [contoh Nirserver](#).

Mengonsumsi acara Amazon DocumentDB dengan Lambda menggunakan Ruby.

```
require 'json'

def lambda_handler(event:, context:)
  event['events'].each do |record|
    log_document_db_event(record)
  end
  'OK'
end

def log_document_db_event(record)
  event_data = record['event'] || {}
  operation_type = event_data['operationType'] || 'Unknown'
  db = event_data.dig('ns', 'db') || 'Unknown'
  collection = event_data.dig('ns', 'coll') || 'Unknown'
  full_document = event_data['fullDocument'] || {}

  puts "Operation type: #{operation_type}"
  puts "db: #{db}"
  puts "collection: #{collection}"
  puts "Full document: #{JSON.pretty_generate(full_document)}"
end
```


Contoh DynamoDB menggunakan untuk Ruby SDK

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan AWS SDK for Ruby with DynamoDB.

Dasar-dasar adalah contoh kode yang menunjukkan kepada Anda bagaimana melakukan operasi penting dalam suatu layanan.

Tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Skenario adalah contoh kode yang menunjukkan kepada Anda bagaimana menyelesaikan tugas tertentu dengan memanggil beberapa fungsi dalam layanan atau dikombinasikan dengan yang lain AWS layanan.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Memulai

Halo DynamoDB

Contoh kode berikut ini menunjukkan cara untuk mulai menggunakan DynamoDB.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
require 'aws-sdk-dynamodb'
require 'logger'

# DynamoDBManager is a class responsible for managing DynamoDB operations
# such as listing all tables in the current AWS account.
class DynamoDBManager
  def initialize(client)
```

```
@client = client
@logger = Logger.new($stdout)
end

# Lists and prints all DynamoDB tables in the current AWS account.
def list_tables
  @logger.info('Here are the DynamoDB tables in your account:')

  paginator = @client.list_tables(limit: 10)
  table_names = []

  paginator.each_page do |page|
    page.table_names.each do |table_name|
      @logger.info("- #{table_name}")
      table_names << table_name
    end
  end

  if table_names.empty?
    @logger.info("You don't have any DynamoDB tables in your account.")
  else
    @logger.info("\nFound #{table_names.length} tables.")
  end
end

if $PROGRAM_NAME == __FILE__
  dynamodb_client = Aws::DynamoDB::Client.new
  manager = DynamoDBManager.new(dynamodb_client)
  manager.list_tables
end
```

- Untuk API detailnya, lihat [ListTables](#) di AWS SDK for Ruby API Referensi.

Topik

- [Hal-hal mendasar](#)
- [Tindakan](#)
- [Skenario](#)
- [Contoh nirservis](#)

Hal-hal mendasar

Pelajari dasar-dasarnya

Contoh kode berikut ini menunjukkan cara:

- Buat tabel yang dapat menyimpan data film.
- Masukkan, dapatkan, dan perbarui satu film dalam tabel tersebut.
- Tulis data film ke tabel dari JSON file sampel.
- Kueri untuk film yang dirilis pada tahun tertentu.
- Pindai film yang dirilis dalam suatu rentang tahun.
- Hapus film dari tabel, lalu hapus tabel tersebut.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

Buat kelas yang merangkum tabel DynamoDB.

```
# Creates an Amazon DynamoDB table that can be used to store movie data.
# The table uses the release year of the movie as the partition key and the
# title as the sort key.
#
# @param table_name [String] The name of the table to create.
# @return [Aws::DynamoDB::Table] The newly created table.
def create_table(table_name)
  @table = @dynamo_resource.create_table(
    table_name: table_name,
    key_schema: [
      {attribute_name: "year", key_type: "HASH"}, # Partition key
      {attribute_name: "title", key_type: "RANGE"} # Sort key
    ],
    attribute_definitions: [
      {attribute_name: "year", attribute_type: "N"},
```

```

      {attribute_name: "title", attribute_type: "S"}
    ],
    provisioned_throughput: {read_capacity_units: 10, write_capacity_units: 10})
  @dynamo_resource.client.wait_until(:table_exists, table_name: table_name)
  @table
rescue Aws::DynamoDB::Errors::ServiceError => e
  @logger.error("Failed create table #{table_name}:\n#{e.code}: #{e.message}")
  raise
end

```

Buat fungsi pembantu untuk mengunduh dan mengekstrak JSON file sampel.

```

# Gets sample movie data, either from a local file or by first downloading it from
# the Amazon DynamoDB Developer Guide.
#
# @param movie_file_name [String] The local file name where the movie data is
# stored in JSON format.
# @return [Hash] The movie data as a Hash.
def fetch_movie_data(movie_file_name)
  if !File.file?(movie_file_name)
    @logger.debug("Downloading #{movie_file_name}...")
    movie_content = URI.open(
      "https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/samples/
moviedata.zip"
    )
    movie_json = ""
    Zip::File.open_buffer(movie_content) do |zip|
      zip.each do |entry|
        movie_json = entry.get_input_stream.read
      end
    end
  else
    movie_json = File.read(movie_file_name)
  end
  movie_data = JSON.parse(movie_json)
  # The sample file lists over 4000 movies. This returns only the first 250.
  movie_data.slice(0, 250)
rescue StandardError => e
  puts("Failure downloading movie data:\n#{e}")
  raise
end

```

Jalankan skenario interaktif untuk membuat tabel dan melakukan tindakan pada tabel tersebut.

```
table_name = "doc-example-table-movies-#{rand(10**4)}"
scaffold = Scaffold.new(table_name)
dynamodb_wrapper = DynamoDBBasics.new(table_name)

new_step(1, "Create a new DynamoDB table if none already exists.")
unless scaffold.exists?(table_name)
  puts("\nNo such table: #{table_name}. Creating it...")
  scaffold.create_table(table_name)
  print "Done!\n".green
end

new_step(2, "Add a new record to the DynamoDB table.")
my_movie = {}
my_movie[:title] = CLI::UI::Prompt.ask("Enter the title of a movie to add to the
table. E.g. The Matrix")
my_movie[:year] = CLI::UI::Prompt.ask("What year was it released? E.g. 1989").to_i
my_movie[:rating] = CLI::UI::Prompt.ask("On a scale of 1 - 10, how do you rate it?
E.g. 7").to_i
my_movie[:plot] = CLI::UI::Prompt.ask("Enter a brief summary of the plot. E.g. A
man awakens to a new reality.")
dynamodb_wrapper.add_item(my_movie)
puts("\nNew record added:")
puts JSON.pretty_generate(my_movie).green
print "Done!\n".green

new_step(3, "Update a record in the DynamoDB table.")
my_movie[:rating] = CLI::UI::Prompt.ask("Let's update the movie you added with a
new rating, e.g. 3:").to_i
response = dynamodb_wrapper.update_item(my_movie)
puts("Updated '#{my_movie[:title]}' with new attributes:")
puts JSON.pretty_generate(response).green
print "Done!\n".green

new_step(4, "Get a record from the DynamoDB table.")
puts("Searching for #{my_movie[:title]} (#{my_movie[:year]})...")
response = dynamodb_wrapper.get_item(my_movie[:title], my_movie[:year])
puts JSON.pretty_generate(response).green
print "Done!\n".green

new_step(5, "Write a batch of items into the DynamoDB table.")
download_file = "moviedata.json"
puts("Downloading movie database to #{download_file}...")
```

```
movie_data = scaffold.fetch_movie_data(download_file)
puts("Writing movie data from #{download_file} into your table...")
scaffold.write_batch(movie_data)
puts("Records added: #{movie_data.length}.")
print "Done!\n".green

new_step(5, "Query for a batch of items by key.")
loop do
  release_year = CLI::UI::Prompt.ask("Enter a year between 1972 and 2018, e.g.
1999:").to_i
  results = dynamodb_wrapper.query_items(release_year)
  if results.any?
    puts("There were #{results.length} movies released in #{release_year}:")
    results.each do |movie|
      print "\t #{movie["title"]}".green
    end
    break
  else
    continue = CLI::UI::Prompt.ask("Found no movies released in #{release_year}!
Try another year? (y/n)")
    break if !continue.eql?("y")
  end
end
print "\nDone!\n".green

new_step(6, "Scan for a batch of items using a filter expression.")
years = {}
years[:start] = CLI::UI::Prompt.ask("Enter a starting year between 1972 and
2018:")
years[:end] = CLI::UI::Prompt.ask("Enter an ending year between 1972 and 2018:")
releases = dynamodb_wrapper.scan_items(years)
if !releases.empty?
  puts("Found #{releases.length} movies.")
  count = Question.ask(
    "How many do you want to see? ", method(:is_int), in_range(1,
releases.length))
  puts("Here are your #{count} movies:")
  releases.take(count).each do |release|
    puts("\t#{release["title"]}")
  end
else
  puts("I don't know about any movies released between #{years[:start]} "\
    "and #{years[:end]}".")
end
```

```
print "\nDone!\n".green

new_step(7, "Delete an item from the DynamoDB table.")
answer = CLI::UI::Prompt.ask("Do you want to remove '#{my_movie[:title]}'? (y/n)
")
if answer.eql?("y")
  dynamodb_wrapper.delete_item(my_movie[:title], my_movie[:year])
  puts("Removed '#{my_movie[:title]}' from the table.")
  print "\nDone!\n".green
end

new_step(8, "Delete the DynamoDB table.")
answer = CLI::UI::Prompt.ask("Delete the table? (y/n)")
if answer.eql?("y")
  scaffold.delete_table
  puts("Deleted #{table_name}.")
else
  puts("Don't forget to delete the table when you're done!")
end
print "\nThanks for watching!\n".green
rescue Aws::Errors::ServiceError
  puts("Something went wrong with the demo.")
rescue Errno::ENOENT
  true
end
```

- Untuk API detailnya, lihat topik berikut di AWS SDK for Ruby API Referensi.

- [BatchWriteItem](#)
- [CreateTable](#)
- [DeleteItem](#)
- [DeleteTable](#)
- [DescribeTable](#)
- [GetItem](#)
- [PutItem](#)
- [Kueri](#)
- [Scan](#)
- [UpdateItem](#)

Tindakan

BatchExecuteStatement

Contoh kode berikut menunjukkan cara menggunakan `BatchExecuteStatement`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

Baca batch item menggunakan PartiQL.

```
class DynamoDBPartiQLBatch

  attr_reader :dynamo_resource
  attr_reader :table

  def initialize(table_name)
    client = Aws::DynamoDB::Client.new(region: "us-east-1")
    @dynamodb = Aws::DynamoDB::Resource.new(client: client)
    @table = @dynamodb.table(table_name)
  end

  # Selects a batch of items from a table using PartiQL
  #
  # @param batch_titles [Array] Collection of movie titles
  # @return [Aws::DynamoDB::Types::BatchExecuteStatementOutput]
  def batch_execute_select(batch_titles)
    request_items = batch_titles.map do |title, year|
      {
        statement: "SELECT * FROM \"#{@table.name}\" WHERE title=? and year=?",
        parameters: [title, year]
      }
    end
    @dynamodb.client.batch_execute_statement({statements: request_items})
  end
end
```


Hapus batch item menggunakan PartiQL.

```
class DynamoDBPartiQLBatch

  attr_reader :dynamo_resource
  attr_reader :table

  def initialize(table_name)
    client = Aws::DynamoDB::Client.new(region: "us-east-1")
    @dynamodb = Aws::DynamoDB::Resource.new(client: client)
    @table = @dynamodb.table(table_name)
  end

  # Deletes a batch of items from a table using PartiQL
  #
  # @param batch_titles [Array] Collection of movie titles
  # @return [Aws::DynamoDB::Types::BatchExecuteStatementOutput]
  def batch_execute_write(batch_titles)
    request_items = batch_titles.map do |title, year|
      {
        statement: "DELETE FROM \"#{@table.name}\" WHERE title=? and year=?",
        parameters: [title, year]
      }
    end
    @dynamodb.client.batch_execute_statement({statements: request_items})
  end
end
```

- Untuk API detailnya, lihat [BatchExecuteStatement](#) di AWS SDK for Ruby API Referensi.

BatchWriteItem

Contoh kode berikut menunjukkan cara menggunakan `BatchWriteItem`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
class DynamoDBBasics
  attr_reader :dynamo_resource
  attr_reader :table

  def initialize(table_name)
    client = Aws::DynamoDB::Client.new(region: "us-east-1")
    @dynamo_resource = Aws::DynamoDB::Resource.new(client: client)
    @table = @dynamo_resource.table(table_name)
  end

  # Fills an Amazon DynamoDB table with the specified data. Items are sent in
  # batches of 25 until all items are written.
  #
  # @param movies [Enumerable] The data to put in the table. Each item must contain
  # at least
  #           the keys required by the schema that was specified
  # when the
  #           table was created.
  def write_batch(movies)
    index = 0
    slice_size = 25
    while index < movies.length
      movie_items = []
      movies[index, slice_size].each do |movie|
        movie_items.append({put_request: { item: movie }})
      end
      @dynamo_resource.client.batch_write_item({request_items: { @table.name =>
movie_items }})
      index += slice_size
    end
  rescue Aws::DynamoDB::Errors::ServiceError => e
    puts(
      "Couldn't load data into table #{@table.name}. Here's why:")
    puts("\t#{e.code}: #{e.message}")
    raise
  end
end
```

- Untuk API detailnya, lihat [BatchWriteItem](#) di AWS SDK for Ruby API Referensi.

CreateTable

Contoh kode berikut menunjukkan cara menggunakan CreateTable.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
# Encapsulates an Amazon DynamoDB table of movie data.
class Scaffold
  attr_reader :dynamo_resource
  attr_reader :table_name
  attr_reader :table

  def initialize(table_name)
    client = Aws::DynamoDB::Client.new(region: "us-east-1")
    @dynamo_resource = Aws::DynamoDB::Resource.new(client: client)
    @table_name = table_name
    @table = nil
    @logger = Logger.new($stdout)
    @logger.level = Logger::DEBUG
  end

  # Creates an Amazon DynamoDB table that can be used to store movie data.
  # The table uses the release year of the movie as the partition key and the
  # title as the sort key.
  #
  # @param table_name [String] The name of the table to create.
  # @return [Aws::DynamoDB::Table] The newly created table.
  def create_table(table_name)
    @table = @dynamo_resource.create_table(
      table_name: table_name,
      key_schema: [
        {attribute_name: "year", key_type: "HASH"}, # Partition key
        {attribute_name: "title", key_type: "RANGE"} # Sort key
      ],
      attribute_definitions: [
        {attribute_name: "year", attribute_type: "N"},
        {attribute_name: "title", attribute_type: "S"}
      ]
    )
  end
end
```

```

    ],
    provisioned_throughput: {read_capacity_units: 10, write_capacity_units: 10})
@dynamo_resource.client.wait_until(:table_exists, table_name: table_name)
@table
rescue Aws::DynamoDB::Errors::ServiceError => e
  @logger.error("Failed create table #{table_name}:\n#{e.code}: #{e.message}")
  raise
end

```

- Untuk API detailnya, lihat [CreateTable](#) di AWS SDK for Ruby API Referensi.

DeleteItem

Contoh kode berikut menunjukkan cara menggunakan `DeleteItem`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```

class DynamoDBBasics
  attr_reader :dynamo_resource
  attr_reader :table

  def initialize(table_name)
    client = Aws::DynamoDB::Client.new(region: "us-east-1")
    @dynamo_resource = Aws::DynamoDB::Resource.new(client: client)
    @table = @dynamo_resource.table(table_name)
  end

  # Deletes a movie from the table.
  #
  # @param title [String] The title of the movie to delete.
  # @param year [Integer] The release year of the movie to delete.
  def delete_item(title, year)
    @table.delete_item(key: {"year" => year, "title" => title})
  rescue Aws::DynamoDB::Errors::ServiceError => e
    puts("Couldn't delete movie #{title}. Here's why:")
  end
end

```

```
puts("\t#{e.code}: #{e.message}")
raise
end
```

- Untuk API detailnya, lihat [DeleteItem](#) di AWS SDK for Ruby API Referensi.

DeleteTable

Contoh kode berikut menunjukkan cara menggunakan `DeleteTable`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
# Encapsulates an Amazon DynamoDB table of movie data.
class Scaffold
  attr_reader :dynamo_resource
  attr_reader :table_name
  attr_reader :table

  def initialize(table_name)
    client = Aws::DynamoDB::Client.new(region: "us-east-1")
    @dynamo_resource = Aws::DynamoDB::Resource.new(client: client)
    @table_name = table_name
    @table = nil
    @logger = Logger.new($stdout)
    @logger.level = Logger::DEBUG
  end

  # Deletes the table.
  def delete_table
    @table.delete
    @table = nil
  rescue Aws::DynamoDB::Errors::ServiceError => e
    puts("Couldn't delete table. Here's why:")
    puts("\t#{e.code}: #{e.message}")
    raise
  end
end
```

```
end
```

- Untuk API detailnya, lihat [DeleteTable](#) di AWS SDK for Ruby API Referensi.

DescribeTable

Contoh kode berikut menunjukkan cara menggunakan `DescribeTable`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
# Encapsulates an Amazon DynamoDB table of movie data.
class Scaffold
  attr_reader :dynamo_resource
  attr_reader :table_name
  attr_reader :table

  def initialize(table_name)
    client = Aws::DynamoDB::Client.new(region: "us-east-1")
    @dynamo_resource = Aws::DynamoDB::Resource.new(client: client)
    @table_name = table_name
    @table = nil
    @logger = Logger.new($stdout)
    @logger.level = Logger::DEBUG
  end

  # Determines whether a table exists. As a side effect, stores the table in
  # a member variable.
  #
  # @param table_name [String] The name of the table to check.
  # @return [Boolean] True when the table exists; otherwise, False.
  def exists?(table_name)
    @dynamo_resource.client.describe_table(table_name: table_name)
    @logger.debug("Table #{table_name} exists")
  rescue Aws::DynamoDB::Errors::ResourceNotFoundException
    @logger.debug("Table #{table_name} doesn't exist")
  end
end
```

```
    false
  rescue Aws::DynamoDB::Errors::ServiceError => e
    puts("Couldn't check for existence of #{table_name}:\n")
    puts("\t#{e.code}: #{e.message}")
    raise
  end
end
```

- Untuk API detailnya, lihat [DescribeTable](#) di AWS SDK for Ruby API Referensi.

ExecuteStatement

Contoh kode berikut menunjukkan cara menggunakan `ExecuteStatement`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

Pilih satu item menggunakan PartiQL.

```
class DynamoDBPartiQLSingle

  attr_reader :dynamo_resource
  attr_reader :table

  def initialize(table_name)
    client = Aws::DynamoDB::Client.new(region: "us-east-1")
    @dynamodb = Aws::DynamoDB::Resource.new(client: client)
    @table = @dynamodb.table(table_name)
  end

  # Gets a single record from a table using PartiQL.
  # Note: To perform more fine-grained selects,
  # use the Client.query instance method instead.
  #
  # @param title [String] The title of the movie to search.
  # @return [Aws::DynamoDB::Types::ExecuteStatementOutput]
  def select_item_by_title(title)
```

```

request = {
  statement: "SELECT * FROM \"#{@table.name}\" WHERE title=?",
  parameters: [title]
}
@dynamodb.client.execute_statement(request)
end

```

Perbarui satu item menggunakan PartiQL.

```

class DynamoDBPartiQLSingle

  attr_reader :dynamo_resource
  attr_reader :table

  def initialize(table_name)
    client = Aws::DynamoDB::Client.new(region: "us-east-1")
    @dynamodb = Aws::DynamoDB::Resource.new(client: client)
    @table = @dynamodb.table(table_name)
  end

  # Updates a single record from a table using PartiQL.
  #
  # @param title [String] The title of the movie to update.
  # @param year [Integer] The year the movie was released.
  # @param rating [Float] The new rating to assign the title.
  # @return [Aws::DynamoDB::Types::ExecuteStatementOutput]
  def update_rating_by_title(title, year, rating)
    request = {
      statement: "UPDATE \"#{@table.name}\" SET info.rating=? WHERE title=? and
year=?",
      parameters: [{ "N": rating }, title, year]
    }
    @dynamodb.client.execute_statement(request)
  end
end

```

Tambahkan satu item menggunakan PartiQL.

```

class DynamoDBPartiQLSingle

  attr_reader :dynamo_resource
  attr_reader :table

```



```

def initialize(table_name)
  client = Aws::DynamoDB::Client.new(region: "us-east-1")
  @dynamodb = Aws::DynamoDB::Resource.new(client: client)
  @table = @dynamodb.table(table_name)
end

# Adds a single record to a table using PartiQL.
#
# @param title [String] The title of the movie to update.
# @param year [Integer] The year the movie was released.
# @param plot [String] The plot of the movie.
# @param rating [Float] The new rating to assign the title.
# @return [Aws::DynamoDB::Types::ExecuteStatementOutput]
def insert_item(title, year, plot, rating)
  request = {
    statement: "INSERT INTO \"#{@table.name}\" VALUE {'title': ?, 'year': ?,
'info': ?}",
    parameters: [title, year, {'plot': plot, 'rating': rating}]
  }
  @dynamodb.client.execute_statement(request)
end

```

Hapus satu item menggunakan PartiQL.

```

class DynamoDBPartiQLSingle

  attr_reader :dynamo_resource
  attr_reader :table

  def initialize(table_name)
    client = Aws::DynamoDB::Client.new(region: "us-east-1")
    @dynamodb = Aws::DynamoDB::Resource.new(client: client)
    @table = @dynamodb.table(table_name)
  end

  # Deletes a single record from a table using PartiQL.
  #
  # @param title [String] The title of the movie to update.
  # @param year [Integer] The year the movie was released.
  # @return [Aws::DynamoDB::Types::ExecuteStatementOutput]
  def delete_item_by_title(title, year)

```

```

request = {
  statement: "DELETE FROM \"#{@table.name}\" WHERE title=? and year=?",
  parameters: [title, year]
}
@dynamodb.client.execute_statement(request)
end

```

- Untuk API detailnya, lihat [ExecuteStatement](#) di AWS SDK for Ruby API Referensi.

GetItem

Contoh kode berikut menunjukkan cara menggunakan `GetItem`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```

class DynamoDBBasics
  attr_reader :dynamo_resource
  attr_reader :table

  def initialize(table_name)
    client = Aws::DynamoDB::Client.new(region: "us-east-1")
    @dynamo_resource = Aws::DynamoDB::Resource.new(client: client)
    @table = @dynamo_resource.table(table_name)
  end

  # Gets movie data from the table for a specific movie.
  #
  # @param title [String] The title of the movie.
  # @param year [Integer] The release year of the movie.
  # @return [Hash] The data about the requested movie.
  def get_item(title, year)
    @table.get_item(key: {"year" => year, "title" => title})
  rescue Aws::DynamoDB::Errors::ServiceError => e
    puts("Couldn't get movie #{title} (#{year}) from table #{@table.name}:\n")
  end
end

```

```
puts("\t#{e.code}: #{e.message}")
raise
end
```

- Untuk API detailnya, lihat [GetItem](#) di AWS SDK for Ruby API Referensi.

ListTables

Contoh kode berikut menunjukkan cara menggunakan `ListTables`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

Tentukan apakah tabel ada.

```
# Encapsulates an Amazon DynamoDB table of movie data.
class Scaffold
  attr_reader :dynamo_resource
  attr_reader :table_name
  attr_reader :table

  def initialize(table_name)
    client = Aws::DynamoDB::Client.new(region: "us-east-1")
    @dynamo_resource = Aws::DynamoDB::Resource.new(client: client)
    @table_name = table_name
    @table = nil
    @logger = Logger.new($stdout)
    @logger.level = Logger::DEBUG
  end

  # Determines whether a table exists. As a side effect, stores the table in
  # a member variable.
  #
  # @param table_name [String] The name of the table to check.
  # @return [Boolean] True when the table exists; otherwise, False.
  def exists?(table_name)
```

```

    @dynamo_resource.client.describe_table(table_name: table_name)
    @logger.debug("Table #{table_name} exists")
  rescue Aws::DynamoDB::Errors::ResourceNotFoundException
    @logger.debug("Table #{table_name} doesn't exist")
    false
  rescue Aws::DynamoDB::Errors::ServiceError => e
    puts("Couldn't check for existence of #{table_name}:\n")
    puts("\t#{e.code}: #{e.message}")
    raise
  end
end

```

- Untuk API detailnya, lihat [ListTables](#) di AWS SDK for Ruby API Referensi.

PutItem

Contoh kode berikut menunjukkan cara menggunakan PutItem.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```

class DynamoDBBasics
  attr_reader :dynamo_resource
  attr_reader :table

  def initialize(table_name)
    client = Aws::DynamoDB::Client.new(region: "us-east-1")
    @dynamo_resource = Aws::DynamoDB::Resource.new(client: client)
    @table = @dynamo_resource.table(table_name)
  end

  # Adds a movie to the table.
  #
  # @param movie [Hash] The title, year, plot, and rating of the movie.
  def add_item(movie)
    @table.put_item(

```

```

    item: {
      "year" => movie[:year],
      "title" => movie[:title],
      "info" => {"plot" => movie[:plot], "rating" => movie[:rating]}})
rescue Aws::DynamoDB::Errors::ServiceError => e
  puts("Couldn't add movie #{title} to table #{@table.name}. Here's why:")
  puts("\t#{e.code}: #{e.message}")
  raise
end

```

- Untuk API detailnya, lihat [PutItem](#) di AWS SDK for Ruby API Referensi.

Query

Contoh kode berikut menunjukkan cara menggunakan Query.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di [Repositori Contoh Kode AWS](#).

```

class DynamoDBBasics
  attr_reader :dynamo_resource
  attr_reader :table

  def initialize(table_name)
    client = Aws::DynamoDB::Client.new(region: "us-east-1")
    @dynamo_resource = Aws::DynamoDB::Resource.new(client: client)
    @table = @dynamo_resource.table(table_name)
  end

  # Queries for movies that were released in the specified year.
  #
  # @param year [Integer] The year to query.
  # @return [Array] The list of movies that were released in the specified year.
  def query_items(year)
    response = @table.query(

```

```

    key_condition_expression: "#yr = :year",
    expression_attribute_names: {"#yr" => "year"},
    expression_attribute_values: {":year" => year})
rescue Aws::DynamoDB::Errors::ServiceError => e
  puts("Couldn't query for movies released in #{year}. Here's why:")
  puts("\t#{e.code}: #{e.message}")
  raise
else
  response.items
end

```

- Untuk API detailnya, lihat [Kueri](#) di AWS SDK for Ruby API Referensi.

Scan

Contoh kode berikut menunjukkan cara menggunakan Scan.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```

class DynamoDBBasics
  attr_reader :dynamo_resource
  attr_reader :table

  def initialize(table_name)
    client = Aws::DynamoDB::Client.new(region: "us-east-1")
    @dynamo_resource = Aws::DynamoDB::Resource.new(client: client)
    @table = @dynamo_resource.table(table_name)
  end

  # Scans for movies that were released in a range of years.
  # Uses a projection expression to return a subset of data for each movie.
  #
  # @param year_range [Hash] The range of years to retrieve.
  # @return [Array] The list of movies released in the specified years.

```

```
def scan_items(year_range)
  movies = []
  scan_hash = {
    filter_expression: "#yr between :start_yr and :end_yr",
    projection_expression: "#yr, title, info.rating",
    expression_attribute_names: {"#yr" => "year"},
    expression_attribute_values: {
      ":start_yr" => year_range[:start], ":end_yr" => year_range[:end]}
  }
  done = false
  start_key = nil
  until done
    scan_hash[:exclusive_start_key] = start_key unless start_key.nil?
    response = @table.scan(scan_hash)
    movies.concat(response.items) unless response.items.empty?
    start_key = response.last_evaluated_key
    done = start_key.nil?
  end
  rescue Aws::DynamoDB::Errors::ServiceError => e
    puts("Couldn't scan for movies. Here's why:")
    puts("\t#{e.code}: #{e.message}")
    raise
  else
    movies
  end
end
```

- Untuk API detailnya, lihat [Memindai](#) di AWS SDK for Ruby API Referensi.

UpdateItem

Contoh kode berikut menunjukkan cara menggunakan UpdateItem.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
class DynamoDBBasics
```

```
attr_reader :dynamo_resource
attr_reader :table

def initialize(table_name)
  client = Aws::DynamoDB::Client.new(region: "us-east-1")
  @dynamo_resource = Aws::DynamoDB::Resource.new(client: client)
  @table = @dynamo_resource.table(table_name)
end

# Updates rating and plot data for a movie in the table.
#
# @param movie [Hash] The title, year, plot, rating of the movie.
def update_item(movie)

  response = @table.update_item(
    key: {"year" => movie[:year], "title" => movie[:title]},
    update_expression: "set info.rating=:r",
    expression_attribute_values: { ":r" => movie[:rating] },
    return_values: "UPDATED_NEW")
  rescue Aws::DynamoDB::Errors::ServiceError => e
    puts("Couldn't update movie #{movie[:title]} (#{movie[:year]}) in table
    #{@table.name}\n")
    puts("\t#{e.code}: #{e.message}")
    raise
  else
    response.attributes
  end
end
```

- Untuk API detailnya, lihat [UpdateItem](#) di AWS SDK for Ruby API Referensi.

Skenario

Melakukan kueri pada tabel menggunakan batch pernyataan PartiQL

Contoh kode berikut ini menunjukkan cara:

- Dapatkan sekumpulan item dengan menjalankan beberapa SELECT pernyataan.
- Tambahkan sekumpulan item dengan menjalankan beberapa INSERT pernyataan.
- Perbarui sekumpulan item dengan menjalankan beberapa UPDATE pernyataan.
- Hapus sekumpulan item dengan menjalankan beberapa DELETE pernyataan.

SDK untuk Ruby

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

Jalankan skenario yang membuat tabel dan menjalankan batch kueri PartiQL.

```
table_name = "doc-example-table-movies-partiql-#{rand(10**4)}"
scaffold = Scaffold.new(table_name)
sdk = DynamoDBPartiQLBatch.new(table_name)

new_step(1, "Create a new DynamoDB table if none already exists.")
unless scaffold.exists?(table_name)
  puts("\nNo such table: #{table_name}. Creating it...")
  scaffold.create_table(table_name)
  print "Done!\n".green
end

new_step(2, "Populate DynamoDB table with movie data.")
download_file = "moviedata.json"
puts("Downloading movie database to #{download_file}...")
movie_data = scaffold.fetch_movie_data(download_file)
puts("Writing movie data from #{download_file} into your table...")
scaffold.write_batch(movie_data)
puts("Records added: #{movie_data.length}.")
print "Done!\n".green

new_step(3, "Select a batch of items from the movies table.")
puts "Let's select some popular movies for side-by-side comparison."
response = sdk.batch_execute_select([["Mean Girls", 2004], ["Goodfellas", 1977],
["The Prancing of the Lambs", 2005]])
puts("Items selected: #{response['responses'].length}\n")
print "\nDone!\n".green

new_step(4, "Delete a batch of items from the movies table.")
sdk.batch_execute_write([["Mean Girls", 2004], ["Goodfellas", 1977], ["The
Prancing of the Lambs", 2005]])
print "\nDone!\n".green

new_step(5, "Delete the table.")
```

```
if scaffold.exists?(table_name)
  scaffold.delete_table
end
end
```


- Untuk API detailnya, lihat [BatchExecuteStatement](#) di AWS SDK for Ruby API Referensi.

Melakukan kueri tabel menggunakan PartiQL

Contoh kode berikut ini menunjukkan cara:

- Dapatkan item dengan menjalankan SELECT pernyataan.
- Tambahkan item dengan menjalankan INSERT pernyataan.
- Perbarui item dengan menjalankan UPDATE pernyataan.
- Hapus item dengan menjalankan DELETE pernyataan.

SDK untuk Ruby

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di [Repositori Contoh Kode AWS](#).

Jalankan skenario yang membuat tabel dan menjalankan kueri PartiQL.

```
table_name = "doc-example-table-movies-partiql-#{rand(10**8)}"
scaffold = Scaffold.new(table_name)
sdk = DynamoDBPartiQLSingle.new(table_name)

new_step(1, "Create a new DynamoDB table if none already exists.")
unless scaffold.exists?(table_name)
  puts("\nNo such table: #{table_name}. Creating it...")
  scaffold.create_table(table_name)
  print "Done!\n".green
end

new_step(2, "Populate DynamoDB table with movie data.")
```

```
download_file = "moviedata.json"
puts("Downloading movie database to #{download_file}...")
movie_data = scaffold.fetch_movie_data(download_file)
puts("Writing movie data from #{download_file} into your table...")
scaffold.write_batch(movie_data)
puts("Records added: #{movie_data.length}.")
print "Done!\n".green

new_step(3, "Select a single item from the movies table.")
response = sdk.select_item_by_title("Star Wars")
puts("Items selected for title 'Star Wars': #{response.items.length}\n")
print "#{response.items.first}".yellow
print "\n\nDone!\n".green

new_step(4, "Update a single item from the movies table.")
puts "Let's correct the rating on The Big Lebowski to 10.0."
sdk.update_rating_by_title("The Big Lebowski", 1998, 10.0)
print "\nDone!\n".green

new_step(5, "Delete a single item from the movies table.")
puts "Let's delete The Silence of the Lambs because it's just too scary."
sdk.delete_item_by_title("The Silence of the Lambs", 1991)
print "\nDone!\n".green

new_step(6, "Insert a new item into the movies table.")
puts "Let's create a less-scary movie called The Prancing of the Lambs."
sdk.insert_item("The Prancing of the Lambs", 2005, "A movie about happy
livestock.", 5.0)
print "\nDone!\n".green

new_step(7, "Delete the table.")
if scaffold.exists?(table_name)
  scaffold.delete_table
end
end
```

- Untuk API detailnya, lihat [ExecuteStatement](#) di AWS SDK for Ruby API Referensi.

Contoh nirserver

Memanggil fungsi Lambda dari pemicu DynamoDB

Contoh kode berikut menunjukkan bagaimana menerapkan fungsi Lambda yang menerima peristiwa yang dipicu oleh menerima catatan dari aliran DynamoDB. Fungsi mengambil payload DynamoDB dan mencatat isi catatan.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di repositori [contoh Nirserver](#).

Mengonsumsi acara DynamoDB dengan Lambda menggunakan Ruby.

```
def lambda_handler(event:, context:)
  return 'received empty event' if event['Records'].empty?

  event['Records'].each do |record|
    log_dynamodb_record(record)
  end

  "Records processed: #{event['Records'].length}"
end

def log_dynamodb_record(record)
  puts record['eventID']
  puts record['eventName']
  puts "DynamoDB Record: #{JSON.generate(record['dynamodb'])}"
end
```

Melaporkan kegagalan item batch untuk fungsi Lambda dengan pemicu DynamoDB

Contoh kode berikut menunjukkan cara mengimplementasikan respons batch sebagian untuk fungsi Lambda yang menerima peristiwa dari aliran DynamoDB. Fungsi melaporkan kegagalan item batch dalam respons, memberi sinyal ke Lambda untuk mencoba lagi pesan tersebut nanti.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di repositori [contoh Nirserver](#).

Melaporkan kegagalan item batch DynamoDB dengan Lambda menggunakan Ruby.

```
def lambda_handler(event:, context:)
  records = event["Records"]
  cur_record_sequence_number = ""

  records.each do |record|
    begin
      # Process your record
      cur_record_sequence_number = record["dynamodb"]["SequenceNumber"]
    rescue StandardError => e
      # Return failed record's sequence number
      return {"batchItemFailures" => [{"itemIdentifier" =>
cur_record_sequence_number}]}
    end
  end

  {"batchItemFailures" => []}
end
```

EC2 Contoh Amazon menggunakan SDK untuk Ruby

Contoh kode berikut menunjukkan kepada Anda cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan AWS SDK for Ruby With AmazonEC2.

Tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Memulai

Halo Amazon EC2

Contoh kode berikut menunjukkan cara memulai menggunakan AmazonEC2.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
require 'aws-sdk-ec2'
require 'logger'

# EC2Manager is a class responsible for managing EC2 operations
# such as listing all EC2 instances in the current AWS account.
class EC2Manager
  def initialize(client)
    @client = client
    @logger = Logger.new($stdout)
  end

  # Lists and prints all EC2 instances in the current AWS account.
  def list_instances
    @logger.info('Listing instances')

    instances = fetch_instances

    if instances.empty?
      @logger.info('You have no instances')
    else
      print_instances(instances)
    end
  end

  private

  # Fetches all EC2 instances using pagination.
  #
  # @return [Array<Aws::EC2::Types::Instance>] List of EC2 instances.
  def fetch_instances
```

```
paginator = @client.describe_instances
instances = []

paginator.each_page do |page|
  page.reservations.each do |reservation|
    reservation.instances.each do |instance|
      instances << instance
    end
  end
end

instances

# Prints details of the given EC2 instances.
#
# @param instances [Array<Aws::EC2::Types::Instance>] List of EC2 instances to
print.
def print_instances(instances)
  instances.each do |instance|
    @logger.info("Instance ID: #{instance.instance_id}")
    @logger.info("Instance Type: #{instance.instance_type}")
    @logger.info("Public IP: #{instance.public_ip_address}")
    @logger.info("Public DNS Name: #{instance.public_dns_name}")
    @logger.info("\n")
  end
end

if $PROGRAM_NAME == __FILE__
  ec2_client = Aws::EC2::Client.new(region: 'us-west-2')
  manager = EC2Manager.new(ec2_client)
  manager.list_instances
end
```

- Untuk API detailnya, lihat [DescribeSecurityGroups](#) di AWS SDK for Ruby API Referensi.

Topik

- [Tindakan](#)

Tindakan

AllocateAddress

Contoh kode berikut menunjukkan cara menggunakan `AllocateAddress`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).


```
# Creates an Elastic IP address in Amazon Virtual Private Cloud (Amazon VPC).
#
# @param ec2_client [Aws::EC2::Client] An initialized EC2 client.
# @return [String] The allocation ID corresponding to the Elastic IP address.
# @example
#   puts allocate_elastic_ip_address(Aws::EC2::Client.new(region: 'us-west-2'))
def allocate_elastic_ip_address(ec2_client)
  response = ec2_client.allocate_address(domain: "vpc")
  return response.allocation_id
rescue StandardError => e
  puts "Error allocating Elastic IP address: #{e.message}"
  return "Error"
end
```

- Untuk API detailnya, lihat [AllocateAddress](#) di AWS SDK for Ruby API Referensi.

AssociateAddress

Contoh kode berikut menunjukkan cara menggunakan `AssociateAddress`.

SDK untuk Ruby

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
# Associates an Elastic IP address with an Amazon Elastic Compute Cloud
# (Amazon EC2) instance.
#
# Prerequisites:
#
# - The allocation ID corresponding to the Elastic IP address.
# - The Amazon EC2 instance.
#
# @param ec2_client [Aws::EC2::Client] An initialized EC2 client.
# @param allocation_id [String] The ID of the allocation corresponding to
#   the Elastic IP address.
# @param instance_id [String] The ID of the instance.
# @return [String] The association ID corresponding to the association of the
#   Elastic IP address to the instance.
# @example
#   puts allocate_elastic_ip_address(
#     Aws::EC2::Client.new(region: 'us-west-2'),
#     'eipalloc-04452e528a66279EX',
#     'i-033c48ef067af3dEX')
def associate_elastic_ip_address_with_instance(
  ec2_client,
  allocation_id,
  instance_id
)
  response = ec2_client.associate_address(
    allocation_id: allocation_id,
    instance_id: instance_id,
  )
  return response.association_id
rescue StandardError => e
  puts "Error associating Elastic IP address with instance: #{e.message}"
  return "Error"
end
```

- Untuk API detailnya, lihat [AssociateAddress](#) di AWS SDK for Ruby API Referensi.

CreateKeyPair

Contoh kode berikut menunjukkan cara menggunakan `CreateKeyPair`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
# This code example does the following:
# 1. Creates a key pair in Amazon Elastic Compute Cloud (Amazon EC2).
# 2. Displays information about available key pairs.
# 3. Deletes the key pair.

require "aws-sdk-ec2"

# @param ec2_client [Aws::EC2::Client] An initialized EC2 client.
# @param key_pair_name [String] The name for the key pair and private
#   key file.
# @return [Boolean] true if the key pair and private key file were
#   created; otherwise, false.
# @example
#   exit 1 unless key_pair_created?(
#     Aws::EC2::Client.new(region: 'us-west-2'),
#     'my-key-pair'
#   )
def key_pair_created?(ec2_client, key_pair_name)
  key_pair = ec2_client.create_key_pair(key_name: key_pair_name)
  puts "Created key pair '#{key_pair.key_name}' with fingerprint " \
    "'#{key_pair.key_fingerprint}' and ID '#{key_pair.key_pair_id}'."
  filename = File.join(Dir.home, key_pair_name + ".pem")
  File.open(filename, "w") { |file| file.write(key_pair.key_material) }
  puts "Private key file saved locally as '#{filename}'."
  return true
rescue Aws::EC2::Errors::InvalidKeyPairDuplicate
  puts "Error creating key pair: a key pair named '#{key_pair_name}' " \
```

```
    "already exists."
    return false
  rescue StandardError => e
    puts "Error creating key pair or saving private key file: #{e.message}"
    return false
  end

# Displays information about available key pairs in
# Amazon Elastic Compute Cloud (Amazon EC2).
#
# @param ec2_client [Aws::EC2::Client] An initialized EC2 client.
# @example
#   describe_key_pairs(Aws::EC2::Client.new(region: 'us-west-2'))
def describe_key_pairs(ec2_client)
  result = ec2_client.describe_key_pairs
  if result.key_pairs.count.zero?
    puts "No key pairs found."
  else
    puts "Key pair names:"
    result.key_pairs.each do |key_pair|
      puts key_pair.key_name
    end
  end
end

rescue StandardError => e
  puts "Error getting information about key pairs: #{e.message}"
end

# Deletes a key pair in Amazon Elastic Compute Cloud (Amazon EC2).
#
# Prerequisites:
#
# - The key pair to delete.
#
# @param ec2_client [Aws::EC2::Client] An initialized EC2 client.
# @param key_pair_name [String] The name of the key pair to delete.
# @return [Boolean] true if the key pair was deleted; otherwise, false.
# @example
#   exit 1 unless key_pair_deleted?(
#     Aws::EC2::Client.new(region: 'us-west-2'),
#     'my-key-pair'
#   )
def key_pair_deleted?(ec2_client, key_pair_name)
  ec2_client.delete_key_pair(key_name: key_pair_name)
  return true
end
```

```
rescue StandardError => e
  puts "Error deleting key pair: #{e.message}"
  return false
end

# Example usage:
def run_me
  key_pair_name = ""
  region = ""
  # Print usage information and then stop.
  if ARGV[0] == "--help" || ARGV[0] == "-h"
    puts "Usage:  ruby ec2-ruby-example-key-pairs.rb KEY_PAIR_NAME REGION"
    puts "Example: ruby ec2-ruby-example-key-pairs.rb my-key-pair us-west-2"
    exit 1
  # If no values are specified at the command prompt, use these default values.
  # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
  elsif ARGV.count.zero?
    key_pair_name = "my-key-pair"
    region = "us-west-2"
  # Otherwise, use the values as specified at the command prompt.
  else
    key_pair_name = ARGV[0]
    region = ARGV[1]
  end

  ec2_client = Aws::EC2::Client.new(region: region)

  puts "Displaying existing key pair names before creating this key pair..."
  describe_key_pairs(ec2_client)

  puts "-" * 10
  puts "Creating key pair..."
  unless key_pair_created?(ec2_client, key_pair_name)
    puts "Stopping program."
    exit 1
  end

  puts "-" * 10
  puts "Displaying existing key pair names after creating this key pair..."
  describe_key_pairs(ec2_client)

  puts "-" * 10
  puts "Deleting key pair..."
  unless key_pair_deleted?(ec2_client, key_pair_name)
```

```

    puts "Stopping program. You must delete the key pair yourself."
    exit 1
end
puts "Key pair deleted."

puts "-" * 10
puts "Now that the key pair is deleted, " \
     "also deleting the related private key pair file..."
filename = File.join(Dir.home, key_pair_name + ".pem")
File.delete(filename)
if File.exist?(filename)
  puts "Could not delete file at '#{filename}'. You must delete it yourself."
else
  puts "File deleted."
end

puts "-" * 10
puts "Displaying existing key pair names after deleting this key pair..."
describe_key_pairs(ec2_client)
end

run_me if $PROGRAM_NAME == __FILE__

```

- Untuk API detailnya, lihat [CreateKeyPair](#) di AWS SDK for Ruby API Referensi.

CreateRouteTable

Contoh kode berikut menunjukkan cara menggunakan `CreateRouteTable`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```

require "aws-sdk-ec2"

# Prerequisites:
#

```

```
# - A VPC in Amazon VPC.
# - A subnet in that VPC.
# - A gateway attached to that subnet.
#
# @param ec2_resource [Aws::EC2::Resource] An initialized
#   Amazon Elastic Compute Cloud (Amazon EC2) resource object.
# @param vpc_id [String] The ID of the VPC for the route table.
# @param subnet_id [String] The ID of the subnet for the route table.
# @param gateway_id [String] The ID of the gateway for the route.
# @param destination_cidr_block [String] The destination CIDR block
#   for the route.
# @param tag_key [String] The key portion of the tag for the route table.
# @param tag_value [String] The value portion of the tag for the route table.
# @return [Boolean] true if the route table was created and associated;
#   otherwise, false.
# @example
#   exit 1 unless route_table_created_and_associated?(
#     Aws::EC2::Resource.new(region: 'us-west-2'),
#     'vpc-0b6f769731EXAMPLE',
#     'subnet-03d9303b57EXAMPLE',
#     'igw-06ca90c011EXAMPLE',
#     '0.0.0.0/0',
#     'my-key',
#     'my-value'
#   )
def route_table_created_and_associated?(
  ec2_resource,
  vpc_id,
  subnet_id,
  gateway_id,
  destination_cidr_block,
  tag_key,
  tag_value
)
  route_table = ec2_resource.create_route_table(vpc_id: vpc_id)
  puts "Created route table with ID '#{route_table.id}'."
  route_table.create_tags(
    tags: [
      {
        key: tag_key,
        value: tag_value
      }
    ]
  )
)
```

```
puts "Added tags to route table."
route_table.create_route(
  destination_cidr_block: destination_cidr_block,
  gateway_id: gateway_id
)
puts "Created route with destination CIDR block " \
    "'#{destination_cidr_block}' and associated with gateway " \
    "with ID '#{gateway_id}'."
route_table.associate_with_subnet(subnet_id: subnet_id)
puts "Associated route table with subnet with ID '#{subnet_id}'."
return true
rescue StandardError => e
  puts "Error creating or associating route table: #{e.message}"
  puts "If the route table was created but not associated, you should " \
    "clean up by deleting the route table."
  return false
end

# Example usage:
def run_me
  vpc_id = ""
  subnet_id = ""
  gateway_id = ""
  destination_cidr_block = ""
  tag_key = ""
  tag_value = ""
  region = ""
  # Print usage information and then stop.
  if ARGV[0] == "--help" || ARGV[0] == "-h"
    puts "Usage: ruby ec2-ruby-example-create-route-table.rb " \
        "VPC_ID SUBNET_ID GATEWAY_ID DESTINATION_CIDR_BLOCK " \
        "TAG_KEY TAG_VALUE REGION"
  # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
  puts "Example: ruby ec2-ruby-example-create-route-table.rb " \
      "vpc-0b6f769731EXAMPLE subnet-03d9303b57EXAMPLE igw-06ca90c011EXAMPLE " \
      "'0.0.0.0/0' my-key my-value us-west-2"
  exit 1
  # If no values are specified at the command prompt, use these default values.
  elsif ARGV.count.zero?
    vpc_id = "vpc-0b6f769731EXAMPLE"
    subnet_id = "subnet-03d9303b57EXAMPLE"
    gateway_id = "igw-06ca90c011EXAMPLE"
    destination_cidr_block = "0.0.0.0/0"
    tag_key = "my-key"
```

```
    tag_value = "my-value"
    # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
    region = "us-west-2"
# Otherwise, use the values as specified at the command prompt.
else
  vpc_id = ARGV[0]
  subnet_id = ARGV[1]
  gateway_id = ARGV[2]
  destination_cidr_block = ARGV[3]
  tag_key = ARGV[4]
  tag_value = ARGV[5]
  region = ARGV[6]
end

ec2_resource = Aws::EC2::Resource.new(region: region)

if route_table_created_and_associated?(
  ec2_resource,
  vpc_id,
  subnet_id,
  gateway_id,
  destination_cidr_block,
  tag_key,
  tag_value
)
  puts "Route table created and associated."
else
  puts "Route table not created or not associated."
end
end


run_me if $PROGRAM_NAME == __FILE__
```

- Untuk API detailnya, lihat [CreateRouteTable](#) di AWS SDK for Ruby API Referensi.

CreateSecurityGroup

Contoh kode berikut menunjukkan cara menggunakan `CreateSecurityGroup`.

SDK untuk Ruby

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
# This code example does the following:
# 1. Creates an Amazon Elastic Compute Cloud (Amazon EC2) security group.
# 2. Adds inbound rules to the security group.
# 3. Displays information about available security groups.
# 4. Deletes the security group.

require "aws-sdk-ec2"

# Creates an Amazon Elastic Compute Cloud (Amazon EC2) security group.
#
# Prerequisites:
#
# - A VPC in Amazon Virtual Private Cloud (Amazon VPC).
#
# @param ec2_client [Aws::EC2::Client] An initialized
#   Amazon EC2 client.
# @param group_name [String] A name for the security group.
# @param description [String] A description for the security group.
# @param vpc_id [String] The ID of the VPC for the security group.
# @return [String] The ID of security group that was created.
# @example
#   puts create_security_group(
#     Aws::EC2::Client.new(region: 'us-west-2'),
#     'my-security-group',
#     'This is my security group.',
#     'vpc-6713dfEX'
#   )
def create_security_group(
  ec2_client,
  group_name,
  description,
  vpc_id
)
```

```
security_group = ec2_client.create_security_group(
  group_name: group_name,
  description: description,
  vpc_id: vpc_id
)
puts "Created security group '#{group_name}' with ID " \
    "'#{security_group.group_id}' in VPC with ID '#{vpc_id}'."
return security_group.group_id
rescue StandardError => e
  puts "Error creating security group: #{e.message}"
  return "Error"
end

# Adds an inbound rule to an Amazon Elastic Compute Cloud (Amazon EC2)
# security group.
#
# Prerequisites:
#
# - The security group.
#
# @param ec2_client [Aws::EC2::Client] An initialized Amazon EC2 client.
# @param security_group_id [String] The ID of the security group.
# @param ip_protocol [String] The network protocol for the inbound rule.
# @param from_port [String] The originating port for the inbound rule.
# @param to_port [String] The destination port for the inbound rule.
# @param cidr_ip_range [String] The CIDR IP range for the inbound rule.
# @return
# @example
#   exit 1 unless security_group_ingress_authorized?(
#     Aws::EC2::Client.new(region: 'us-west-2'),
#     'sg-030a858e078f1b9EX',
#     'tcp',
#     '80',
#     '80',
#     '0.0.0.0/0'
#   )
def security_group_ingress_authorized?(
  ec2_client,
  security_group_id,
  ip_protocol,
  from_port,
  to_port,
  cidr_ip_range
)
```

```

ec2_client.authorize_security_group_ingress(
  group_id: security_group_id,
  ip_permissions: [
    {
      ip_protocol: ip_protocol,
      from_port: from_port,
      to_port: to_port,
      ip_ranges: [
        {
          cidr_ip: cidr_ip_range
        }
      ]
    }
  ]
)
puts "Added inbound rule to security group '#{security_group_id}' for protocol " \
    "'#{ip_protocol}' from port '#{from_port}' to port '#{to_port}' " \
    "with CIDR IP range '#{cidr_ip_range}'."
return true
rescue StandardError => e
  puts "Error adding inbound rule to security group: #{e.message}"
  return false
end

# Displays information about a security group's IP permissions set in
# Amazon Elastic Compute Cloud (Amazon EC2).
#
# Prerequisites:
#
# - A security group with inbound rules, outbound rules, or both.
#
# @param p [Aws::EC2::Types::IpPermission] The IP permissions set.
# @example
#   ec2_client = Aws::EC2::Client.new(region: 'us-west-2')
#   response = ec2_client.describe_security_groups
#   unless sg.ip_permissions.empty?
#     describe_security_group_permissions(
#       response.security_groups[0].ip_permissions[0]
#     )
#   end
def describe_security_group_permissions(perm)
  print " Protocol: #{perm.ip_protocol == '-1' ? 'All' : perm.ip_protocol}"

  unless perm.from_port.nil?

```

```
    if perm.from_port == "-1" || perm.from_port == -1
      print ", From: All"
    else
      print ", From: #{perm.from_port}"
    end
  end

  unless perm.to_port.nil?
    if perm.to_port == "-1" || perm.to_port == -1
      print ", To: All"
    else
      print ", To: #{perm.to_port}"
    end
  end

  if perm.key?(:ipv_6_ranges) && perm.ipv_6_ranges.count.positive?
    print ", CIDR IPv6: #{perm.ipv_6_ranges[0].cidr_ipv_6}"
  end

  if perm.key?(:ip_ranges) && perm.ip_ranges.count.positive?
    print ", CIDR IPv4: #{perm.ip_ranges[0].cidr_ip}"
  end

  print "\n"
end

# Displays information about available security groups in
# Amazon Elastic Compute Cloud (Amazon EC2).
#
# @param ec2_client [Aws::EC2::Client] An initialized Amazon EC2 client.
# @example
#   describe_security_groups(Aws::EC2::Client.new(region: 'us-west-2'))
def describe_security_groups(ec2_client)
  response = ec2_client.describe_security_groups

  if response.security_groups.count.positive?
    response.security_groups.each do |sg|
      puts "-" * (sg.group_name.length + 13)
      puts "Name:          #{sg.group_name}"
      puts "Description:    #{sg.description}"
      puts "Group ID:       #{sg.group_id}"
      puts "Owner ID:       #{sg.owner_id}"
      puts "VPC ID:         #{sg.vpc_id}"
    end
  end
end
```

```
    if sg.tags.count.positive?
      puts "Tags:"
      sg.tags.each do |tag|
        puts "  Key: #{tag.key}, Value: #{tag.value}"
      end
    end

    unless sg.ip_permissions.empty?
      puts "Inbound rules:" if sg.ip_permissions.count.positive?
      sg.ip_permissions.each do |p|
        describe_security_group_permissions(p)
      end
    end

    unless sg.ip_permissions_egress.empty?
      puts "Outbound rules:" if sg.ip_permissions.count.positive?
      sg.ip_permissions_egress.each do |p|
        describe_security_group_permissions(p)
      end
    end
  end
else
  puts "No security groups found."
end
rescue StandardError => e
  puts "Error getting information about security groups: #{e.message}"
end

# Deletes an Amazon Elastic Compute Cloud (Amazon EC2)
# security group.
#
# Prerequisites:
#
# - The security group.
#
# @param ec2_client [Aws::EC2::Client] An initialized
#   Amazon EC2 client.
# @param security_group_id [String] The ID of the security group to delete.
# @return [Boolean] true if the security group was deleted; otherwise, false.
# @example
#   exit 1 unless security_group_deleted?(
#     Aws::EC2::Client.new(region: 'us-west-2'),
#     'sg-030a858e078f1b9EX'
#   )
```

```
def security_group_deleted?(ec2_client, security_group_id)
  ec2_client.delete_security_group(group_id: security_group_id)
  puts "Deleted security group '#{security_group_id}'."
  return true
rescue StandardError => e
  puts "Error deleting security group: #{e.message}"
  return false
end

# Example usage:
def run_me
  group_name = ""
  description = ""
  vpc_id = ""
  ip_protocol_http = ""
  from_port_http = ""
  to_port_http = ""
  cidr_ip_range_http = ""
  ip_protocol_ssh = ""
  from_port_ssh = ""
  to_port_ssh = ""
  cidr_ip_range_ssh = ""
  region = ""
  # Print usage information and then stop.
  if ARGV[0] == "--help" || ARGV[0] == "-h"
    puts "Usage:  ruby ec2-ruby-example-security-group.rb " \
      "GROUP_NAME DESCRIPTION VPC_ID IP_PROTOCOL_1 FROM_PORT_1 TO_PORT_1 " \
      "CIDR_IP_RANGE_1 IP_PROTOCOL_2 FROM_PORT_2 TO_PORT_2 " \
      "CIDR_IP_RANGE_2 REGION"
    puts "Example: ruby ec2-ruby-example-security-group.rb " \
      "my-security-group 'This is my security group.' vpc-6713dfEX " \
      "tcp 80 80 '0.0.0.0/0' tcp 22 22 '0.0.0.0/0' us-west-2"
    exit 1
  # If no values are specified at the command prompt, use these default values.
  elsif ARGV.count.zero?
    group_name = "my-security-group"
    description = "This is my security group."
    vpc_id = "vpc-6713dfEX"
    ip_protocol_http = "tcp"
    from_port_http = "80"
    to_port_http = "80"
    cidr_ip_range_http = "0.0.0.0/0"
    ip_protocol_ssh = "tcp"
    from_port_ssh = "22"
```

```
to_port_ssh = "22"
cidr_ip_range_ssh = "0.0.0.0/0"
# Replace us-west-2 with the AWS Region you're using for Amazon EC2.
region = "us-west-2"
# Otherwise, use the values as specified at the command prompt.
else
  group_name = ARGV[0]
  description = ARGV[1]
  vpc_id = ARGV[2]
  ip_protocol_http = ARGV[3]
  from_port_http = ARGV[4]
  to_port_http = ARGV[5]
  cidr_ip_range_http = ARGV[6]
  ip_protocol_ssh = ARGV[7]
  from_port_ssh = ARGV[8]
  to_port_ssh = ARGV[9]
  cidr_ip_range_ssh = ARGV[10]
  region = ARGV[11]
end

security_group_id = ""
security_group_exists = false
ec2_client = Aws::EC2::Client.new(region: region)

puts "Attempting to create security group..."
security_group_id = create_security_group(
  ec2_client,
  group_name,
  description,
  vpc_id
)
if security_group_id == "Error"
  puts "Could not create security group. Skipping this step."
else
  security_group_exists = true
end

if security_group_exists
  puts "Attempting to add inbound rules to security group..."
  unless security_group_ingress_authorized?(
    ec2_client,
    security_group_id,
    ip_protocol_http,
    from_port_http,
```

```
    to_port_http,
    cidr_ip_range_http
  )
  puts "Could not add inbound HTTP rule to security group. " \
    "Skipping this step."
end

unless security_group_ingress_authorized?(
  ec2_client,
  security_group_id,
  ip_protocol_ssh,
  from_port_ssh,
  to_port_ssh,
  cidr_ip_range_ssh
)
  puts "Could not add inbound SSH rule to security group. " \
    "Skipping this step."
end
end

puts "\nInformation about available security groups:"
describe_security_groups(ec2_client)

if security_group_exists
  puts "\nAttempting to delete security group..."
  unless security_group_deleted?(ec2_client, security_group_id)
    puts "Could not delete security group. You must delete it yourself."
  end
end
end


run_me if $PROGRAM_NAME == __FILE__
```

- Untuk API detailnya, lihat [CreateSecurityGroup](#) di AWS SDK for Ruby API Referensi.

CreateSubnet

Contoh kode berikut menunjukkan cara menggunakan `CreateSubnet`.

SDK untuk Ruby

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
require "aws-sdk-ec2"

# Creates a subnet within a virtual private cloud (VPC) in
# Amazon Virtual Private Cloud (Amazon VPC) and then tags
# the subnet.
#
# Prerequisites:
#
# - A VPC in Amazon VPC.
#
# @param ec2_resource [Aws::EC2::Resource] An initialized
#   Amazon Elastic Compute Cloud (Amazon EC2) resource object.
# @param vpc_id [String] The ID of the VPC for the subnet.
# @param cidr_block [String] The IPv4 CIDR block for the subnet.
# @param availability_zone [String] The ID of the Availability Zone
#   for the subnet.
# @param tag_key [String] The key portion of the tag for the subnet.
# @param tag_value [String] The value portion of the tag for the subnet.
# @return [Boolean] true if the subnet was created and tagged;
#   otherwise, false.
# @example
#   exit 1 unless subnet_created_and_tagged?(
#     Aws::EC2::Resource.new(region: 'us-west-2'),
#     'vpc-6713dfEX',
#     '10.0.0.0/24',
#     'us-west-2a',
#     'my-key',
#     'my-value'
#   )
def subnet_created_and_tagged?(
  ec2_resource,
  vpc_id,
  cidr_block,
```

```
availability_zone,
tag_key,
tag_value
)
subnet = ec2_resource.create_subnet(
  vpc_id: vpc_id,
  cidr_block: cidr_block,
  availability_zone: availability_zone
)
subnet.create_tags(
  tags: [
    {
      key: tag_key,
      value: tag_value
    }
  ]
)
puts "Subnet created with ID '#{subnet.id}' in VPC with ID '#{vpc_id}' " \
    "and CIDR block '#{cidr_block}' in availability zone " \
    "'#{availability_zone}' and tagged with key '#{tag_key}' and " \
    "value '#{tag_value}'."
return true
rescue StandardError => e
  puts "Error creating or tagging subnet: #{e.message}"
  return false
end

# Example usage:
def run_me
  vpc_id = ""
  cidr_block = ""
  availability_zone = ""
  tag_key = ""
  tag_value = ""
  region = ""
  # Print usage information and then stop.
  if ARGV[0] == "--help" || ARGV[0] == "-h"
    puts "Usage:  ruby ec2-ruby-example-create-subnet.rb " \
        "VPC_ID CIDR_BLOCK AVAILABILITY_ZONE TAG_KEY TAG_VALUE REGION"
    # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
    puts "Example: ruby ec2-ruby-example-create-subnet.rb " \
        "vpc-6713dfEX 10.0.0.0/24 us-west-2a my-key my-value us-west-2"
    exit 1
  # If no values are specified at the command prompt, use these default values.
```

```
elsif ARGV.count.zero?
  vpc_id = "vpc-6713dfEX"
  cidr_block = "10.0.0.0/24"
  availability_zone = "us-west-2a"
  tag_key = "my-key"
  tag_value = "my-value"
  # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
  region = "us-west-2"
# Otherwise, use the values as specified at the command prompt.
else
  vpc_id = ARGV[0]
  cidr_block = ARGV[1]
  availability_zone = ARGV[2]
  tag_key = ARGV[3]
  tag_value = ARGV[4]
  region = ARGV[5]
end

ec2_resource = Aws::EC2::Resource.new(region: region)

if subnet_created_and_tagged?(
  ec2_resource,
  vpc_id,
  cidr_block,
  availability_zone,
  tag_key,
  tag_value
)
  puts "Subnet created and tagged."
else
  puts "Subnet not created or not tagged."
end
end

run_me if $PROGRAM_NAME == __FILE__
```

- Untuk API detailnya, lihat [CreateSubnet](#) di AWS SDK for Ruby API Referensi.

CreateVpc

Contoh kode berikut menunjukkan cara menggunakan `CreateVpc`.

SDK untuk Ruby

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
require "aws-sdk-ec2"

# Creates a virtual private cloud (VPC) in
# Amazon Virtual Private Cloud (Amazon VPC) and then tags
# the VPC.
#
# @param ec2_resource [Aws::EC2::Resource] An initialized
#   Amazon Elastic Compute Cloud (Amazon EC2) resource object.
# @param cidr_block [String] The IPv4 CIDR block for the subnet.
# @param tag_key [String] The key portion of the tag for the VPC.
# @param tag_value [String] The value portion of the tag for the VPC.
# @return [Boolean] true if the VPC was created and tagged;
#   otherwise, false.
# @example
#   exit 1 unless vpc_created_and_tagged?(
#     Aws::EC2::Resource.new(region: 'us-west-2'),
#     '10.0.0.0/24',
#     'my-key',
#     'my-value'
#   )
def vpc_created_and_tagged?(
  ec2_resource,
  cidr_block,
  tag_key,
  tag_value
)
  vpc = ec2_resource.create_vpc(cidr_block: cidr_block)

  # Create a public DNS by enabling DNS support and DNS hostnames.
  vpc.modify_attribute(enable_dns_support: { value: true })
  vpc.modify_attribute(enable_dns_hostnames: { value: true })

  vpc.create_tags(tags: [{ key: tag_key, value: tag_value }])
end
```

```
puts "Created VPC with ID '#{vpc.id}' and tagged with key " \
    "'#{tag_key}' and value '#{tag_value}'."
return true
rescue StandardError => e
  puts "#{e.message}"
  return false
end

# Example usage:
def run_me
  cidr_block = ""
  tag_key = ""
  tag_value = ""
  region = ""
  # Print usage information and then stop.
  if ARGV[0] == "--help" || ARGV[0] == "-h"
    puts "Usage:  ruby ec2-ruby-example-create-vpc.rb " \
        "CIDR_BLOCK TAG_KEY TAG_VALUE REGION"
    # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
    puts "Example: ruby ec2-ruby-example-create-vpc.rb " \
        "10.0.0.0/24 my-key my-value us-west-2"
    exit 1
  # If no values are specified at the command prompt, use these default values.
  elsif ARGV.count.zero?
    cidr_block = "10.0.0.0/24"
    tag_key = "my-key"
    tag_value = "my-value"
    # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
    region = "us-west-2"
  # Otherwise, use the values as specified at the command prompt.
  else
    cidr_block = ARGV[0]
    tag_key = ARGV[1]
    tag_value = ARGV[2]
    region = ARGV[3]
  end

  ec2_resource = Aws::EC2::Resource.new(region: region)

  if vpc_created_and_tagged?(
    ec2_resource,
    cidr_block,
    tag_key,
```

```
    tag_value
  )
  puts "VPC created and tagged."
else
  puts "VPC not created or not tagged."
end
end
end

run_me if $PROGRAM_NAME == __FILE__
```

- Untuk API detailnya, lihat [CreateVpc](#) di AWS SDK for Ruby API Referensi.

DescribeInstances

Contoh kode berikut menunjukkan cara menggunakan `DescribeInstances`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
require "aws-sdk-ec2"

# @param ec2_resource [Aws::EC2::Resource] An initialized EC2 resource object.
# @example
# list_instance_ids_states(Aws::EC2::Resource.new(region: 'us-west-2'))
def list_instance_ids_states(ec2_resource)
  response = ec2_resource.instances
  if response.count.zero?
    puts "No instances found."
  else
    puts "Instances -- ID, state:"
    response.each do |instance|
      puts "#{instance.id}, #{instance.state.name}"
    end
  end
end

rescue StandardError => e
```

```
puts "Error getting information about instances: #{e.message}"
end

# Example usage:
def run_me
  region = ""
  # Print usage information and then stop.
  if ARGV[0] == "--help" || ARGV[0] == "-h"
    puts "Usage: ruby ec2-ruby-example-get-all-instance-info.rb REGION"
    # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
    puts "Example: ruby ec2-ruby-example-get-all-instance-info.rb us-west-2"
    exit 1
  # If no values are specified at the command prompt, use these default values.
  # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
  elsif ARGV.count.zero?
    region = "us-west-2"
  # Otherwise, use the values as specified at the command prompt.
  else
    region = ARGV[0]
  end
  ec2_resource = Aws::EC2::Resource.new(region: region)
  list_instance_ids_states(ec2_resource)
end

run_me if $PROGRAM_NAME == __FILE__
```

- Untuk API detailnya, lihat [DescribeInstances](#) di AWS SDK for Ruby API Referensi.

DescribeRegions

Contoh kode berikut menunjukkan cara menggunakan `DescribeRegions`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
require "aws-sdk-ec2"

# @param ec2_client [Aws::EC2::Client] An initialized EC2 client.
# @example
#   list_regions_endpoints(Aws::EC2::Client.new(region: 'us-west-2'))
def list_regions_endpoints(ec2_client)
  result = ec2_client.describe_regions
  # Enable pretty printing.
  max_region_string_length = 16
  max_endpoint_string_length = 33
  # Print header.
  print "Region"
  print " " * (max_region_string_length - "Region".length)
  print "  Endpoint\n"
  print "-" * max_region_string_length
  print " "
  print "-" * max_endpoint_string_length
  print "\n"
  # Print Regions and their endpoints.
  result.regions.each do |region|
    print region.region_name
    print " " * (max_region_string_length - region.region_name.length)
    print " "
    print region.endpoint
    print "\n"
  end
end

# Displays a list of Amazon Elastic Compute Cloud (Amazon EC2)
# Availability Zones available to you depending on the AWS Region
# of the Amazon EC2 client.
#
# @param ec2_client [Aws::EC2::Client] An initialized EC2 client.
# @example
#   list_availability_zones(Aws::EC2::Client.new(region: 'us-west-2'))
def list_availability_zones(ec2_client)
  result = ec2_client.describe_availability_zones
  # Enable pretty printing.
  max_region_string_length = 16
  max_zone_string_length = 18
  max_state_string_length = 9
  # Print header.
  print "Region"
  print " " * (max_region_string_length - "Region".length)
```



```
print " Zone"
print " " * (max_zone_string_length - "Zone".length)
print " State\n"
print "-" * max_region_string_length
print " "
print "-" * max_zone_string_length
print " "
print "-" * max_state_string_length
print "\n"
# Print Regions, Availability Zones, and their states.
result.availability_zones.each do |zone|
  print zone.region_name
  print " " * (max_region_string_length - zone.region_name.length)
  print " "
  print zone.zone_name
  print " " * (max_zone_string_length - zone.zone_name.length)
  print " "
  print zone.state
  # Print any messages for this Availability Zone.
  if zone.messages.count.positive?
    print "\n"
    puts " Messages for this zone:"
    zone.messages.each do |message|
      print "   #{message.message}\n"
    end
  end
  print "\n"
end
end

# Example usage:
def run_me
  region = ""
  # Print usage information and then stop.
  if ARGV[0] == "--help" || ARGV[0] == "-h"
    puts "Usage: ruby ec2-ruby-example-regions-availability-zones.rb REGION"
    # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
    puts "Example: ruby ec2-ruby-example-regions-availability-zones.rb us-west-2"
    exit 1
  # If no values are specified at the command prompt, use these default values.
  # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
  elsif ARGV.count.zero?
    region = "us-west-2"
  # Otherwise, use the values as specified at the command prompt.
```

```

else
  region = ARGV[0]
end

ec2_client = Aws::EC2::Client.new(region: region)

puts "AWS Regions for Amazon EC2 that are available to you:"
list_regions_endpoints(ec2_client)
puts "\n\nAmazon EC2 Availability Zones that are available to you for AWS Region
'#{region}':"
list_availability_zones(ec2_client)
end

run_me if $PROGRAM_NAME == __FILE__

```

- Untuk API detailnya, lihat [DescribeRegions](#) di AWS SDK for Ruby API Referensi.

ReleaseAddress

Contoh kode berikut menunjukkan cara menggunakan `ReleaseAddress`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```

# Releases an Elastic IP address from an
# Amazon Elastic Compute Cloud (Amazon EC2) instance.
#
# Prerequisites:
#
# - An Amazon EC2 instance with an associated Elastic IP address.
#
# @param ec2_client [Aws::EC2::Client] An initialized EC2 client.
# @param allocation_id [String] The ID of the allocation corresponding to
# the Elastic IP address.
# @return [Boolean] true if the Elastic IP address was released;
# otherwise, false.

```

```
# @example
#   exit 1 unless elastic_ip_address_released?(
#     Aws::EC2::Client.new(region: 'us-west-2'),
#     'eipalloc-04452e528a66279EX'
#   )
def elastic_ip_address_released?(ec2_client, allocation_id)
  ec2_client.release_address(allocation_id: allocation_id)
  return true
rescue StandardError => e
  puts("Error releasing Elastic IP address: #{e.message}")
  return false
end
```

- Untuk API detailnya, lihat [ReleaseAddress](#) di AWS SDK for Ruby API Referensi.

StartInstances

Contoh kode berikut menunjukkan cara menggunakan `StartInstances`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
require "aws-sdk-ec2"

# Attempts to start an Amazon Elastic Compute Cloud (Amazon EC2) instance.
#
# Prerequisites:
#
# - The Amazon EC2 instance.
#
# @param ec2_client [Aws::EC2::Client] An initialized EC2 client.
# @param instance_id [String] The ID of the instance.
# @return [Boolean] true if the instance was started; otherwise, false.
# @example
#   exit 1 unless instance_started?(
```

```
# Aws::EC2::Client.new(region: 'us-west-2'),
# 'i-123abc'
# )
def instance_started?(ec2_client, instance_id)
  response = ec2_client.describe_instance_status(instance_ids: [instance_id])

  if response.instance_statuses.count.positive?
    state = response.instance_statuses[0].instance_state.name
    case state
    when "pending"
      puts "Error starting instance: the instance is pending. Try again later."
      return false
    when "running"
      puts "The instance is already running."
      return true
    when "terminated"
      puts "Error starting instance: " \
        "the instance is terminated, so you cannot start it."
      return false
    end
  end
end

ec2_client.start_instances(instance_ids: [instance_id])
ec2_client.wait_until(:instance_running, instance_ids: [instance_id])
puts "Instance started."
return true
rescue StandardError => e
  puts "Error starting instance: #{e.message}"
  return false
end

# Example usage:
def run_me
  instance_id = ""
  region = ""
  # Print usage information and then stop.
  if ARGV[0] == "--help" || ARGV[0] == "-h"
    puts "Usage:  ruby ec2-ruby-example-start-instance-i-123abc.rb " \
      "INSTANCE_ID REGION "
  # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
  puts "Example: ruby ec2-ruby-example-start-instance-i-123abc.rb " \
    "i-123abc us-west-2"
  exit 1
  # If no values are specified at the command prompt, use these default values.
```

```
# Replace us-west-2 with the AWS Region you're using for Amazon EC2.
elsif ARGV.count.zero?
  instance_id = "i-123abc"
  region = "us-west-2"
# Otherwise, use the values as specified at the command prompt.
else
  instance_id = ARGV[0]
  region = ARGV[1]
end

ec2_client = Aws::EC2::Client.new(region: region)

puts "Attempting to start instance '#{instance_id}' " \
      "(this might take a few minutes)..."
unless instance_started?(ec2_client, instance_id)
  puts "Could not start instance."
end
end

run_me if $PROGRAM_NAME == __FILE__
```

- Untuk API detailnya, lihat [StartInstances](#) di AWS SDK for Ruby API Referensi.

StopInstances

Contoh kode berikut menunjukkan cara menggunakan `StopInstances`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
require "aws-sdk-ec2"

# Prerequisites:
#
# - The Amazon EC2 instance.
```

```
#
# @param ec2_client [Aws::EC2::Client] An initialized EC2 client.
# @param instance_id [String] The ID of the instance.
# @return [Boolean] true if the instance was stopped; otherwise, false.
# @example
#   exit 1 unless instance_stopped?(
#     Aws::EC2::Client.new(region: 'us-west-2'),
#     'i-123abc'
#   )
def instance_stopped?(ec2_client, instance_id)
  response = ec2_client.describe_instance_status(instance_ids: [instance_id])

  if response.instance_statuses.count.positive?
    state = response.instance_statuses[0].instance_state.name
    case state
    when "stopping"
      puts "The instance is already stopping."
      return true
    when "stopped"
      puts "The instance is already stopped."
      return true
    when "terminated"
      puts "Error stopping instance: " \
        "the instance is terminated, so you cannot stop it."
      return false
    end
  end
end

ec2_client.stop_instances(instance_ids: [instance_id])
ec2_client.wait_until(:instance_stopped, instance_ids: [instance_id])
puts "Instance stopped."
return true
rescue StandardError => e
  puts "Error stopping instance: #{e.message}"
  return false
end

# Example usage:
def run_me
  instance_id = ""
  region = ""
  # Print usage information and then stop.
  if ARGV[0] == "--help" || ARGV[0] == "-h"
    puts "Usage:  ruby ec2-ruby-example-stop-instance-i-123abc.rb " \
```

```
"INSTANCE_ID REGION "  
# Replace us-west-2 with the AWS Region you're using for Amazon EC2.  
puts "Example: ruby ec2-ruby-example-start-instance-i-123abc.rb " \  
  "i-123abc us-west-2"  
exit 1  
# If no values are specified at the command prompt, use these default values.  
# Replace us-west-2 with the AWS Region you're using for Amazon EC2.  
elsif ARGV.count.zero?  
  instance_id = "i-123abc"  
  region = "us-west-2"  
# Otherwise, use the values as specified at the command prompt.  
else  
  instance_id = ARGV[0]  
  region = ARGV[1]  
end  
  
ec2_client = Aws::EC2::Client.new(region: region)  
  
puts "Attempting to stop instance '#{instance_id}' " \  
  "(this might take a few minutes)..."  
unless instance_stopped?(ec2_client, instance_id)  
  puts "Could not stop instance."  
end  
end  
  
run_me if $PROGRAM_NAME == __FILE__
```

- Untuk API detailnya, lihat [StopInstances](#) di AWS SDK for Ruby API Referensi.

TerminateInstances

Contoh kode berikut menunjukkan cara menggunakan `TerminateInstances`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
require "aws-sdk-ec2"

# Prerequisites:
#
# - The Amazon EC2 instance.
#
# @param ec2_client [Aws::EC2::Client] An initialized EC2 client.
# @param instance_id [String] The ID of the instance.
# @return [Boolean] true if the instance was terminated; otherwise, false.
# @example
#   exit 1 unless instance_terminated?(
#     Aws::EC2::Client.new(region: 'us-west-2'),
#     'i-123abc'
#   )
def instance_terminated?(ec2_client, instance_id)
  response = ec2_client.describe_instance_status(instance_ids: [instance_id])

  if response.instance_statuses.count.positive? &&
    response.instance_statuses[0].instance_state.name == "terminated"

    puts "The instance is already terminated."
    return true
  end

  ec2_client.terminate_instances(instance_ids: [instance_id])
  ec2_client.wait_until(:instance_terminated, instance_ids: [instance_id])
  puts "Instance terminated."
  return true
rescue StandardError => e
  puts "Error terminating instance: #{e.message}"
  return false
end

# Example usage:
def run_me
  instance_id = ""
  region = ""
  # Print usage information and then stop.
  if ARGV[0] == "--help" || ARGV[0] == "-h"
    puts "Usage:  ruby ec2-ruby-example-terminate-instance-i-123abc.rb " \
      "INSTANCE_ID REGION "
  # Replace us-west-2 with the AWS Region you're using for Amazon EC2.
  end
end
```



```
puts "Example: ruby ec2-ruby-example-terminate-instance-i-123abc.rb " \
    "i-123abc us-west-2"
exit 1
# If no values are specified at the command prompt, use these default values.
# Replace us-west-2 with the AWS Region you're using for Amazon EC2.
elsif ARGV.count.zero?
  instance_id = "i-123abc"
  region = "us-west-2"
# Otherwise, use the values as specified at the command prompt.
else
  instance_id = ARGV[0]
  region = ARGV[1]
end

ec2_client = Aws::EC2::Client.new(region: region)

puts "Attempting to terminate instance '#{instance_id}' " \
    "(this might take a few minutes)..."
unless instance_terminated?(ec2_client, instance_id)
  puts "Could not terminate instance."
end
end

run_me if $PROGRAM_NAME == __FILE__
```

- Untuk API detailnya, lihat [TerminateInstances](#) di AWS SDK for Ruby API Referensi.

Contoh Elastic SDK Beanstalk yang digunakan untuk Ruby

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan AWS SDK for Ruby with Elastic Beanstalk.

Tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

DescribeApplications

Contoh kode berikut menunjukkan cara menggunakan `DescribeApplications`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
# Class to manage Elastic Beanstalk applications
class ElasticBeanstalkManager
  def initialize(eb_client, logger: Logger.new($stdout))
    @eb_client = eb_client
    @logger = logger
  end

  # Lists applications and their environments
  def list_applications
    @eb_client.describe_applications.applications.each do |application|
      log_application_details(application)
      list_environments(application.application_name)
    end
  rescue Aws::ElasticBeanstalk::Errors::ServiceError => e
    @logger.error("Elastic Beanstalk Service Error: #{e.message}")
  end

  private

  # Logs application details
  def log_application_details(application)
    @logger.info("Name:          #{application.application_name}")
    @logger.info("Description: #{application.description}")
  end

  # Lists and logs details of environments for a given application
```

```

def list_environments(application_name)
  @eb_client.describe_environments(application_name:
application_name).environments.each do |env|
    @logger.info("  Environment:  #{env.environment_name}")
    @logger.info("    URL:          #{env.cname}")
    @logger.info("    Health:         #{env.health}")
  end
  rescue Aws::ElasticBeanstalk::Errors::ServiceError => e
    @logger.error("Error listing environments for application #{application_name}:
#{e.message}")
  end
end
end

```

- Untuk API detailnya, lihat [DescribeApplications](#) di AWS SDK for Ruby API Referensi.

List Available Solution Stacks

Contoh kode berikut menunjukkan cara menggunakan `ListAvailableSolutionStacks`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```

# Manages listing of AWS Elastic Beanstalk solution stacks
# @param [Aws::ElasticBeanstalk::Client] eb_client
# @param [String] filter - Returns subset of results based on match
# @param [Logger] logger
class StackLister
  # Initialize with AWS Elastic Beanstalk client
  def initialize(eb_client, filter, logger: Logger.new($stdout))
    @eb_client = eb_client
    @filter = filter.downcase
    @logger = logger
  end

  # Lists and logs Elastic Beanstalk solution stacks
  def list_stacks

```

```
stacks = @eb_client.list_available_solution_stacks.solution_stacks
orig_length = stacks.length
filtered_length = 0

stacks.each do |stack|
  if @filter.empty? || stack.downcase.include?(@filter)
    @logger.info(stack)
    filtered_length += 1
  end
end

log_summary(filtered_length, orig_length)
rescue Aws::Errors::ServiceError => e
  @logger.error("Error listing solution stacks: #{e.message}")
end

private

# Logs summary of listed stacks
def log_summary(filtered_length, orig_length)
  if @filter.empty?
    @logger.info("Showed #{orig_length} stack(s)")
  else
    @logger.info("Showed #{filtered_length} stack(s) of #{orig_length}")
  end
end
end
```

- Untuk API detailnya, lihat [ListAvailableSolutionStacks](#) di AWS SDK for Ruby API Referensi.

UpdateApplication

Contoh kode berikut menunjukkan cara menggunakan `UpdateApplication`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
# Manages deployment of Rails applications to AWS Elastic Beanstalk
class RailsAppDeployer
  def initialize(eb_client, s3_client, app_name, logger: Logger.new($stdout))
    @eb_client = eb_client
    @s3_client = s3_client
    @app_name = app_name
    @logger = logger
  end

  # Deploys the latest application version to Elastic Beanstalk
  def deploy
    create_storage_location
    zip_file_name = create_zip_file
    upload_zip_to_s3(zip_file_name)
    create_and_deploy_new_application_version(zip_file_name)
  end

  private

  # Creates a new S3 storage location for the application
  def create_storage_location
    resp = @eb_client.create_storage_location
    @logger.info("Created storage location in bucket #{resp.s3_bucket}")
  rescue Aws::Errors::ServiceError => e
    @logger.error("Failed to create storage location: #{e.message}")
  end

  # Creates a ZIP file of the application using git
  def create_zip_file
    zip_file_basename = SecureRandom.urlsafe_base64
    zip_file_name = "#{zip_file_basename}.zip"
    `git archive --format=zip -o #{zip_file_name} HEAD`
    zip_file_name
  end

  # Uploads the ZIP file to the S3 bucket
  def upload_zip_to_s3(zip_file_name)
    zip_contents = File.read(zip_file_name)
    key = "#{@app_name}/#{zip_file_name}"
    @s3_client.put_object(body: zip_contents, bucket: fetch_bucket_name, key: key)
  rescue Aws::Errors::ServiceError => e
    @logger.error("Failed to upload ZIP file to S3: #{e.message}")
  end
end
```

```
# Fetches the S3 bucket name from Elastic Beanstalk application versions
def fetch_bucket_name
  app_versions = @eb_client.describe_application_versions(application_name:
@app_name)
  av = app_versions.application_versions.first
  av.source_bundle.s3_bucket
rescue Aws::Errors::ServiceError => e
  @logger.error("Failed to fetch bucket name: #{e.message}")
  raise
end

# Creates a new application version and deploys it
def create_and_deploy_new_application_version(zip_file_name)
  version_label = File.basename(zip_file_name, ".zip")
  @eb_client.create_application_version(
    process: false,
    application_name: @app_name,
    version_label: version_label,
    source_bundle: {
      s3_bucket: fetch_bucket_name,
      s3_key: "#{@app_name}/#{zip_file_name}"
    },
    description: "Updated #{Time.now.strftime('%d/%m/%Y')}}"
  )
  update_environment(version_label)
rescue Aws::Errors::ServiceError => e
  @logger.error("Failed to create or deploy application version: #{e.message}")
end

# Updates the environment to the new application version
def update_environment(version_label)
  env_name = fetch_environment_name
  @eb_client.update_environment(
    environment_name: env_name,
    version_label: version_label
  )
rescue Aws::Errors::ServiceError => e
  @logger.error("Failed to update environment: #{e.message}")
end

# Fetches the environment name of the application
def fetch_environment_name
  envs = @eb_client.describe_environments(application_name: @app_name)
```

```
envs.environments.first.environment_name
rescue Aws::Errors::ServiceError => e
  @logger.error("Failed to fetch environment name: #{e.message}")
  raise
end
end
```

- Untuk API detailnya, lihat [UpdateApplication](#) di AWS SDK for Ruby API Referensi.

EventBridge contoh menggunakan SDK untuk Ruby

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan AWS SDK for Ruby with EventBridge.

Skenario adalah contoh kode yang menunjukkan kepada Anda bagaimana menyelesaikan tugas tertentu dengan memanggil beberapa fungsi dalam layanan atau dikombinasikan dengan yang lain AWS layanan.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Skenario](#)

Skenario

Buat dan picu aturan

Contoh kode berikut menunjukkan cara membuat dan memicu aturan di Amazon EventBridge.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

Panggil fungsi dalam urutan yang benar.

```
require "aws-sdk-sns"
require "aws-sdk-iam"
require "aws-sdk-cloudwatchevents"
require "aws-sdk-ec2"
require "aws-sdk-cloudwatch"
require "aws-sdk-cloudwatchlogs"
require "securerandom"
```

Memeriksa apakah topik Amazon Simple Notification Service (AmazonSNS) yang ditentukan ada di antara yang disediakan untuk fungsi ini.

```
# Checks whether the specified Amazon SNS
# topic exists among those provided to this function.
# This is a helper function that is called by the topic_exists? function.
#
# @param topics [Array] An array of Aws::SNS::Types::Topic objects.
# @param topic_arn [String] The ARN of the topic to find.
# @return [Boolean] true if the topic ARN was found; otherwise, false.
# @example
#   sns_client = Aws::SNS::Client.new(region: 'us-east-1')
#   response = sns_client.list_topics
#   if topic_found?(
#     response.topics,
#     'arn:aws:sns:us-east-1:111111111111:aws-doc-sdk-examples-topic'
#   )
#     puts 'Topic found.'
#   end

def topic_found?(topics, topic_arn)
  topics.each do |topic|
    return true if topic.topic_arn == topic_arn
  end
  return false
end
```

Memeriksa apakah topik yang ditentukan ada di antara yang tersedia untuk penelepon di AmazonSNS.

```
# Checks whether the specified topic exists among those available to the
# caller in Amazon SNS.
```



```

#
# @param sns_client [Aws::SNS::Client] An initialized Amazon SNS client.
# @param topic_arn [String] The ARN of the topic to find.
# @return [Boolean] true if the topic ARN was found; otherwise, false.
# @example
#   exit 1 unless topic_exists?(
#     Aws::SNS::Client.new(region: 'us-east-1'),
#     'arn:aws:sns:us-east-1:111111111111:aws-doc-sdk-examples-topic'
#   )
def topic_exists?(sns_client, topic_arn)
  puts "Searching for topic with ARN '#{topic_arn}'..."
  response = sns_client.list_topics
  if response.topics.count.positive?
    if topic_found?(response.topics, topic_arn)
      puts "Topic found."
      return true
    end
  end
  while response.next_page? do
    response = response.next_page
    if response.topics.count.positive?
      if topic_found?(response.topics, topic_arn)
        puts "Topic found."
        return true
      end
    end
  end
  puts "Topic not found."
  return false
rescue StandardError => e
  puts "Topic not found: #{e.message}"
  return false
end

```

Buat topik di Amazon SNS dan kemudian berlangganan alamat email untuk menerima pemberitahuan tentang topik itu.

```

# Creates a topic in Amazon SNS
# and then subscribes an email address to receive notifications to that topic.
#
# @param sns_client [Aws::SNS::Client] An initialized Amazon SNS client.
# @param topic_name [String] The name of the topic to create.

```

```

# @param email_address [String] The email address of the recipient to notify.
# @return [String] The ARN of the topic that was created.
# @example
#   puts create_topic(
#     Aws::SNS::Client.new(region: 'us-east-1'),
#     'aws-doc-sdk-examples-topic',
#     'mary@example.com'
#   )
def create_topic(sns_client, topic_name, email_address)
  puts "Creating the topic named '#{topic_name}'..."
  topic_response = sns_client.create_topic(name: topic_name)
  puts "Topic created with ARN '#{topic_response.topic_arn}'."
  subscription_response = sns_client.subscribe(
    topic_arn: topic_response.topic_arn,
    protocol: "email",
    endpoint: email_address,
    return_subscription_arn: true
  )
  puts "Subscription created with ARN " \
    "'#{subscription_response.subscription_arn}'. Have the owner of the " \
    "email address '#{email_address}' check their inbox in a few minutes " \
    "and confirm the subscription to start receiving notification emails."
  return topic_response.topic_arn
rescue StandardError => e
  puts "Error creating or subscribing to topic: #{e.message}"
  return "Error"
end

```

Periksa apakah peran yang ditentukan AWS Identity and Access Management (IAM) ada di antara yang disediakan untuk fungsi ini.

```

# Checks whether the specified AWS Identity and Access Management (IAM)
# role exists among those provided to this function.
# This is a helper function that is called by the role_exists? function.
#
# @param roles [Array] An array of Aws::IAM::Role objects.
# @param role_arn [String] The ARN of the role to find.
# @return [Boolean] true if the role ARN was found; otherwise, false.
# @example
#   iam_client = Aws::IAM::Client.new(region: 'us-east-1')
#   response = iam_client.list_roles
#   if role_found?(

```

```

#   response.roles,
#   'arn:aws:iam::111111111111:role/aws-doc-sdk-examples-ec2-state-change'
# )
#   puts 'Role found.'
# end
def role_found?(roles, role_arn)
  roles.each do |role|
    return true if role.arn == role_arn
  end
  return false
end
end

```

Periksa apakah peran yang ditentukan ada di antara yang tersedia untuk pemanggil di IAM.

```

# Checks whether the specified role exists among those available to the
# caller in AWS Identity and Access Management (IAM).
#
# @param iam_client [Aws::IAM::Client] An initialized IAM client.
# @param role_arn [String] The ARN of the role to find.
# @return [Boolean] true if the role ARN was found; otherwise, false.
# @example
#   exit 1 unless role_exists?(
#     Aws::IAM::Client.new(region: 'us-east-1'),
#     'arn:aws:iam::111111111111:role/aws-doc-sdk-examples-ec2-state-change'
#   )
def role_exists?(iam_client, role_arn)
  puts "Searching for role with ARN '#{role_arn}'..."
  response = iam_client.list_roles
  if response.roles.count.positive?
    if role_found?(response.roles, role_arn)
      puts "Role found."
      return true
    end
  end
  while response.next_page? do
    response = response.next_page
    if response.roles.count.positive?
      if role_found?(response.roles, role_arn)
        puts "Role found."
        return true
      end
    end
  end
end
end

```

```
end
puts "Role not found."
return false
rescue StandardError => e
  puts "Role not found: #{e.message}"
  return false
end
```

Buat peran dalam IAM.

```
# Creates a role in AWS Identity and Access Management (IAM).
# This role is used by a rule in Amazon EventBridge to allow
# that rule to operate within the caller's account.
# This role is designed to be used specifically by this code example.
#
# @param iam_client [Aws::IAM::Client] An initialized IAM client.
# @param role_name [String] The name of the role to create.
# @return [String] The ARN of the role that was created.
# @example
#   puts create_role(
#     Aws::IAM::Client.new(region: 'us-east-1'),
#     'aws-doc-sdk-examples-ec2-state-change'
#   )
def create_role(iam_client, role_name)
  puts "Creating the role named '#{role_name}'..."
  response = iam_client.create_role(
    assume_role_policy_document: {
      'Version': "2012-10-17",
      'Statement': [
        {
          'Sid': "",
          'Effect': "Allow",
          'Principal': {
            'Service': "events.amazonaws.com"
          },
          'Action': "sts:AssumeRole"
        }
      ]
    }
  ).to_json,
  path: "/",
  role_name: role_name
)
```

```

puts "Role created with ARN '#{response.role.arn}'."
puts "Adding access policy to role..."
iam_client.put_role_policy(
  policy_document: {
    'Version': "2012-10-17",
    'Statement': [
      {
        'Sid': "CloudWatchEventsFullAccess",
        'Effect': "Allow",
        'Resource': "*",
        'Action': "events:*"
      },
      {
        'Sid': "IAMPassRoleForCloudWatchEvents",
        'Effect': "Allow",
        'Resource': "arn:aws:iam::*:role/AWS_Events_Invoke_Targets",
        'Action': "iam:PassRole"
      }
    ]
  }.to_json,
  policy_name: "CloudWatchEventsPolicy",
  role_name: role_name
)
puts "Access policy added to role."
return response.role.arn
rescue StandardError => e
  puts "Error creating role or adding policy to it: #{e.message}"
  puts "If the role was created, you must add the access policy " \
    "to the role yourself, or delete the role yourself and try again."
  return "Error"
end

```

Memeriksa apakah EventBridge aturan yang ditentukan ada di antara yang disediakan untuk fungsi ini.

```

# Checks whether the specified Amazon EventBridge rule exists among
# those provided to this function.
# This is a helper function that is called by the rule_exists? function.
#
# @param rules [Array] An array of Aws::CloudWatchEvents::Types::Rule objects.
# @param rule_arn [String] The name of the rule to find.
# @return [Boolean] true if the name of the rule was found; otherwise, false.

```

```

# @example
#   cloudwatchevents_client = Aws::CloudWatch::Client.new(region: 'us-east-1')
#   response = cloudwatchevents_client.list_rules
#   if rule_found?(response.rules, 'aws-doc-sdk-examples-ec2-state-change')
#     puts 'Rule found.'
#   end
def rule_found?(rules, rule_name)
  rules.each do |rule|
    return true if rule.name == rule_name
  end
  return false
end
end

```

Memeriksa apakah aturan yang ditentukan ada di antara yang tersedia untuk pemanggil di EventBridge.

```

# Checks whether the specified rule exists among those available to the
# caller in Amazon EventBridge.
#
# @param cloudwatchevents_client [Aws::CloudWatchEvents::Client]
#   An initialized Amazon EventBridge client.
# @param rule_name [String] The name of the rule to find.
# @return [Boolean] true if the rule name was found; otherwise, false.
# @example
#   exit 1 unless rule_exists?(
#     Aws::CloudWatch::Client.new(region: 'us-east-1')
#     'aws-doc-sdk-examples-ec2-state-change'
#   )
def rule_exists?(cloudwatchevents_client, rule_name)
  puts "Searching for rule with name '#{rule_name}'..."
  response = cloudwatchevents_client.list_rules
  if response.rules.count.positive?
    if rule_found?(response.rules, rule_name)
      puts "Rule found."
      return true
    end
  end
  while response.next_page? do
    response = response.next_page
    if response.rules.count.positive?
      if rule_found?(response.rules, rule_name)
        puts "Rule found."
        return true
      end
    end
  end
end

```

```

        end
      end
    end
  end
  puts "Rule not found."
  return false
rescue StandardError => e
  puts "Rule not found: #{e.message}"
  return false
end

```

Buat aturan di EventBridge.

```

# Creates a rule in Amazon EventBridge.
# This rule is triggered whenever an available instance in
# Amazon EC2 changes to the specified state.
# This rule is designed to be used specifically by this code example.
#
# Prerequisites:
#
# - A role in AWS Identity and Access Management (IAM) that is designed
#   to be used specifically by this code example.
# - A topic in Amazon SNS.
#
# @param cloudwatchevents_client [Aws::CloudWatchEvents::Client]
#   An initialized Amazon EventBridge client.
# @param rule_name [String] The name of the rule to create.
# @param rule_description [String] Some description for this rule.
# @param instance_state [String] The state that available instances in
#   Amazon EC2 must change to, to
#   trigger this rule.
# @param role_arn [String] The Amazon Resource Name (ARN) of the IAM role.
# @param target_id [String] Some identifying string for the rule's target.
# @param topic_arn [String] The ARN of the Amazon SNS topic.
# @return [Boolean] true if the rule was created; otherwise, false.
# @example
#   exit 1 unless rule_created?(
#     Aws::CloudWatchEvents::Client.new(region: 'us-east-1'),
#     'aws-doc-sdk-examples-ec2-state-change',
#     'Triggers when any available EC2 instance starts.',
#     'running',
#     'arn:aws:iam::111111111111:role/aws-doc-sdk-examples-ec2-state-change',

```

```
# 'sns-topic',
# 'arn:aws:sns:us-east-1:111111111111:aws-doc-sdk-examples-topic'
# )
def rule_created?(
  cloudwatchevents_client,
  rule_name,
  rule_description,
  instance_state,
  role_arn,
  target_id,
  topic_arn
)
  puts "Creating rule with name '#{rule_name}'..."
  put_rule_response = cloudwatchevents_client.put_rule(
    name: rule_name,
    description: rule_description,
    event_pattern: {
      'source': [
        "aws.ec2"
      ],
      'detail-type': [
        "EC2 Instance State-change Notification"
      ],
      'detail': {
        'state': [
          instance_state
        ]
      }
    }.to_json,
    state: "ENABLED",
    role_arn: role_arn
  )
  puts "Rule created with ARN '#{put_rule_response.rule_arn}'."

  put_targets_response = cloudwatchevents_client.put_targets(
    rule: rule_name,
    targets: [
      {
        id: target_id,
        arn: topic_arn
      }
    ]
  )
  if put_targets_response.key?(:failed_entry_count) &&
```



```

    put_targets_response.failed_entry_count > 0
  puts "Error(s) adding target to rule:"
  put_targets_response.failed_entries.each do |failure|
    puts failure.error_message
  end
  return false
else
  return true
end
end
rescue StandardError => e
  puts "Error creating rule or adding target to rule: #{e.message}"
  puts "If the rule was created, you must add the target " \
    "to the rule yourself, or delete the rule yourself and try again."
  return false
end
end

```

Periksa untuk melihat apakah grup log yang ditentukan ada di antara yang tersedia untuk pemanggil di Amazon CloudWatch Logs.

```

# Checks to see whether the specified log group exists among those available
# to the caller in Amazon CloudWatch Logs.
#
# @param cloudwatchlogs_client [Aws::CloudWatchLogs::Client] An initialized
#   Amazon CloudWatch Logs client.
# @param log_group_name [String] The name of the log group to find.
# @return [Boolean] true if the log group name was found; otherwise, false.
# @example
#   exit 1 unless log_group_exists?(
#     Aws::CloudWatchLogs::Client.new(region: 'us-east-1'),
#     'aws-doc-sdk-examples-cloudwatch-log'
#   )
def log_group_exists?(cloudwatchlogs_client, log_group_name)
  puts "Searching for log group with name '#{log_group_name}'..."
  response = cloudwatchlogs_client.describe_log_groups(
    log_group_name_prefix: log_group_name
  )
  if response.log_groups.count.positive?
    response.log_groups.each do |log_group|
      if log_group.log_group_name == log_group_name
        puts "Log group found."
        return true
      end
    end
  end
end

```

```
    end
  end
  puts "Log group not found."
  return false
rescue StandardError => e
  puts "Log group not found: #{e.message}"
  return false
end
```

Buat grup log di CloudWatch Log.

```
# Creates a log group in Amazon CloudWatch Logs.
#
# @param cloudwatchlogs_client [Aws::CloudWatchLogs::Client] An initialized
#   Amazon CloudWatch Logs client.
# @param log_group_name [String] The name of the log group to create.
# @return [Boolean] true if the log group name was created; otherwise, false.
# @example
#   exit 1 unless log_group_created?(
#     Aws::CloudWatchLogs::Client.new(region: 'us-east-1'),
#     'aws-doc-sdk-examples-cloudwatch-log'
#   )
def log_group_created?(cloudwatchlogs_client, log_group_name)
  puts "Attempting to create log group with the name '#{log_group_name}'..."
  cloudwatchlogs_client.create_log_group(log_group_name: log_group_name)
  puts "Log group created."
  return true
rescue StandardError => e
  puts "Error creating log group: #{e.message}"
  return false
end
```

Tulis acara ke aliran log di CloudWatch Log.

```
# Writes an event to a log stream in Amazon CloudWatch Logs.
#
# Prerequisites:
#
# - A log group in Amazon CloudWatch Logs.
# - A log stream within the log group.
#
```

```
# @param cloudwatchlogs_client [Aws::CloudWatchLogs::Client] An initialized
#   Amazon CloudWatch Logs client.
# @param log_group_name [String] The name of the log group.
# @param log_stream_name [String] The name of the log stream within
#   the log group.
# @param message [String] The message to write to the log stream.
# @param sequence_token [String] If available, the sequence token from the
#   message that was written immediately before this message. This sequence
#   token is returned by Amazon CloudWatch Logs whenever you programmatically
#   write a message to the log stream.
# @return [String] The sequence token that is returned by
#   Amazon CloudWatch Logs after successfully writing the message to the
#   log stream.
# @example
#   puts log_event(
#     Aws::EC2::Client.new(region: 'us-east-1'),
#     'aws-doc-sdk-examples-cloudwatch-log'
#     '2020/11/19/53f985be-199f-408e-9a45-fc242df41fEX',
#     "Instance 'i-033c48ef067af3dEX' restarted.",
#     '495426724868310740095796045676567882148068632824696073EX'
#   )
def log_event(
  cloudwatchlogs_client,
  log_group_name,
  log_stream_name,
  message,
  sequence_token
)
  puts "Attempting to log '#{message}' to log stream '#{log_stream_name}'..."
  event = {
    log_group_name: log_group_name,
    log_stream_name: log_stream_name,
    log_events: [
      {
        timestamp: (Time.now.utc.to_f.round(3) * 1_000).to_i,
        message: message
      }
    ]
  }
  unless sequence_token.empty?
    event[:sequence_token] = sequence_token
  end

  response = cloudwatchlogs_client.put_log_events(event)
```

```

    puts "Message logged."
    return response.next_sequence_token
  rescue StandardError => e
    puts "Message not logged: #{e.message}"
  end
end

```

Mulai ulang instans Amazon Elastic Compute Cloud (AmazonEC2) dan tambahkan informasi tentang aktivitas terkait ke aliran CloudWatch log di Log.

```

# Restarts an Amazon EC2 instance
# and adds information about the related activity to a log stream
# in Amazon CloudWatch Logs.
#
# Prerequisites:
#
# - The Amazon EC2 instance to restart.
# - The log group in Amazon CloudWatch Logs to add related activity
#   information to.
#
# @param ec2_client [Aws::EC2::Client] An initialized Amazon EC2 client.
# @param cloudwatchlogs_client [Aws::CloudWatchLogs::Client]
#   An initialized Amazon CloudWatch Logs client.
# @param instance_id [String] The ID of the instance.
# @param log_group_name [String] The name of the log group.
# @return [Boolean] true if the instance was restarted and the information
#   was written to the log stream; otherwise, false.
# @example
#   exit 1 unless instance_restarted?(
#     Aws::EC2::Client.new(region: 'us-east-1'),
#     Aws::CloudWatchLogs::Client.new(region: 'us-east-1'),
#     'i-033c48ef067af3dEX',
#     'aws-doc-sdk-examples-cloudwatch-log'
#   )
def instance_restarted?(
  ec2_client,
  cloudwatchlogs_client,
  instance_id,
  log_group_name
)
  log_stream_name = "#{Time.now.year}/#{Time.now.month}/#{Time.now.day}/" \
    "#{SecureRandom.uuid}"

```

```
cloudwatchlogs_client.create_log_stream(
  log_group_name: log_group_name,
  log_stream_name: log_stream_name
)
sequence_token = ""

puts "Attempting to stop the instance with the ID '#{instance_id}'. " \
     "This might take a few minutes..."
ec2_client.stop_instances(instance_ids: [instance_id])
ec2_client.wait_until(:instance_stopped, instance_ids: [instance_id])
puts "Instance stopped."
sequence_token = log_event(
  cloudwatchlogs_client,
  log_group_name,
  log_stream_name,
  "Instance '#{instance_id}' stopped.",
  sequence_token
)

puts "Attempting to restart the instance. This might take a few minutes..."
ec2_client.start_instances(instance_ids: [instance_id])
ec2_client.wait_until(:instance_running, instance_ids: [instance_id])
puts "Instance restarted."
sequence_token = log_event(
  cloudwatchlogs_client,
  log_group_name,
  log_stream_name,
  "Instance '#{instance_id}' restarted.",
  sequence_token
)

return true
rescue StandardError => e
  puts "Error creating log stream or stopping or restarting the instance: " \
       "#{e.message}"
  log_event(
    cloudwatchlogs_client,
    log_group_name,
    log_stream_name,
    "Error stopping or starting instance '#{instance_id}': #{e.message}",
    sequence_token
  )
  return false
end
```

Menampilkan informasi tentang aktivitas untuk aturan di EventBridge.

```
# Displays information about activity for a rule in Amazon EventBridge.
#
# Prerequisites:
#
# - A rule in Amazon EventBridge.
#
# @param cloudwatch_client [Amazon::CloudWatch::Client] An initialized
#   Amazon CloudWatch client.
# @param rule_name [String] The name of the rule.
# @param start_time [Time] The timestamp that determines the first datapoint
#   to return. Can also be expressed as DateTime, Date, Integer, or String.
# @param end_time [Time] The timestamp that determines the last datapoint
#   to return. Can also be expressed as DateTime, Date, Integer, or String.
# @param period [Integer] The interval, in seconds, to check for activity.
# @example
#   display_rule_activity(
#     Aws::CloudWatch::Client.new(region: 'us-east-1'),
#     'aws-doc-sdk-examples-ec2-state-change',
#     Time.now - 600, # Start checking from 10 minutes ago.
#     Time.now, # Check up until now.
#     60 # Check every minute during those 10 minutes.
#   )
def display_rule_activity(
  cloudwatch_client,
  rule_name,
  start_time,
  end_time,
  period
)
  puts "Attempting to display rule activity..."
  response = cloudwatch_client.get_metric_statistics(
    namespace: "AWS/Events",
    metric_name: "Invocations",
    dimensions: [
      {
        name: "RuleName",
        value: rule_name
      }
    ],

```

```

    start_time: start_time,
    end_time: end_time,
    period: period,
    statistics: ["Sum"],
    unit: "Count"
  )

  if response.key?(:datapoints) && response.datapoints.count.positive?
    puts "The event rule '#{rule_name}' was triggered:"
    response.datapoints.each do |datapoint|
      puts "  #{datapoint.sum} time(s) at #{datapoint.timestamp}"
    end
  else
    puts "The event rule '#{rule_name}' was not triggered during the " \
      "specified time period."
  end
rescue StandardError => e
  puts "Error getting information about event rule activity: #{e.message}"
end

```

Menampilkan informasi log untuk semua aliran log dalam grup CloudWatch log Log.

```

# Displays log information for all of the log streams in a log group in
# Amazon CloudWatch Logs.
#
# Prerequisites:
#
# - A log group in Amazon CloudWatch Logs.
#
# @param cloudwatchlogs_client [Amazon::CloudWatchLogs::Client] An initialized
#   Amazon CloudWatch Logs client.
# @param log_group_name [String] The name of the log group.
# @example
#   display_log_data(
#     Amazon::CloudWatchLogs::Client.new(region: 'us-east-1'),
#     'aws-doc-sdk-examples-cloudwatch-log'
#   )
def display_log_data(cloudwatchlogs_client, log_group_name)
  puts "Attempting to display log stream data for the log group " \
    "named '#{log_group_name}'..."
  describe_log_streams_response = cloudwatchlogs_client.describe_log_streams(
    log_group_name: log_group_name,

```

```

    order_by: "LastEventTime",
    descending: true
  )
  if describe_log_streams_response.key?(:log_streams) &&
    describe_log_streams_response.log_streams.count.positive?
    describe_log_streams_response.log_streams.each do |log_stream|
      get_log_events_response = cloudwatchlogs_client.get_log_events(
        log_group_name: log_group_name,
        log_stream_name: log_stream.log_stream_name
      )
      puts "\nLog messages for '#{log_stream.log_stream_name}':"
      puts "-" * (log_stream.log_stream_name.length + 20)
      if get_log_events_response.key?(:events) &&
        get_log_events_response.events.count.positive?
        get_log_events_response.events.each do |event|
          puts event.message
        end
      else
        puts "No log messages for this log stream."
      end
    end
  end
end
rescue StandardError => e
  puts "Error getting information about the log streams or their messages: " \
    "#{e.message}"
end

```

Tampilkan pengingat ke penelepon untuk membersihkan AWS sumber daya terkait secara manual yang tidak lagi mereka butuhkan.

```

# Displays a reminder to the caller to manually clean up any associated
# AWS resources that they no longer need.
#
# @param topic_name [String] The name of the Amazon SNS topic.
# @param role_name [String] The name of the IAM role.
# @param rule_name [String] The name of the Amazon EventBridge rule.
# @param log_group_name [String] The name of the Amazon CloudWatch Logs log group.
# @param instance_id [String] The ID of the Amazon EC2 instance.
# @example
#   manual_cleanup_notice(
#     'aws-doc-sdk-examples-topic',

```



```
# 'aws-doc-sdk-examples-cloudwatch-events-rule-role',
# 'aws-doc-sdk-examples-ec2-state-change',
# 'aws-doc-sdk-examples-cloudwatch-log',
# 'i-033c48ef067af3dEX'
# )
def manual_cleanup_notice(
  topic_name, role_name, rule_name, log_group_name, instance_id
)
  puts "-" * 10
  puts "Some of the following AWS resources might still exist in your account."
  puts "If you no longer want to use this code example, then to clean up"
  puts "your AWS account and avoid unexpected costs, you might want to"
  puts "manually delete any of the following resources if they exist:"
  puts "- The Amazon SNS topic named '#{topic_name}'."
  puts "- The IAM role named '#{role_name}'."
  puts "- The Amazon EventBridge rule named '#{rule_name}'."
  puts "- The Amazon CloudWatch Logs log group named '#{log_group_name}'."
  puts "- The Amazon EC2 instance with the ID '#{instance_id}'."
end

# Example usage:
def run_me
  # Properties for the Amazon SNS topic.
  topic_name = "aws-doc-sdk-examples-topic"
  email_address = "mary@example.com"
  # Properties for the IAM role.
  role_name = "aws-doc-sdk-examples-cloudwatch-events-rule-role"
  # Properties for the Amazon EventBridge rule.
  rule_name = "aws-doc-sdk-examples-ec2-state-change"
  rule_description = "Triggers when any available EC2 instance starts."
  instance_state = "running"
  target_id = "sns-topic"
  # Properties for the Amazon EC2 instance.
  instance_id = "i-033c48ef067af3dEX"
  # Properties for displaying the event rule's activity.
  start_time = Time.now - 600 # Go back over the past 10 minutes
                                # (10 minutes * 60 seconds = 600 seconds).
  end_time = Time.now
  period = 60 # Look back every 60 seconds over the past 10 minutes.
  # Properties for the Amazon CloudWatch Logs log group.
  log_group_name = "aws-doc-sdk-examples-cloudwatch-log"
  # AWS service clients for this code example.
  region = "us-east-1"
  sts_client = Aws::STS::Client.new(region: region)
```

```
sns_client = Aws::SNS::Client.new(region: region)
iam_client = Aws::IAM::Client.new(region: region)
cloudwatchevents_client = Aws::CloudWatchEvents::Client.new(region: region)
ec2_client = Aws::EC2::Client.new(region: region)
cloudwatch_client = Aws::CloudWatch::Client.new(region: region)
cloudwatchlogs_client = Aws::CloudWatchLogs::Client.new(region: region)

# Get the caller's account ID for use in forming
# Amazon Resource Names (ARNs) that this code relies on later.
account_id = sts_client.get_caller_identity.account

# If the Amazon SNS topic doesn't exist, create it.
topic_arn = "arn:aws:sns:#{region}:#{account_id}:#{topic_name}"
unless topic_exists?(sns_client, topic_arn)
  topic_arn = create_topic(sns_client, topic_name, email_address)
  if topic_arn == "Error"
    puts "Could not create the Amazon SNS topic correctly. Program stopped."
    manual_cleanup_notice(
      topic_name, role_name, rule_name, log_group_name, instance_id
    )
    exit 1
  end
end

# If the IAM role doesn't exist, create it.
role_arn = "arn:aws:iam:#{account_id}:role/#{role_name}"
unless role_exists?(iam_client, role_arn)
  role_arn = create_role(iam_client, role_name)
  if role_arn == "Error"
    puts "Could not create the IAM role correctly. Program stopped."
    manual_cleanup_notice(
      topic_name, role_name, rule_name, log_group_name, instance_id
    )
  end
end

# If the Amazon EventBridge rule doesn't exist, create it.
unless rule_exists?(cloudwatchevents_client, rule_name)
  unless rule_created?(
    cloudwatchevents_client,
    rule_name,
    rule_description,
    instance_state,
    role_arn,
```

```
        target_id,
        topic_arn
    )
    puts "Could not create the Amazon EventBridge rule correctly. " \
        "Program stopped."
    manual_cleanup_notice(
        topic_name, role_name, rule_name, log_group_name, instance_id
    )
end
end

# If the Amazon CloudWatch Logs log group doesn't exist, create it.
unless log_group_exists?(cloudwatchlogs_client, log_group_name)
    unless log_group_created?(cloudwatchlogs_client, log_group_name)
        puts "Could not create the Amazon CloudWatch Logs log group " \
            "correctly. Program stopped."
        manual_cleanup_notice(
            topic_name, role_name, rule_name, log_group_name, instance_id
        )
    end
end

# Restart the Amazon EC2 instance, which triggers the rule.
unless instance_restarted?(
    ec2_client,
    cloudwatchlogs_client,
    instance_id,
    log_group_name
)
    puts "Could not restart the instance to trigger the rule. " \
        "Continuing anyway to show information about the rule and logs..."
end

# Display how many times the rule was triggered over the past 10 minutes.
display_rule_activity(
    cloudwatch_client,
    rule_name,
    start_time,
    end_time,
    period
)

# Display related log data in Amazon CloudWatch Logs.
display_log_data(cloudwatchlogs_client, log_group_name)
```

```
# Reminder the caller to clean up any AWS resources that are used
# by this code example and are no longer needed.
manual_cleanup_notice(
  topic_name, role_name, rule_name, log_group_name, instance_id
)
end

run_me if $PROGRAM_NAME == __FILE__
```

- Untuk API detailnya, lihat topik berikut di AWS SDK for Ruby API Referensi.
 - [PutEvents](#)
 - [PutRule](#)

AWS Glue contoh menggunakan SDK untuk Ruby

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan AWS SDK for Ruby with AWS Glue.

Dasar-dasar adalah contoh kode yang menunjukkan kepada Anda bagaimana melakukan operasi penting dalam suatu layanan.

Tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Memulai

Halo AWS Glue

Contoh kode berikut menunjukkan cara untuk mulai menggunakan AWS Glue.

SDK untuk Ruby

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
require 'aws-sdk-glue'
require 'logger'

# GlueManager is a class responsible for managing AWS Glue operations
# such as listing all Glue jobs in the current AWS account.
class GlueManager
  def initialize(client)
    @client = client
    @logger = Logger.new($stdout)
  end

  # Lists and prints all Glue jobs in the current AWS account.
  def list_jobs
    @logger.info('Here are the Glue jobs in your account:')

    paginator = @client.get_jobs(max_results: 10)
    jobs = []

    paginator.each_page do |page|
      jobs.concat(page.jobs)
    end

    if jobs.empty?
      @logger.info("You don't have any Glue jobs.")
    else
      jobs.each do |job|
        @logger.info("- #{job.name}")
      end
    end
  end
end

if $PROGRAM_NAME == __FILE__
```

```
glue_client = Aws::Glue::Client.new
manager = GlueManager.new(glue_client)
manager.list_jobs
end
```

- Untuk API detailnya, lihat [ListJobs](#) di AWS SDK for Ruby API Referensi.

Topik

- [Hal-hal mendasar](#)
- [Tindakan](#)

Hal-hal mendasar

Pelajari dasar-dasarnya

Contoh kode berikut ini menunjukkan cara:

- Buat crawler yang merayapi bucket Amazon S3 publik dan membuat database CSV metadata yang diformat.
- Daftar informasi tentang database dan tabel di situs Anda AWS Glue Data Catalog.
- Buat pekerjaan untuk mengekstrak CSV data dari bucket S3, mengubah data, dan memuat output yang JSON diformat menjadi bucket S3 lain.
- Buat daftar informasi tentang menjalankan pekerjaan, melihat data yang diubah, dan membersihkan sumber daya.

Untuk informasi selengkapnya, lihat [Tutorial: Memulai AWS Glue Studio](#).

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

Buat kelas yang membungkus AWS Glue fungsi yang digunakan dalam skenario.

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing a
simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods for
interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end

  # Retrieves information about a specific crawler.
  #
  # @param name [String] The name of the crawler to retrieve information about.
  # @return [Aws::Glue::Types::Crawler, nil] The crawler object if found, or nil if
not found.
  def get_crawler(name)
    @glue_client.get_crawler(name: name)
  rescue Aws::Glue::Errors::EntityNotFoundException
    @logger.info("Crawler #{name} doesn't exist.")
    false
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not get crawler #{name}: \n#{e.message}")
    raise
  end

  # Creates a new crawler with the specified configuration.
  #
  # @param name [String] The name of the crawler.
  # @param role_arn [String] The ARN of the IAM role to be used by the crawler.
  # @param db_name [String] The name of the database where the crawler stores its
metadata.
  # @param db_prefix [String] The prefix to be added to the names of tables that the
crawler creates.
  # @param s3_target [String] The S3 path that the crawler will crawl.
  # @return [void]
  def create_crawler(name, role_arn, db_name, db_prefix, s3_target)
    @glue_client.create_crawler(
      name: name,
      role: role_arn,
```

```
        database_name: db_name,
        targets: {
          s3_targets: [
            {
              path: s3_target
            }
          ]
        }
      )
    rescue Aws::Glue::Errors::GlueException => e
      @logger.error("Glue could not create crawler: \n#{e.message}")
      raise
    end

    # Starts a crawler with the specified name.
    #
    # @param name [String] The name of the crawler to start.
    # @return [void]
    def start_crawler(name)
      @glue_client.start_crawler(name: name)
    rescue Aws::Glue::Errors::ServiceError => e
      @logger.error("Glue could not start crawler #{name}: \n#{e.message}")
      raise
    end

    # Deletes a crawler with the specified name.
    #
    # @param name [String] The name of the crawler to delete.
    # @return [void]
    def delete_crawler(name)
      @glue_client.delete_crawler(name: name)
    rescue Aws::Glue::Errors::ServiceError => e
      @logger.error("Glue could not delete crawler #{name}: \n#{e.message}")
      raise
    end

    # Retrieves information about a specific database.
    #
    # @param name [String] The name of the database to retrieve information about.
    # @return [Aws::Glue::Types::Database, nil] The database object if found, or nil
    if not found.
    def get_database(name)
      response = @glue_client.get_database(name: name)
      response.database
    end
  end
end
```



```
rescue Aws::Glue::Errors::GlueException => e
  @logger.error("Glue could not get database #{name}: \n#{e.message}")
  raise
end

# Retrieves a list of tables in the specified database.
#
# @param db_name [String] The name of the database to retrieve tables from.
# @return [Array<Aws::Glue::Types::Table>]
def get_tables(db_name)
  response = @glue_client.get_tables(database_name: db_name)
  response.table_list
rescue Aws::Glue::Errors::GlueException => e
  @logger.error("Glue could not get tables #{db_name}: \n#{e.message}")
  raise
end

# Creates a new job with the specified configuration.
#
# @param name [String] The name of the job.
# @param description [String] The description of the job.
# @param role_arn [String] The ARN of the IAM role to be used by the job.
# @param script_location [String] The location of the ETL script for the job.
# @return [void]
def create_job(name, description, role_arn, script_location)
  @glue_client.create_job(
    name: name,
    description: description,
    role: role_arn,
    command: {
      name: "glueetl",
      script_location: script_location,
      python_version: "3"
    },
    glue_version: "3.0"
  )
rescue Aws::Glue::Errors::GlueException => e
  @logger.error("Glue could not create job #{name}: \n#{e.message}")
  raise
end

# Starts a job run for the specified job.
#
# @param name [String] The name of the job to start the run for.
```

```
# @param input_database [String] The name of the input database for the job.
# @param input_table [String] The name of the input table for the job.
# @param output_bucket_name [String] The name of the output S3 bucket for the job.
# @return [String] The ID of the started job run.
def start_job_run(name, input_database, input_table, output_bucket_name)
  response = @glue_client.start_job_run(
    job_name: name,
    arguments: {
      '--input_database': input_database,
      '--input_table': input_table,
      '--output_bucket_url': "s3://#{output_bucket_name}/"
    }
  )
  response.job_run_id
rescue Aws::Glue::Errors::GlueException => e
  @logger.error("Glue could not start job run #{name}: \n#{e.message}")
  raise
end

# Retrieves a list of jobs in AWS Glue.
#
# @return [Aws::Glue::Types::ListJobsResponse]
def list_jobs
  @glue_client.list_jobs
rescue Aws::Glue::Errors::GlueException => e
  @logger.error("Glue could not list jobs: \n#{e.message}")
  raise
end

# Retrieves a list of job runs for the specified job.
#
# @param job_name [String] The name of the job to retrieve job runs for.
# @return [Array<Aws::Glue::Types::JobRun>]
def get_job_runs(job_name)
  response = @glue_client.get_job_runs(job_name: job_name)
  response.job_runs
rescue Aws::Glue::Errors::GlueException => e
  @logger.error("Glue could not get job runs: \n#{e.message}")
end

# Retrieves data for a specific job run.
#
# @param job_name [String] The name of the job run to retrieve data for.
# @return [Glue::Types::GetJobRunResponse]
```

```
def get_job_run(job_name, run_id)
  @glue_client.get_job_run(job_name: job_name, run_id: run_id)
rescue Aws::Glue::Errors::GlueException => e
  @logger.error("Glue could not get job runs: \n#{e.message}")
end

# Deletes a job with the specified name.
#
# @param job_name [String] The name of the job to delete.
# @return [void]
def delete_job(job_name)
  @glue_client.delete_job(job_name: job_name)
rescue Aws::Glue::Errors::ServiceError => e
  @logger.error("Glue could not delete job: \n#{e.message}")
end

# Deletes a table with the specified name.
#
# @param database_name [String] The name of the catalog database in which the
table resides.
# @param table_name [String] The name of the table to be deleted.
# @return [void]
def delete_table(database_name, table_name)
  @glue_client.delete_table(database_name: database_name, name: table_name)
rescue Aws::Glue::Errors::ServiceError => e
  @logger.error("Glue could not delete job: \n#{e.message}")
end

# Removes a specified database from a Data Catalog.
#
# @param database_name [String] The name of the database to delete.
# @return [void]
def delete_database(database_name)
  @glue_client.delete_database(name: database_name)
rescue Aws::Glue::Errors::ServiceError => e
  @logger.error("Glue could not delete database: \n#{e.message}")
end

# Uploads a job script file to an S3 bucket.
#
# @param file_path [String] The local path of the job script file.
# @param bucket_resource [Aws::S3::Bucket] The S3 bucket resource to upload the
file to.
# @return [void]
```

```
def upload_job_script(file_path, bucket_resource)
  File.open(file_path) do |file|
    bucket_resource.client.put_object({
      body: file,
      bucket: bucket_resource.name,
      key: file_path
    })
  end
rescue Aws::S3::Errors::S3UploadFailedError => e
  @logger.error("S3 could not upload job script: \n#{e.message}")
  raise
end

end
```

Buat kelas yang menjalankan skenario.

```
class GlueCrawlerJobScenario
  def initialize(glue_client, glue_service_role, glue_bucket, logger)
    @glue_client = glue_client
    @glue_service_role = glue_service_role
    @glue_bucket = glue_bucket
    @logger = logger
  end

  def run(crawler_name, db_name, db_prefix, data_source, job_script, job_name)
    wrapper = GlueWrapper.new(@glue_client, @logger)

    new_step(1, "Create a crawler")
    puts "Checking for crawler #{crawler_name}."
    crawler = wrapper.get_crawler(crawler_name)
    if crawler == false
      puts "Creating crawler #{crawler_name}."
      wrapper.create_crawler(crawler_name, @glue_service_role.arn, db_name,
db_prefix, data_source)
      puts "Successfully created #{crawler_name}:"
      crawler = wrapper.get_crawler(crawler_name)
      puts JSON.pretty_generate(crawler).yellow
    end
    print "\nDone!\n".green

    new_step(2, "Run a crawler to output a database.")
  end
end
```

```
puts "Location of input data analyzed by crawler: #{data_source}"
puts "Outputs: a Data Catalog database in CSV format containing metadata on
input."
wrapper.start_crawler(crawler_name)
puts "Starting crawler... (this typically takes a few minutes)"
crawler_state = nil
while crawler_state != "READY"
  custom_wait(15)
  crawler = wrapper.get_crawler(crawler_name)
  crawler_state = crawler[0]["state"]
  print "Status check: #{crawler_state}.".yellow
end
print "\nDone!\n".green

new_step(3, "Query the database.")
database = wrapper.get_database(db_name)
puts "The crawler created database #{db_name}:"
print "#{database}.".yellow
puts "\nThe database contains these tables:"
tables = wrapper.get_tables(db_name)
tables.each_with_index do |table, index|
  print "\t#{index + 1}. #{table['name']}".yellow
end
print "\nDone!\n".green

new_step(4, "Create a job definition that runs an ETL script.")
puts "Uploading Python ETL script to S3..."
wrapper.upload_job_script(job_script, @glue_bucket)
puts "Creating job definition #{job_name}:\n"
response = wrapper.create_job(job_name, "Getting started example job.",
@glue_service_role.arn, "s3://#{@glue_bucket.name}/#{job_script}")
puts JSON.pretty_generate(response).yellow
print "\nDone!\n".green

new_step(5, "Start a new job")
job_run_status = nil
job_run_id = wrapper.start_job_run(
  job_name,
  db_name,
  tables[0]["name"],
  @glue_bucket.name
)
puts "Job #{job_name} started. Let's wait for it to run."
until ["SUCCEEDED", "STOPPED", "FAILED", "TIMEOUT"].include?(job_run_status)
```

```
    custom_wait(10)
    job_run = wrapper.get_job_runs(job_name)
    job_run_status = job_run[0]["job_run_state"]
    print "Status check: #{job_name}/#{job_run_id} - #{job_run_status}.".yellow
  end
  print "\nDone!\n".green

  new_step(6, "View results from a successful job run.")
  if job_run_status == "SUCCEEDED"
    puts "Data from your job run is stored in your S3 bucket
'#{@glue_bucket.name}'. Files include:"
    begin

      # Print the key name of each object in the bucket.
      @glue_bucket.objects.each do |object_summary|
        if object_summary.key.include?("run-")
          print "#{object_summary.key}".yellow
        end
      end

      # Print the first 256 bytes of a run file
      desired_sample_objects = 1
      @glue_bucket.objects.each do |object_summary|
        if object_summary.key.include?("run-")
          if desired_sample_objects > 0
            sample_object = @glue_bucket.object(object_summary.key)
            sample = sample_object.get(range: "bytes=0-255").body.read
            puts "\nSample run file contents:"
            print "#{sample}".yellow
            desired_sample_objects -= 1
          end
        end
      end

      rescue Aws::S3::Errors::ServiceError => e
        logger.error(
          "Couldn't get job run data. Here's why: %s: %s",
          e.response.error.code, e.response.error.message
        )
        raise
      end
    end
    print "\nDone!\n".green

    new_step(7, "Delete job definition and crawler.")
```

```

    wrapper.delete_job(job_name)
    puts "Job deleted: #{job_name}."
    wrapper.delete_crawler(crawler_name)
    puts "Crawler deleted: #{crawler_name}."
    wrapper.delete_table(db_name, tables[0]["name"])
    puts "Table deleted: #{tables[0]["name"]} in #{db_name}."
    wrapper.delete_database(db_name)
    puts "Database deleted: #{db_name}."
    print "\nDone!\n".green
  end
end

def main

  banner("../helpers/banner.txt")
  puts
  "#####"
  puts "#
                                     #".yellow
  puts "#                               EXAMPLE CODE DEMO:
                                     #".yellow
  puts "#                               AWS Glue
                                     #".yellow
  puts "#
                                     #".yellow
  puts
  "#####"
  puts ""
  puts "You have launched a demo of AWS Glue using the AWS for Ruby v3 SDK. Over the
next 60 seconds, it will"
  puts "do the following:"
  puts "  1. Create a crawler."
  puts "  2. Run a crawler to output a database."
  puts "  3. Query the database."
  puts "  4. Create a job definition that runs an ETL script."
  puts "  5. Start a new job."
  puts "  6. View results from a successful job run."
  puts "  7. Delete job definition and crawler."
  puts ""

  confirm_begin
  billing
  security
  puts "\e[H\e[2J"

```

```
# Set input file names
job_script_filepath = "job_script.py"
resource_names = YAML.load_file("resource_names.yaml")

# Instantiate existing IAM role.
iam = Aws::IAM::Resource.new(region: "us-east-1")
iam_role_name = resource_names["glue_service_role"]
iam_role = iam.role(iam_role_name)

# Instantiate existing S3 bucket.
s3 = Aws::S3::Resource.new(region: "us-east-1")
s3_bucket_name = resource_names["glue_bucket"]
s3_bucket = s3.bucket(s3_bucket_name)

scenario = GlueCrawlerJobScenario.new(
  Aws::Glue::Client.new(region: "us-east-1"),
  iam_role,
  s3_bucket,
  @logger
)

random_int = rand(10 ** 4)
scenario.run(
  "doc-example-crawler-#{random_int}",
  "doc-example-database-#{random_int}",
  "doc-example-#{random_int}-",
  "s3://crawler-public-us-east-1/flight/2016/csv",
  job_script_filepath,
  "doc-example-job-#{random_int}"
)

puts "-" * 88
puts "You have reached the end of this tour of AWS Glue."
puts "To destroy CDK-created resources, run:\n      cdk destroy"
puts "-" * 88

end
```

Buat ETL skrip yang digunakan oleh AWS Glue untuk mengekstrak, mengubah, dan memuat data selama pekerjaan berjalan.


```
import sys
from awsglue.transforms import *
from awsglue.utils import getResolvedOptions
from pyspark.context import SparkContext
from awsglue.context import GlueContext
from awsglue.job import Job

"""
These custom arguments must be passed as Arguments to the StartJobRun request.
    --input_database    The name of a metadata database that is contained in your
                        AWS Glue Data Catalog and that contains tables that
describe
                        the data to be processed.
    --input_table       The name of a table in the database that describes the data
to
                        be processed.
    --output_bucket_url An S3 bucket that receives the transformed output data.
"""
args = getResolvedOptions(
    sys.argv, ["JOB_NAME", "input_database", "input_table", "output_bucket_url"]
)
sc = SparkContext()
glueContext = GlueContext(sc)
spark = glueContext.spark_session
job = Job(glueContext)
job.init(args["JOB_NAME"], args)

# Script generated for node S3 Flight Data.
S3FlightData_node1 = glueContext.create_dynamic_frame.from_catalog(
    database=args["input_database"],
    table_name=args["input_table"],
    transformation_ctx="S3FlightData_node1",
)

# This mapping performs two main functions:
# 1. It simplifies the output by removing most of the fields from the data.
# 2. It renames some fields. For example, `fl_date` is renamed to `flight_date`.
ApplyMapping_node2 = ApplyMapping.apply(
    frame=S3FlightData_node1,
    mappings=[
        ("year", "long", "year", "long"),
        ("month", "long", "month", "tinyint"),
        ("day_of_month", "long", "day", "tinyint"),
```

```
    ("fl_date", "string", "flight_date", "string"),
    ("carrier", "string", "carrier", "string"),
    ("fl_num", "long", "flight_num", "long"),
    ("origin_city_name", "string", "origin_city_name", "string"),
    ("origin_state_abr", "string", "origin_state_abr", "string"),
    ("dest_city_name", "string", "dest_city_name", "string"),
    ("dest_state_abr", "string", "dest_state_abr", "string"),
    ("dep_time", "long", "departure_time", "long"),
    ("wheels_off", "long", "wheels_off", "long"),
    ("wheels_on", "long", "wheels_on", "long"),
    ("arr_time", "long", "arrival_time", "long"),
    ("mon", "string", "mon", "string"),
  ],
  transformation_ctx="ApplyMapping_node2",
)

# Script generated for node Revised Flight Data.
RevisedFlightData_node3 = glueContext.write_dynamic_frame.from_options(
  frame=ApplyMapping_node2,
  connection_type="s3",
  format="json",
  connection_options={"path": args["output_bucket_url"], "partitionKeys": []},
  transformation_ctx="RevisedFlightData_node3",
)

job.commit()
```

- Untuk API detailnya, lihat topik berikut di AWS SDK for Ruby API Referensi.
 - [CreateCrawler](#)
 - [CreateJob](#)
 - [DeleteCrawler](#)
 - [DeleteDatabase](#)
 - [DeleteJob](#)
 - [DeleteTable](#)
 - [GetCrawler](#)
 - [GetDatabase](#)
 - [GetDatabases](#)
 - [GetJob](#)

- [GetJobRun](#)
- [GetJobRuns](#)
- [GetTables](#)
- [ListJobs](#)
- [StartCrawler](#)
- [StartJobRun](#)

Tindakan

CreateCrawler

Contoh kode berikut menunjukkan cara menggunakan `CreateCrawler`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing a
# simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods for
# interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
# calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end

  # Creates a new crawler with the specified configuration.
  #
  # @param name [String] The name of the crawler.
  # @param role_arn [String] The ARN of the IAM role to be used by the crawler.
```

```
# @param db_name [String] The name of the database where the crawler stores its
metadata.
# @param db_prefix [String] The prefix to be added to the names of tables that the
crawler creates.
# @param s3_target [String] The S3 path that the crawler will crawl.
# @return [void]
def create_crawler(name, role_arn, db_name, db_prefix, s3_target)
  @glue_client.create_crawler(
    name: name,
    role: role_arn,
    database_name: db_name,
    targets: {
      s3_targets: [
        {
          path: s3_target
        }
      ]
    }
  )
rescue Aws::Glue::Errors::GlueException => e
  @logger.error("Glue could not create crawler: \n#{e.message}")
  raise
end
```

- Untuk API detailnya, lihat [CreateCrawler](#) di AWS SDK for Ruby API Referensi.

CreateJob

Contoh kode berikut menunjukkan cara menggunakan `CreateJob`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing a
simplified interface for common operations.
```

```
# It encapsulates the functionality of the AWS SDK for Glue and provides methods for
interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end


  # Creates a new job with the specified configuration.
  #
  # @param name [String] The name of the job.
  # @param description [String] The description of the job.
  # @param role_arn [String] The ARN of the IAM role to be used by the job.
  # @param script_location [String] The location of the ETL script for the job.
  # @return [void]
  def create_job(name, description, role_arn, script_location)
    @glue_client.create_job(
      name: name,
      description: description,
      role: role_arn,
      command: {
        name: "glueetl",
        script_location: script_location,
        python_version: "3"
      },
      glue_version: "3.0"
    )
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not create job #{name}: \n#{e.message}")
    raise
  end
end
```

- Untuk API detailnya, lihat [CreateJob](#) di AWS SDK for Ruby API Referensi.

DeleteCrawler

Contoh kode berikut menunjukkan cara menggunakan `DeleteCrawler`.

SDK untuk Ruby

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing a
simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods for
interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end

  # Deletes a crawler with the specified name.
  #
  # @param name [String] The name of the crawler to delete.
  # @return [void]
  def delete_crawler(name)
    @glue_client.delete_crawler(name: name)
  rescue Aws::Glue::Errors::ServiceError => e
    @logger.error("Glue could not delete crawler #{name}: \n#{e.message}")
    raise
  end
end
```

- Untuk API detailnya, lihat [DeleteCrawler](#) di AWS SDK for Ruby API Referensi.

DeleteDatabase

Contoh kode berikut menunjukkan cara menggunakan `DeleteDatabase`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing a
simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods for
interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end

  # Removes a specified database from a Data Catalog.
  #
  # @param database_name [String] The name of the database to delete.
  # @return [void]
  def delete_database(database_name)
    @glue_client.delete_database(name: database_name)
  rescue Aws::Glue::Errors::ServiceError => e
    @logger.error("Glue could not delete database: \n#{e.message}")
  end
end
```

- Untuk API detailnya, lihat [DeleteDatabase](#) di AWS SDK for Ruby API Referensi.

DeleteJob

Contoh kode berikut menunjukkan cara menggunakan `DeleteJob`.

SDK untuk Ruby

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing a
simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods for
interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end

  # Deletes a job with the specified name.
  #
  # @param job_name [String] The name of the job to delete.
  # @return [void]
  def delete_job(job_name)
    @glue_client.delete_job(job_name: job_name)
  rescue Aws::Glue::Errors::ServiceError => e
    @logger.error("Glue could not delete job: \n#{e.message}")
  end
end
```

- Untuk API detailnya, lihat [DeleteJob](#) di AWS SDK for Ruby API Referensi.

DeleteTable

Contoh kode berikut menunjukkan cara menggunakan `DeleteTable`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing a
simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods for
interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end

  # Deletes a table with the specified name.
  #
  # @param database_name [String] The name of the catalog database in which the
table resides.
  # @param table_name [String] The name of the table to be deleted.
  # @return [void]
  def delete_table(database_name, table_name)
    @glue_client.delete_table(database_name: database_name, name: table_name)
  rescue Aws::Glue::Errors::ServiceError => e
    @logger.error("Glue could not delete job: \n#{e.message}")
  end
end
```

- Untuk API detailnya, lihat [DeleteTable](#) di AWS SDK for Ruby API Referensi.

GetCrawler

Contoh kode berikut menunjukkan cara menggunakan `GetCrawler`.

SDK untuk Ruby

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing a
simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods for
interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end

  # Retrieves information about a specific crawler.
  #
  # @param name [String] The name of the crawler to retrieve information about.
  # @return [Aws::Glue::Types::Crawler, nil] The crawler object if found, or nil if
not found.
  def get_crawler(name)
    @glue_client.get_crawler(name: name)
  rescue Aws::Glue::Errors::EntityNotFoundException
    @logger.info("Crawler #{name} doesn't exist.")
    false
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not get crawler #{name}: \n#{e.message}")
    raise
  end
end
```

- Untuk API detailnya, lihat [GetCrawler](#) di AWS SDK for Ruby API Referensi.

GetDatabase

Contoh kode berikut menunjukkan cara menggunakan `GetDatabase`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing a
# simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods for
# interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
# calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end

  # Retrieves information about a specific database.
  #
  # @param name [String] The name of the database to retrieve information about.
  # @return [Aws::Glue::Types::Database, nil] The database object if found, or nil
  # if not found.
  def get_database(name)
    response = @glue_client.get_database(name: name)
    response.database
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not get database #{name}: \n#{e.message}")
    raise
  end
end
```

- Untuk API detailnya, lihat [GetDatabase](#) di AWS SDK for Ruby API Referensi.

GetJobRun

Contoh kode berikut menunjukkan cara menggunakan `GetJobRun`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing a
simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods for
interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end

  # Retrieves data for a specific job run.
  #
  # @param job_name [String] The name of the job run to retrieve data for.
  # @return [Glue::Types::GetJobRunResponse]
  def get_job_run(job_name, run_id)
    @glue_client.get_job_run(job_name: job_name, run_id: run_id)
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not get job runs: \n#{e.message}")
  end
end
```

- Untuk API detailnya, lihat [GetJobRun](#) di AWS SDK for Ruby API Referensi.

GetJobRuns

Contoh kode berikut menunjukkan cara menggunakan `GetJobRuns`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing a
simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods for
interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end

  # Retrieves a list of job runs for the specified job.
  #
  # @param job_name [String] The name of the job to retrieve job runs for.
  # @return [Array<Aws::Glue::Types::JobRun>]
  def get_job_runs(job_name)
    response = @glue_client.get_job_runs(job_name: job_name)
    response.job_runs
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not get job runs: \n#{e.message}")
  end
end
```

- Untuk API detailnya, lihat [GetJobRuns](#) di AWS SDK for Ruby API Referensi.

GetTables

Contoh kode berikut menunjukkan cara menggunakan `GetTables`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing a
simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods for
interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end

  # Retrieves a list of tables in the specified database.
  #
  # @param db_name [String] The name of the database to retrieve tables from.
  # @return [Array<Aws::Glue::Types::Table>]
  def get_tables(db_name)
    response = @glue_client.get_tables(database_name: db_name)
    response.table_list
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not get tables #{db_name}: \n#{e.message}")
    raise
  end
end
```

- Untuk API detailnya, lihat [GetTables](#) di AWS SDK for Ruby API Referensi.

ListJobs

Contoh kode berikut menunjukkan cara menggunakan `ListJobs`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing a
simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods for
interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end

  # Retrieves a list of jobs in AWS Glue.
  #
  # @return [Aws::Glue::Types::ListJobsResponse]
  def list_jobs
    @glue_client.list_jobs
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not list jobs: \n#{e.message}")
    raise
  end
end
```

- Untuk API detailnya, lihat [ListJobs](#) di AWS SDK for Ruby API Referensi.

StartCrawler

Contoh kode berikut menunjukkan cara menggunakan `StartCrawler`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing a
simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods for
interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end


  # Starts a crawler with the specified name.
  #
  # @param name [String] The name of the crawler to start.
  # @return [void]
  def start_crawler(name)
    @glue_client.start_crawler(name: name)
  rescue Aws::Glue::Errors::ServiceError => e
    @logger.error("Glue could not start crawler #{name}: \n#{e.message}")
    raise
  end
end
```

- Untuk API detailnya, lihat [StartCrawler](#) di AWS SDK for Ruby API Referensi.

StartJobRun

Contoh kode berikut menunjukkan cara menggunakan `StartJobRun`.

SDK untuk Ruby

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
# The `GlueWrapper` class serves as a wrapper around the AWS Glue API, providing a
simplified interface for common operations.
# It encapsulates the functionality of the AWS SDK for Glue and provides methods for
interacting with Glue crawlers, databases, tables, jobs, and S3 resources.
# The class initializes with a Glue client and a logger, allowing it to make API
calls and log any errors or informational messages.
class GlueWrapper
  def initialize(glue_client, logger)
    @glue_client = glue_client
    @logger = logger
  end

  # Starts a job run for the specified job.
  #
  # @param name [String] The name of the job to start the run for.
  # @param input_database [String] The name of the input database for the job.
  # @param input_table [String] The name of the input table for the job.
  # @param output_bucket_name [String] The name of the output S3 bucket for the job.
  # @return [String] The ID of the started job run.
  def start_job_run(name, input_database, input_table, output_bucket_name)
    response = @glue_client.start_job_run(
      job_name: name,
      arguments: {
        '--input_database': input_database,
        '--input_table': input_table,
        '--output_bucket_url': "s3://#{output_bucket_name}/"
      }
    )
    response.job_run_id
  rescue Aws::Glue::Errors::GlueException => e
    @logger.error("Glue could not start job run #{name}: \n#{e.message}")
    raise
  end
end
```

- Untuk API detailnya, lihat [StartJobRun](#) di AWS SDK for Ruby API Referensi.

IAM contoh menggunakan SDK untuk Ruby

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan AWS SDK for Ruby with IAM.

Tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Skenario adalah contoh kode yang menunjukkan kepada Anda bagaimana menyelesaikan tugas tertentu dengan memanggil beberapa fungsi dalam layanan atau dikombinasikan dengan yang lain AWS layanan.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Memulai

Halo IAM

Contoh kode berikut menunjukkan cara untuk mulai menggunakan IAM.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
require 'aws-sdk-iam'
require 'logger'

# IAMManager is a class responsible for managing IAM operations
```

```
# such as listing all IAM policies in the current AWS account.
class IAMManager
  def initialize(client)
    @client = client
    @logger = Logger.new($stdout)
  end

  # Lists and prints all IAM policies in the current AWS account.
  def list_policies
    @logger.info('Here are the IAM policies in your account:')

    paginator = @client.list_policies
    policies = []

    paginator.each_page do |page|
      policies.concat(page.policies)
    end

    if policies.empty?
      @logger.info("You don't have any IAM policies.")
    else
      policies.each do |policy|
        @logger.info("- #{policy.policy_name}")
      end
    end
  end
end

if $PROGRAM_NAME == __FILE__
  iam_client = Aws::IAM::Client.new
  manager = IAMManager.new(iam_client)
  manager.list_policies
end
```

- Untuk API detailnya, lihat [ListPolicies](#) di AWS SDK for Ruby API Referensi.

Topik

- [Tindakan](#)
- [Skenario](#)

Tindakan

AttachRolePolicy

Contoh kode berikut menunjukkan cara menggunakan `AttachRolePolicy`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

Modul contoh ini mencantumkan, membuat, melampirkan, dan melepaskan kebijakan peran.

```
# Manages policies in AWS Identity and Access Management (IAM)
class RolePolicyManager
  # Initialize with an AWS IAM client
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = "PolicyManager"
  end

  # Creates a policy
  #
  # @param policy_name [String] The name of the policy
  # @param policy_document [Hash] The policy document
  # @return [String] The policy ARN if successful, otherwise nil
  def create_policy(policy_name, policy_document)
    response = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document.to_json
    )
    response.policy.arn
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating policy: #{e.message}")
    nil
  end
end
```

```
# Fetches an IAM policy by its ARN
# @param policy_arn [String] the ARN of the IAM policy to retrieve
# @return [Aws::IAM::Types::GetPolicyResponse] the policy object if found
def get_policy(policy_arn)
  response = @iam_client.get_policy(policy_arn: policy_arn)
  policy = response.policy
  @logger.info("Got policy '#{policy.policy_name}'. Its ID is:
#{policy.policy_id}.")
  policy
rescue Aws::IAM::Errors::NoSuchEntity
  @logger.error("Couldn't get policy '#{policy_arn}'. The policy does not exist.")
  raise
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Couldn't get policy '#{policy_arn}'. Here's why: #{e.code}:
#{e.message}")
  raise
end

# Attaches a policy to a role
#
# @param role_name [String] The name of the role
# @param policy_arn [String] The policy ARN
# @return [Boolean] true if successful, false otherwise
def attach_policy_to_role(role_name, policy_arn)
  @iam_client.attach_role_policy(
    role_name: role_name,
    policy_arn: policy_arn
  )
  true
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error attaching policy to role: #{e.message}")
  false
end

# Lists policy ARNs attached to a role
#
# @param role_name [String] The name of the role
# @return [Array<String>] List of policy ARNs
def list_attached_policy_arns(role_name)
  response = @iam_client.list_attached_role_policies(role_name: role_name)
  response.attached_policies.map(&:policy_arn)
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error listing policies attached to role: #{e.message}")
  []
end
```

```
end

# Detaches a policy from a role
#
# @param role_name [String] The name of the role
# @param policy_arn [String] The policy ARN
# @return [Boolean] true if successful, false otherwise
def detach_policy_from_role(role_name, policy_arn)
  @iam_client.detach_role_policy(
    role_name: role_name,
    policy_arn: policy_arn
  )
  true
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error detaching policy from role: #{e.message}")
  false
end
end
```

- Untuk API detailnya, lihat [AttachRolePolicy](#) di AWS SDK for Ruby API Referensi.

AttachUserPolicy

Contoh kode berikut menunjukkan cara menggunakan `AttachUserPolicy`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
# Attaches a policy to a user
#
# @param user_name [String] The name of the user
# @param policy_arn [String] The Amazon Resource Name (ARN) of the policy
# @return [Boolean] true if successful, false otherwise
def attach_policy_to_user(user_name, policy_arn)
  @iam_client.attach_user_policy(
    user_name: user_name,
```

```

    policy_arn: policy_arn
  )
  true
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error attaching policy to user: #{e.message}")
  false
end

```

- Untuk API detailnya, lihat [AttachUserPolicy](#) di AWS SDK for Ruby API Referensi.

CreateAccessKey

Contoh kode berikut menunjukkan cara menggunakan `CreateAccessKey`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

Modul contoh ini mencantumkan, membuat, menonaktifkan, dan menghapus kunci akses.

```

# Manages access keys for IAM users
class AccessKeyManager
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = "AccessKeyManager"
  end

  # Lists access keys for a user
  #
  # @param user_name [String] The name of the user.
  def list_access_keys(user_name)
    response = @iam_client.list_access_keys(user_name: user_name)
    if response.access_key_metadata.empty?
      @logger.info("No access keys found for user '#{user_name}'.")
    else
      response.access_key_metadata.map(&:access_key_id)
    end
  end
end

```

```
end
rescue Aws::IAM::Errors::NoSuchEntity => e
  @logger.error("Error listing access keys: cannot find user '#{user_name}'.")
  []
rescue StandardError => e
  @logger.error("Error listing access keys: #{e.message}")
  []
end

# Creates an access key for a user
#
# @param user_name [String] The name of the user.
# @return [Boolean]
def create_access_key(user_name)
  response = @iam_client.create_access_key(user_name: user_name)
  access_key = response.access_key
  @logger.info("Access key created for user '#{user_name}':
#{access_key.access_key_id}")
  access_key
rescue Aws::IAM::Errors::LimitExceeded => e
  @logger.error("Error creating access key: limit exceeded. Cannot create more.")
  nil
rescue StandardError => e
  @logger.error("Error creating access key: #{e.message}")
  nil
end

# Deactivates an access key
#
# @param user_name [String] The name of the user.
# @param access_key_id [String] The ID for the access key.
# @return [Boolean]
def deactivate_access_key(user_name, access_key_id)
  @iam_client.update_access_key(
    user_name: user_name,
    access_key_id: access_key_id,
    status: "Inactive"
  )
  true
rescue StandardError => e
  @logger.error("Error deactivating access key: #{e.message}")
  false
end
```



```
# Deletes an access key
#
# @param user_name [String] The name of the user.
# @param access_key_id [String] The ID for the access key.
# @return [Boolean]
def delete_access_key(user_name, access_key_id)
  @iam_client.delete_access_key(
    user_name: user_name,
    access_key_id: access_key_id
  )
  true
rescue StandardError => e
  @logger.error("Error deleting access key: #{e.message}")
  false
end
end
```

- Untuk API detailnya, lihat [CreateAccessKey](#) di AWS SDK for Ruby API Referensi.

CreateAccountAlias

Contoh kode berikut menunjukkan cara menggunakan `CreateAccountAlias`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

Buat daftar, buat, dan hapus alias akun.

```
class IAMAliasManager
  # Initializes the IAM client and logger
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client.
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end
end
```

```
end

# Lists available AWS account aliases.
def list_aliases
  response = @iam_client.list_account_aliases

  if response.account_aliases.count.positive?
    @logger.info("Account aliases are:")
    response.account_aliases.each { |account_alias| @logger.info("#{account_alias}") }
  else
    @logger.info("No account aliases found.")
  end
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error listing account aliases: #{e.message}")
end

# Creates an AWS account alias.
#
# @param account_alias [String] The name of the account alias to create.
# @return [Boolean] true if the account alias was created; otherwise, false.
def create_account_alias(account_alias)
  @iam_client.create_account_alias(account_alias: account_alias)
  true
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error creating account alias: #{e.message}")
  false
end

# Deletes an AWS account alias.
#
# @param account_alias [String] The name of the account alias to delete.
# @return [Boolean] true if the account alias was deleted; otherwise, false.
def delete_account_alias(account_alias)
  @iam_client.delete_account_alias(account_alias: account_alias)
  true
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error deleting account alias: #{e.message}")
  false
end
end
```

- Untuk API detailnya, lihat [CreateAccountAlias](#) di AWS SDK for Ruby API Referensi.

CreatePolicy

Contoh kode berikut menunjukkan cara menggunakan `CreatePolicy`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

Modul contoh ini mencantumkan, membuat, melampirkan, dan melepaskan kebijakan peran.

```
# Manages policies in AWS Identity and Access Management (IAM)
class RolePolicyManager
  # Initialize with an AWS IAM client
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = "PolicyManager"
  end

  # Creates a policy
  #
  # @param policy_name [String] The name of the policy
  # @param policy_document [Hash] The policy document
  # @return [String] The policy ARN if successful, otherwise nil
  def create_policy(policy_name, policy_document)
    response = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document.to_json
    )
    response.policy.arn
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating policy: #{e.message}")
    nil
  end

  # Fetches an IAM policy by its ARN
  # @param policy_arn [String] the ARN of the IAM policy to retrieve
end
```

```
# @return [Aws::IAM::Types::GetPolicyResponse] the policy object if found
def get_policy(policy_arn)
  response = @iam_client.get_policy(policy_arn: policy_arn)
  policy = response.policy
  @logger.info("Got policy '#{policy.policy_name}'. Its ID is:
#{policy.policy_id}.")
  policy
rescue Aws::IAM::Errors::NoSuchEntity
  @logger.error("Couldn't get policy '#{policy_arn}'. The policy does not exist.")
  raise
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Couldn't get policy '#{policy_arn}'. Here's why: #{e.code}:
#{e.message}")
  raise
end

# Attaches a policy to a role
#
# @param role_name [String] The name of the role
# @param policy_arn [String] The policy ARN
# @return [Boolean] true if successful, false otherwise
def attach_policy_to_role(role_name, policy_arn)
  @iam_client.attach_role_policy(
    role_name: role_name,
    policy_arn: policy_arn
  )
  true
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error attaching policy to role: #{e.message}")
  false
end

# Lists policy ARNs attached to a role
#
# @param role_name [String] The name of the role
# @return [Array<String>] List of policy ARNs
def list_attached_policy_arns(role_name)
  response = @iam_client.list_attached_role_policies(role_name: role_name)
  response.attached_policies.map(&:policy_arn)
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error listing policies attached to role: #{e.message}")
  []
end
```

```
# Detaches a policy from a role
#
# @param role_name [String] The name of the role
# @param policy_arn [String] The policy ARN
# @return [Boolean] true if successful, false otherwise
def detach_policy_from_role(role_name, policy_arn)
  @iam_client.detach_role_policy(
    role_name: role_name,
    policy_arn: policy_arn
  )
  true
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error detaching policy from role: #{e.message}")
  false
end
end
```

- Untuk API detailnya, lihat [CreatePolicy](#) di AWS SDK for Ruby API Referensi.

CreateRole

Contoh kode berikut menunjukkan cara menggunakan `CreateRole`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
# Creates a role and attaches policies to it.
#
# @param role_name [String] The name of the role.
# @param assume_role_policy_document [Hash] The trust relationship policy
document.
# @param policy_arns [Array<String>] The ARNs of the policies to attach.
# @return [String, nil] The ARN of the new role if successful, or nil if an error
occurred.
def create_role(role_name, assume_role_policy_document, policy_arns)
```

```

response = @iam_client.create_role(
  role_name: role_name,
  assume_role_policy_document: assume_role_policy_document.to_json
)
role_arn = response.role.arn

policy_arns.each do |policy_arn|
  @iam_client.attach_role_policy(
    role_name: role_name,
    policy_arn: policy_arn
  )
end

role_arn
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error creating role: #{e.message}")
  nil
end

```

- Untuk API detailnya, lihat [CreateRole](#) di AWS SDK for Ruby API Referensi.

CreateServiceLinkedRole

Contoh kode berikut menunjukkan cara menggunakan `CreateServiceLinkedRole`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```

# Creates a service-linked role
#
# @param service_name [String] The service name to create the role for.
# @param description [String] The description of the service-linked role.
# @param suffix [String] Suffix for customizing role name.
# @return [String] The name of the created role
def create_service_linked_role(service_name, description, suffix)
  response = @iam_client.create_service_linked_role(

```

```

    aws_service_name: service_name, description: description, custom_suffix:
suffix,)
    role_name = response.role.role_name
    @logger.info("Created service-linked role #{role_name}.")
    role_name
  rescue Aws::Errors::ServiceError => e
    @logger.error("Couldn't create service-linked role for #{service_name}. Here's
why:")
    @logger.error("\t#{e.code}: #{e.message}")
    raise
  end
end

```

- Untuk API detailnya, lihat [CreateServiceLinkedRole](#) di AWS SDK for Ruby API Referensi.

CreateUser

Contoh kode berikut menunjukkan cara menggunakan CreateUser.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```

# Creates a user and their login profile
#
# @param user_name [String] The name of the user
# @param initial_password [String] The initial password for the user
# @return [String, nil] The ID of the user if created, or nil if an error occurred
def create_user(user_name, initial_password)
  response = @iam_client.create_user(user_name: user_name)
  @iam_client.wait_until(:user_exists, user_name: user_name)
  @iam_client.create_login_profile(
    user_name: user_name,
    password: initial_password,
    password_reset_required: true
  )
  @logger.info("User '#{user_name}' created successfully.")
  response.user.user_id
end

```

```
rescue Aws::IAM::Errors::EntityAlreadyExists
  @logger.error("Error creating user '#{user_name}': user already exists.")
  nil
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error creating user '#{user_name}': #{e.message}")
  nil
end
```

- Untuk API detailnya, lihat [CreateUser](#) di AWS SDK for Ruby API Referensi.

DeleteAccessKey

Contoh kode berikut menunjukkan cara menggunakan `DeleteAccessKey`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

Modul contoh ini mencantumkan, membuat, menonaktifkan, dan menghapus kunci akses.

```
# Manages access keys for IAM users
class AccessKeyManager
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = "AccessKeyManager"
  end

  # Lists access keys for a user
  #
  # @param user_name [String] The name of the user.
  def list_access_keys(user_name)
    response = @iam_client.list_access_keys(user_name: user_name)
    if response.access_key_metadata.empty?
      @logger.info("No access keys found for user '#{user_name}'.")
    else
      response.access_key_metadata.map(&:access_key_id)
    end
  end
end
```



```
end
rescue Aws::IAM::Errors::NoSuchEntity => e
  @logger.error("Error listing access keys: cannot find user '#{user_name}'.")
  []
rescue StandardError => e
  @logger.error("Error listing access keys: #{e.message}")
  []
end

# Creates an access key for a user
#
# @param user_name [String] The name of the user.
# @return [Boolean]
def create_access_key(user_name)
  response = @iam_client.create_access_key(user_name: user_name)
  access_key = response.access_key
  @logger.info("Access key created for user '#{user_name}':
#{access_key.access_key_id}")
  access_key
rescue Aws::IAM::Errors::LimitExceeded => e
  @logger.error("Error creating access key: limit exceeded. Cannot create more.")
  nil
rescue StandardError => e
  @logger.error("Error creating access key: #{e.message}")
  nil
end

# Deactivates an access key
#
# @param user_name [String] The name of the user.
# @param access_key_id [String] The ID for the access key.
# @return [Boolean]
def deactivate_access_key(user_name, access_key_id)
  @iam_client.update_access_key(
    user_name: user_name,
    access_key_id: access_key_id,
    status: "Inactive"
  )
  true
rescue StandardError => e
  @logger.error("Error deactivating access key: #{e.message}")
  false
end
```

```
# Deletes an access key
#
# @param user_name [String] The name of the user.
# @param access_key_id [String] The ID for the access key.
# @return [Boolean]
def delete_access_key(user_name, access_key_id)
  @iam_client.delete_access_key(
    user_name: user_name,
    access_key_id: access_key_id
  )
  true
rescue StandardError => e
  @logger.error("Error deleting access key: #{e.message}")
  false
end
end
```

- Untuk API detailnya, lihat [DeleteAccessKey](#) di AWS SDK for Ruby API Referensi.

DeleteAccountAlias

Contoh kode berikut menunjukkan cara menggunakan `DeleteAccountAlias`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

Buat daftar, buat, dan hapus alias akun.

```
class IAMAliasManager
  # Initializes the IAM client and logger
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client.
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end
end
```

```
end

# Lists available AWS account aliases.
def list_aliases
  response = @iam_client.list_account_aliases

  if response.account_aliases.count.positive?
    @logger.info("Account aliases are:")
    response.account_aliases.each { |account_alias| @logger.info("#{account_alias}") }
  else
    @logger.info("No account aliases found.")
  end
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error listing account aliases: #{e.message}")
end

# Creates an AWS account alias.
#
# @param account_alias [String] The name of the account alias to create.
# @return [Boolean] true if the account alias was created; otherwise, false.
def create_account_alias(account_alias)
  @iam_client.create_account_alias(account_alias: account_alias)
  true
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error creating account alias: #{e.message}")
  false
end

# Deletes an AWS account alias.
#
# @param account_alias [String] The name of the account alias to delete.
# @return [Boolean] true if the account alias was deleted; otherwise, false.
def delete_account_alias(account_alias)
  @iam_client.delete_account_alias(account_alias: account_alias)
  true
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error deleting account alias: #{e.message}")
  false
end
end
```

- Untuk API detailnya, lihat [DeleteAccountAlias](#) di AWS SDK for Ruby API Referensi.

DeleteRole

Contoh kode berikut menunjukkan cara menggunakan `DeleteRole`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
# Deletes a role and its attached policies.
#
# @param role_name [String] The name of the role to delete.
def delete_role(role_name)
  begin
    # Detach and delete attached policies
    @iam_client.list_attached_role_policies(role_name: role_name).each do |
response|
      response.attached_policies.each do |policy|
        @iam_client.detach_role_policy({
          role_name: role_name,
          policy_arn: policy.policy_arn
        })
        # Check if the policy is a customer managed policy (not AWS managed)
        unless policy.policy_arn.include?("aws:policy/")
          @iam_client.delete_policy({ policy_arn: policy.policy_arn })
          @logger.info("Deleted customer managed policy #{policy.policy_name}.")
        end
      end
    end
  end

  # Delete the role
  @iam_client.delete_role({ role_name: role_name })
  @logger.info("Deleted role #{role_name}.")
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Couldn't detach policies and delete role #{role_name}. Here's
why:")
    @logger.error("\t#{e.code}: #{e.message}")
    raise
  end
end
```

- Untuk API detailnya, lihat [DeleteRole](#) di AWS SDK for Ruby API Referensi.

DeleteServerCertificate

Contoh kode berikut menunjukkan cara menggunakan `DeleteServerCertificate`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di [Repositori Contoh Kode AWS](#).

Daftar, perbarui, dan hapus sertifikat server.

```
class ServerCertificateManager
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = "ServerCertificateManager"
  end

  # Creates a new server certificate.
  # @param name [String] the name of the server certificate
  # @param certificate_body [String] the contents of the certificate
  # @param private_key [String] the private key contents
  # @return [Boolean] returns true if the certificate was successfully created
  def create_server_certificate(name, certificate_body, private_key)
    @iam_client.upload_server_certificate({
      server_certificate_name: name,
      certificate_body: certificate_body,
      private_key: private_key,
    })

    true
  rescue Aws::IAM::Errors::ServiceError => e
    puts "Failed to create server certificate: #{e.message}"
    false
  end
end
```

```
# Lists available server certificate names.
def list_server_certificate_names
  response = @iam_client.list_server_certificates

  if response.server_certificate_metadata_list.empty?
    @logger.info("No server certificates found.")
    return
  end

  response.server_certificate_metadata_list.each do |certificate_metadata|
    @logger.info("Certificate Name:
#{certificate_metadata.server_certificate_name}")
  end
end

rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error listing server certificates: #{e.message}")
end

# Updates the name of a server certificate.
def update_server_certificate_name(current_name, new_name)
  @iam_client.update_server_certificate(
    server_certificate_name: current_name,
    new_server_certificate_name: new_name
  )
  @logger.info("Server certificate name updated from '#{current_name}' to
'#{new_name}'.")
  true
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error updating server certificate name: #{e.message}")
  false
end

# Deletes a server certificate.
def delete_server_certificate(name)
  @iam_client.delete_server_certificate(server_certificate_name: name)
  @logger.info("Server certificate '#{name}' deleted.")
  true
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error deleting server certificate: #{e.message}")
  false
end
end
```

- Untuk API detailnya, lihat [DeleteServerCertificate](#) di AWS SDK for Ruby API Referensi.

DeleteServiceLinkedRole

Contoh kode berikut menunjukkan cara menggunakan `DeleteServiceLinkedRole`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
# Deletes a service-linked role.
#
# @param role_name [String] The name of the role to delete.
def delete_service_linked_role(role_name)
  response = @iam_client.delete_service_linked_role(role_name: role_name)
  task_id = response.deletion_task_id
  check_deletion_status(role_name, task_id)
rescue Aws::Errors::ServiceError => e
  handle_deletion_error(e, role_name)
end

private

# Checks the deletion status of a service-linked role
#
# @param role_name [String] The name of the role being deleted
# @param task_id [String] The task ID for the deletion process
def check_deletion_status(role_name, task_id)
  loop do
    response = @iam_client.get_service_linked_role_deletion_status(
      deletion_task_id: task_id)
    status = response.status
    @logger.info("Deletion of #{role_name} #{status}.")
    break if %w[SUCCEEDED FAILED].include?(status)
    sleep(3)
  end
end

# Handles deletion error
#
# @param e [Aws::Errors::ServiceError] The error encountered during deletion
```

```
# @param role_name [String] The name of the role attempted to delete
def handle_deletion_error(e, role_name)
  unless e.code == "NoSuchEntity"
    @logger.error("Couldn't delete #{role_name}. Here's why:")
    @logger.error("\t#{e.code}: #{e.message}")
    raise
  end
end
```

- Untuk API detailnya, lihat [DeleteServiceLinkedRole](#) di AWS SDK for Ruby API Referensi.

DeleteUser

Contoh kode berikut menunjukkan cara menggunakan `DeleteUser`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
# Deletes a user and their associated resources
#
# @param user_name [String] The name of the user to delete
def delete_user(user_name)
  user = @iam_client.list_access_keys(user_name: user_name).access_key_metadata
  user.each do |key|
    @iam_client.delete_access_key({ access_key_id: key.access_key_id, user_name:
user_name })
    @logger.info("Deleted access key #{key.access_key_id} for user
'#{user_name}'.")
  end

  @iam_client.delete_user(user_name: user_name)
  @logger.info("Deleted user '#{user_name}'.")
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error deleting user '#{user_name}': #{e.message}")
end
```


- Untuk API detailnya, lihat [DeleteUser](#) di AWS SDK for Ruby API Referensi.

DeleteUserPolicy

Contoh kode berikut menunjukkan cara menggunakan `DeleteUserPolicy`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
# Deletes a user and their associated resources
#
# @param user_name [String] The name of the user to delete
def delete_user(user_name)
  user = @iam_client.list_access_keys(user_name: user_name).access_key_metadata
  user.each do |key|
    @iam_client.delete_access_key({ access_key_id: key.access_key_id, user_name:
user_name })
    @logger.info("Deleted access key #{key.access_key_id} for user
'#{user_name}'.")
  end

  @iam_client.delete_user(user_name: user_name)
  @logger.info("Deleted user '#{user_name}'.")
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error deleting user '#{user_name}': #{e.message}")
end
```

- Untuk API detailnya, lihat [DeleteUserPolicy](#) di AWS SDK for Ruby API Referensi.

DetachRolePolicy

Contoh kode berikut menunjukkan cara menggunakan `DetachRolePolicy`.

SDK untuk Ruby

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

Modul contoh ini mencantumkan, membuat, melampirkan, dan melepaskan kebijakan peran.

```
# Manages policies in AWS Identity and Access Management (IAM)
class RolePolicyManager
  # Initialize with an AWS IAM client
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = "PolicyManager"
  end

  # Creates a policy
  #
  # @param policy_name [String] The name of the policy
  # @param policy_document [Hash] The policy document
  # @return [String] The policy ARN if successful, otherwise nil
  def create_policy(policy_name, policy_document)
    response = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document.to_json
    )
    response.policy.arn
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating policy: #{e.message}")
    nil
  end

  # Fetches an IAM policy by its ARN
  # @param policy_arn [String] the ARN of the IAM policy to retrieve
  # @return [Aws::IAM::Types::GetPolicyResponse] the policy object if found
  def get_policy(policy_arn)
    response = @iam_client.get_policy(policy_arn: policy_arn)
    policy = response.policy
  end
end
```

```
@logger.info("Got policy '#{policy.policy_name}'. Its ID is:
#{policy.policy_id}.")
  policy
rescue Aws::IAM::Errors::NoSuchEntity
  @logger.error("Couldn't get policy '#{policy_arn}'. The policy does not exist.")
  raise
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Couldn't get policy '#{policy_arn}'. Here's why: #{e.code}:
#{e.message}")
  raise
end

# Attaches a policy to a role
#
# @param role_name [String] The name of the role
# @param policy_arn [String] The policy ARN
# @return [Boolean] true if successful, false otherwise
def attach_policy_to_role(role_name, policy_arn)
  @iam_client.attach_role_policy(
    role_name: role_name,
    policy_arn: policy_arn
  )
  true
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error attaching policy to role: #{e.message}")
  false
end

# Lists policy ARNs attached to a role
#
# @param role_name [String] The name of the role
# @return [Array<String>] List of policy ARNs
def list_attached_policy_arns(role_name)
  response = @iam_client.list_attached_role_policies(role_name: role_name)
  response.attached_policies.map(&:policy_arn)
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error listing policies attached to role: #{e.message}")
  []
end

# Detaches a policy from a role
#
# @param role_name [String] The name of the role
# @param policy_arn [String] The policy ARN
```

```
# @return [Boolean] true if successful, false otherwise
def detach_policy_from_role(role_name, policy_arn)
  @iam_client.detach_role_policy(
    role_name: role_name,
    policy_arn: policy_arn
  )
  true
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error detaching policy from role: #{e.message}")
  false
end
end
```

- Untuk API detailnya, lihat [DetachRolePolicy](#) di AWS SDK for Ruby API Referensi.

DetachUserPolicy

Contoh kode berikut menunjukkan cara menggunakan `DetachUserPolicy`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
# Detaches a policy from a user
#
# @param user_name [String] The name of the user
# @param policy_arn [String] The ARN of the policy to detach
# @return [Boolean] true if the policy was successfully detached, false otherwise
def detach_user_policy(user_name, policy_arn)
  @iam_client.detach_user_policy(
    user_name: user_name,
    policy_arn: policy_arn
  )
  @logger.info("Policy '#{policy_arn}' detached from user '#{user_name}'
successfully.")
  true
end
```

```
rescue Aws::IAM::Errors::NoSuchEntity
  @logger.error("Error detaching policy: Policy or user does not exist.")
  false
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error detaching policy from user '#{user_name}': #{e.message}")
  false
end
```

- Untuk API detailnya, lihat [DetachUserPolicy](#) di AWS SDK for Ruby API Referensi.

GetAccountPasswordPolicy

Contoh kode berikut menunjukkan cara menggunakan `GetAccountPasswordPolicy`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
# Class to manage IAM account password policies
class PasswordPolicyManager
  attr_accessor :iam_client, :logger

  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = "IAMPolicyManager"
  end

  # Retrieves and logs the account password policy
  def print_account_password_policy
    begin
      response = @iam_client.get_account_password_policy
      @logger.info("The account password policy is:
#{response.password_policy.to_h}")
    rescue Aws::IAM::Errors::NoSuchEntity
      @logger.info("The account does not have a password policy.")
    end
  end
end
```

```

    rescue Aws::Errors::ServiceError => e
      @logger.error("Couldn't print the account password policy. Error: #{e.code} -
#{e.message}")
      raise
    end
  end
end
end

```

- Untuk API detailnya, lihat [GetAccountPasswordPolicy](#) di AWS SDK for Ruby API Referensi.

GetPolicy

Contoh kode berikut menunjukkan cara menggunakan `GetPolicy`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```

# Fetches an IAM policy by its ARN
# @param policy_arn [String] the ARN of the IAM policy to retrieve
# @return [Aws::IAM::Types::GetPolicyResponse] the policy object if found
def get_policy(policy_arn)
  response = @iam_client.get_policy(policy_arn: policy_arn)
  policy = response.policy
  @logger.info("Got policy '#{policy.policy_name}'. Its ID is:
#{policy.policy_id}.")
  policy
rescue Aws::IAM::Errors::NoSuchEntity
  @logger.error("Couldn't get policy '#{policy_arn}'. The policy does not exist.")
  raise
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Couldn't get policy '#{policy_arn}'. Here's why: #{e.code}:
#{e.message}")
  raise
end

```

- Untuk API detailnya, lihat [GetPolicy](#) di AWS SDK for Ruby API Referensi.

GetRole

Contoh kode berikut menunjukkan cara menggunakan `GetRole`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
# Gets data about a role.
#
# @param name [String] The name of the role to look up.
# @return [Aws::IAM::Role] The retrieved role.
def get_role(name)
  role = @iam_client.get_role({
    role_name: name,
  }).role

  puts("Got data for role '#{role.role_name}'. Its ARN is '#{role.arn}'.")
rescue Aws::Errors::ServiceError => e
  puts("Couldn't get data for role '#{name}' Here's why:")
  puts("\t#{e.code}: #{e.message}")
  raise
else
  role
end
```

- Untuk API detailnya, lihat [GetRole](#) di AWS SDK for Ruby API Referensi.

GetUser

Contoh kode berikut menunjukkan cara menggunakan `GetUser`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
# Retrieves a user's details
#
# @param user_name [String] The name of the user to retrieve
# @return [Aws::IAM::Types::User, nil] The user object if found, or nil if an
error occurred
def get_user(user_name)
  response = @iam_client.get_user(user_name: user_name)
  response.user
rescue Aws::IAM::Errors::NoSuchEntity
  @logger.error("User '#{user_name}' not found.")
  nil
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error retrieving user '#{user_name}': #{e.message}")
  nil
end
```

- Untuk API detailnya, lihat [GetUser](#) di AWS SDK for Ruby API Referensi.

ListAccessKeys

Contoh kode berikut menunjukkan cara menggunakan `ListAccessKeys`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

Modul contoh ini mencantumkan, membuat, menonaktifkan, dan menghapus kunci akses.


```
# Manages access keys for IAM users
class AccessKeyManager
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = "AccessKeyManager"
  end

  # Lists access keys for a user
  #
  # @param user_name [String] The name of the user.
  def list_access_keys(user_name)
    response = @iam_client.list_access_keys(user_name: user_name)
    if response.access_key_metadata.empty?
      @logger.info("No access keys found for user '#{user_name}'.")
    else
      response.access_key_metadata.map(&:access_key_id)
    end
  rescue Aws::IAM::Errors::NoSuchEntity => e
    @logger.error("Error listing access keys: cannot find user '#{user_name}'.")
    []
  rescue StandardError => e
    @logger.error("Error listing access keys: #{e.message}")
    []
  end

  # Creates an access key for a user
  #
  # @param user_name [String] The name of the user.
  # @return [Boolean]
  def create_access_key(user_name)
    response = @iam_client.create_access_key(user_name: user_name)
    access_key = response.access_key
    @logger.info("Access key created for user '#{user_name}':
    #{access_key.access_key_id}")
    access_key
  rescue Aws::IAM::Errors::LimitExceeded => e
    @logger.error("Error creating access key: limit exceeded. Cannot create more.")
    nil
  rescue StandardError => e
    @logger.error("Error creating access key: #{e.message}")
    nil
  end
end
```

```
# Deactivates an access key
#
# @param user_name [String] The name of the user.
# @param access_key_id [String] The ID for the access key.
# @return [Boolean]
def deactivate_access_key(user_name, access_key_id)
  @iam_client.update_access_key(
    user_name: user_name,
    access_key_id: access_key_id,
    status: "Inactive"
  )
  true
rescue StandardError => e
  @logger.error("Error deactivating access key: #{e.message}")
  false
end

# Deletes an access key
#
# @param user_name [String] The name of the user.
# @param access_key_id [String] The ID for the access key.
# @return [Boolean]
def delete_access_key(user_name, access_key_id)
  @iam_client.delete_access_key(
    user_name: user_name,
    access_key_id: access_key_id
  )
  true
rescue StandardError => e
  @logger.error("Error deleting access key: #{e.message}")
  false
end
end
```

- Untuk API detailnya, lihat [ListAccessKeys](#) di AWS SDK for Ruby API Referensi.

ListAccountAliases

Contoh kode berikut menunjukkan cara menggunakan `ListAccountAliases`.

SDK untuk Ruby

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di [Repositori Contoh Kode AWS](#).

Buat daftar, buat, dan hapus alias akun.

```
class IAMAliasManager
  # Initializes the IAM client and logger
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client.
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end

  # Lists available AWS account aliases.
  def list_aliases
    response = @iam_client.list_account_aliases

    if response.account_aliases.count.positive?
      @logger.info("Account aliases are:")
      response.account_aliases.each { |account_alias| @logger.info("#{account_alias}") }
    else
      @logger.info("No account aliases found.")
    end
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error listing account aliases: #{e.message}")
  end

  # Creates an AWS account alias.
  #
  # @param account_alias [String] The name of the account alias to create.
  # @return [Boolean] true if the account alias was created; otherwise, false.
  def create_account_alias(account_alias)
    @iam_client.create_account_alias(account_alias: account_alias)
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating account alias: #{e.message}")
  end
end
```

```

    false
  end

  # Deletes an AWS account alias.
  #
  # @param account_alias [String] The name of the account alias to delete.
  # @return [Boolean] true if the account alias was deleted; otherwise, false.
  def delete_account_alias(account_alias)
    @iam_client.delete_account_alias(account_alias: account_alias)
    true
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error deleting account alias: #{e.message}")
    false
  end
end
end

```

- Untuk API detailnya, lihat [ListAccountAliases](#) di AWS SDK for Ruby API Referensi.

ListAttachedRolePolicies

Contoh kode berikut menunjukkan cara menggunakan `ListAttachedRolePolicies`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

Modul contoh ini mencantumkan, membuat, melampirkan, dan melepaskan kebijakan peran.

```

# Manages policies in AWS Identity and Access Management (IAM)
class RolePolicyManager
  # Initialize with an AWS IAM client
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = "PolicyManager"
  end
end

```

```
end

# Creates a policy
#
# @param policy_name [String] The name of the policy
# @param policy_document [Hash] The policy document
# @return [String] The policy ARN if successful, otherwise nil
def create_policy(policy_name, policy_document)
  response = @iam_client.create_policy(
    policy_name: policy_name,
    policy_document: policy_document.to_json
  )
  response.policy.arn
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error creating policy: #{e.message}")
  nil
end

# Fetches an IAM policy by its ARN
# @param policy_arn [String] the ARN of the IAM policy to retrieve
# @return [Aws::IAM::Types::GetPolicyResponse] the policy object if found
def get_policy(policy_arn)
  response = @iam_client.get_policy(policy_arn: policy_arn)
  policy = response.policy
  @logger.info("Got policy '#{policy.policy_name}'. Its ID is:
#{policy.policy_id}.")
  policy
rescue Aws::IAM::Errors::NoSuchEntity
  @logger.error("Couldn't get policy '#{policy_arn}'. The policy does not exist.")
  raise
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Couldn't get policy '#{policy_arn}'. Here's why: #{e.code}:
#{e.message}")
  raise
end

# Attaches a policy to a role
#
# @param role_name [String] The name of the role
# @param policy_arn [String] The policy ARN
# @return [Boolean] true if successful, false otherwise
def attach_policy_to_role(role_name, policy_arn)
  @iam_client.attach_role_policy(
    role_name: role_name,
```

```

    policy_arn: policy_arn
  )
  true
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error attaching policy to role: #{e.message}")
  false
end

# Lists policy ARNs attached to a role
#
# @param role_name [String] The name of the role
# @return [Array<String>] List of policy ARNs
def list_attached_policy_arns(role_name)
  response = @iam_client.list_attached_role_policies(role_name: role_name)
  response.attached_policies.map(&:policy_arn)
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error listing policies attached to role: #{e.message}")
  []
end

# Detaches a policy from a role
#
# @param role_name [String] The name of the role
# @param policy_arn [String] The policy ARN
# @return [Boolean] true if successful, false otherwise
def detach_policy_from_role(role_name, policy_arn)
  @iam_client.detach_role_policy(
    role_name: role_name,
    policy_arn: policy_arn
  )
  true
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error detaching policy from role: #{e.message}")
  false
end
end
end

```

- Untuk API detailnya, lihat [ListAttachedRolePolicies](#) di AWS SDK for Ruby API Referensi.

ListGroups

Contoh kode berikut menunjukkan cara menggunakan `ListGroups`.

SDK untuk Ruby

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
# A class to manage IAM operations via the AWS SDK client
class IamGroupManager
  # Initializes the IamGroupManager class
  # @param iam_client [Aws::IAM::Client] An instance of the IAM client
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end

  # Lists up to a specified number of groups for the account.
  # @param count [Integer] The maximum number of groups to list.
  # @return [Aws::IAM::Client::Response]
  def list_groups(count)
    response = @iam_client.list_groups(max_items: count)
    response.groups.each do |group|
      @logger.info("\t#{group.group_name}")
    end
    response
  rescue Aws::Errors::ServiceError => e
    @logger.error("Couldn't list groups for the account. Here's why:")
    @logger.error("\t#{e.code}: #{e.message}")
    raise
  end
end
```

- Untuk API detailnya, lihat [ListGroups](#) di AWS SDK for Ruby API Referensi.

List Policies

Contoh kode berikut menunjukkan cara menggunakan `ListPolicies`.

SDK untuk Ruby

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

Modul contoh ini mencantumkan, membuat, melampirkan, dan melepaskan kebijakan peran.

```
# Manages policies in AWS Identity and Access Management (IAM)
class RolePolicyManager
  # Initialize with an AWS IAM client
  #
  # @param iam_client [Aws::IAM::Client] An initialized IAM client
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = "PolicyManager"
  end

  # Creates a policy
  #
  # @param policy_name [String] The name of the policy
  # @param policy_document [Hash] The policy document
  # @return [String] The policy ARN if successful, otherwise nil
  def create_policy(policy_name, policy_document)
    response = @iam_client.create_policy(
      policy_name: policy_name,
      policy_document: policy_document.to_json
    )
    response.policy.arn
  rescue Aws::IAM::Errors::ServiceError => e
    @logger.error("Error creating policy: #{e.message}")
    nil
  end

  # Fetches an IAM policy by its ARN
  # @param policy_arn [String] the ARN of the IAM policy to retrieve
  # @return [Aws::IAM::Types::GetPolicyResponse] the policy object if found
  def get_policy(policy_arn)
    response = @iam_client.get_policy(policy_arn: policy_arn)
    policy = response.policy
  end
end
```



```
@logger.info("Got policy '#{policy.policy_name}'. Its ID is:
#{policy.policy_id}.")
  policy
rescue Aws::IAM::Errors::NoSuchEntity
  @logger.error("Couldn't get policy '#{policy_arn}'. The policy does not exist.")
  raise
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Couldn't get policy '#{policy_arn}'. Here's why: #{e.code}:
#{e.message}")
  raise
end

# Attaches a policy to a role
#
# @param role_name [String] The name of the role
# @param policy_arn [String] The policy ARN
# @return [Boolean] true if successful, false otherwise
def attach_policy_to_role(role_name, policy_arn)
  @iam_client.attach_role_policy(
    role_name: role_name,
    policy_arn: policy_arn
  )
  true
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error attaching policy to role: #{e.message}")
  false
end

# Lists policy ARNs attached to a role
#
# @param role_name [String] The name of the role
# @return [Array<String>] List of policy ARNs
def list_attached_policy_arns(role_name)
  response = @iam_client.list_attached_role_policies(role_name: role_name)
  response.attached_policies.map(&:policy_arn)
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error listing policies attached to role: #{e.message}")
  []
end

# Detaches a policy from a role
#
# @param role_name [String] The name of the role
# @param policy_arn [String] The policy ARN
```

```
# @return [Boolean] true if successful, false otherwise
def detach_policy_from_role(role_name, policy_arn)
  @iam_client.detach_role_policy(
    role_name: role_name,
    policy_arn: policy_arn
  )
  true
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error detaching policy from role: #{e.message}")
  false
end
end
```

- Untuk API detailnya, lihat [ListPolicies](#) di AWS SDK for Ruby API Referensi.

ListRolePolicies

Contoh kode berikut menunjukkan cara menggunakan `ListRolePolicies`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
# Lists policy ARNs attached to a role
#
# @param role_name [String] The name of the role
# @return [Array<String>] List of policy ARNs
def list_attached_policy_arns(role_name)
  response = @iam_client.list_attached_role_policies(role_name: role_name)
  response.attached_policies.map(&:policy_arn)
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error listing policies attached to role: #{e.message}")
  []
end
```

- Untuk API detailnya, lihat [ListRolePolicies](#) di AWS SDK for Ruby API Referensi.

ListRoles

Contoh kode berikut menunjukkan cara menggunakan `ListRoles`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
# Lists IAM roles up to a specified count.
# @param count [Integer] the maximum number of roles to list.
# @return [Array<String>] the names of the roles.
def list_roles(count)
  role_names = []
  roles_counted = 0

  @iam_client.list_roles.each_page do |page|
    page.roles.each do |role|
      break if roles_counted >= count
      @logger.info("\t#{roles_counted + 1}: #{role.role_name}")
      role_names << role.role_name
      roles_counted += 1
    end
    break if roles_counted >= count
  end

  role_names
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Couldn't list roles for the account. Here's why:")
  @logger.error("\t#{e.code}: #{e.message}")
  raise
end
```

- Untuk API detailnya, lihat [ListRoles](#) di AWS SDK for Ruby API Referensi.

ListSAMLProviders

Contoh kode berikut menunjukkan cara menggunakan `ListSAMLProviders`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
class SAMLProviderLister
  # Initializes the SAMLProviderLister with IAM client and a logger.
  # @param iam_client [Aws::IAM::Client] The IAM client object.
  # @param logger [Logger] The logger object for logging output.
  def initialize(iam_client, logger = Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
  end

  # Lists up to a specified number of SAML providers for the account.
  # @param count [Integer] The maximum number of providers to list.
  # @return [Aws::IAM::Client::Response]
  def list_saml_providers(count)
    response = @iam_client.list_saml_providers
    response.saml_provider_list.take(count).each do |provider|
      @logger.info("\t#{provider.arn}")
    end
    response
  rescue Aws::Errors::ServiceError => e
    @logger.error("Couldn't list SAML providers. Here's why:")
    @logger.error("\t#{e.code}: #{e.message}")
    raise
  end
end
```

- Untuk API detailnya, lihat [ListSAMLProviders](#) di AWS SDK for Ruby API Referensi.

ListServerCertificates

Contoh kode berikut menunjukkan cara menggunakan `ListServerCertificates`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

Daftar, perbarui, dan hapus sertifikat server.

```
class ServerCertificateManager
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = "ServerCertificateManager"
  end

  # Creates a new server certificate.
  # @param name [String] the name of the server certificate
  # @param certificate_body [String] the contents of the certificate
  # @param private_key [String] the private key contents
  # @return [Boolean] returns true if the certificate was successfully created
  def create_server_certificate(name, certificate_body, private_key)
    @iam_client.upload_server_certificate({
      server_certificate_name: name,
      certificate_body: certificate_body,
      private_key: private_key,
    })

    true
  rescue Aws::IAM::Errors::ServiceError => e
    puts "Failed to create server certificate: #{e.message}"
    false
  end

  # Lists available server certificate names.
  def list_server_certificate_names
    response = @iam_client.list_server_certificates

    if response.server_certificate_metadata_list.empty?
```

```
@logger.info("No server certificates found.")
return
end

response.server_certificate_metadata_list.each do |certificate_metadata|
  @logger.info("Certificate Name:
#{certificate_metadata.server_certificate_name}")
  end
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error listing server certificates: #{e.message}")
end

# Updates the name of a server certificate.
def update_server_certificate_name(current_name, new_name)
  @iam_client.update_server_certificate(
    server_certificate_name: current_name,
    new_server_certificate_name: new_name
  )
  @logger.info("Server certificate name updated from '#{current_name}' to
 '#{new_name}'.")
  true
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error updating server certificate name: #{e.message}")
  false
end

# Deletes a server certificate.
def delete_server_certificate(name)
  @iam_client.delete_server_certificate(server_certificate_name: name)
  @logger.info("Server certificate '#{name}' deleted.")
  true
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error deleting server certificate: #{e.message}")
  false
end
end
```

- Untuk API detailnya, lihat [ListServerCertificates](#) di AWS SDK for Ruby API Referensi.

ListUsers

Contoh kode berikut menunjukkan cara menggunakan `ListUsers`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
# Lists all users in the AWS account
#
# @return [Array<Aws::IAM::Types::User>] An array of user objects
def list_users
  users = []
  @iam_client.list_users.each_page do |page|
    page.users.each do |user|
      users << user
    end
  end
  users
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error listing users: #{e.message}")
  []
end
```

- Untuk API detailnya, lihat [ListUsers](#) di AWS SDK for Ruby API Referensi.

PutUserPolicy

Contoh kode berikut menunjukkan cara menggunakan `PutUserPolicy`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
# Creates an inline policy for a specified user.
```

```
# @param username [String] The name of the IAM user.
# @param policy_name [String] The name of the policy to create.
# @param policy_document [String] The JSON policy document.
# @return [Boolean]
def create_user_policy(username, policy_name, policy_document)
  @iam_client.put_user_policy({
    user_name: username,
    policy_name: policy_name,
    policy_document: policy_document
  })
  @logger.info("Policy #{policy_name} created for user #{username}.")
  true
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Couldn't create policy #{policy_name} for user #{username}.
Here's why:")
  @logger.error("\t#{e.code}: #{e.message}")
  false
end
```

- Untuk API detailnya, lihat [PutUserPolicy](#) di AWS SDK for Ruby API Referensi.

UpdateServerCertificate

Contoh kode berikut menunjukkan cara menggunakan `UpdateServerCertificate`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

Daftar, perbarui, dan hapus sertifikat server.

```
class ServerCertificateManager
  def initialize(iam_client, logger: Logger.new($stdout))
    @iam_client = iam_client
    @logger = logger
    @logger.progname = "ServerCertificateManager"
  end
end
```



```
# Creates a new server certificate.
# @param name [String] the name of the server certificate
# @param certificate_body [String] the contents of the certificate
# @param private_key [String] the private key contents
# @return [Boolean] returns true if the certificate was successfully created
def create_server_certificate(name, certificate_body, private_key)
  @iam_client.upload_server_certificate({
    server_certificate_name: name,
    certificate_body: certificate_body,
    private_key: private_key,
  })

  true
rescue Aws::IAM::Errors::ServiceError => e
  puts "Failed to create server certificate: #{e.message}"
  false
end

# Lists available server certificate names.
def list_server_certificate_names
  response = @iam_client.list_server_certificates

  if response.server_certificate_metadata_list.empty?
    @logger.info("No server certificates found.")
    return
  end

  response.server_certificate_metadata_list.each do |certificate_metadata|
    @logger.info("Certificate Name:
#{certificate_metadata.server_certificate_name}")
  end
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error listing server certificates: #{e.message}")
end

# Updates the name of a server certificate.
def update_server_certificate_name(current_name, new_name)
  @iam_client.update_server_certificate(
    server_certificate_name: current_name,
    new_server_certificate_name: new_name
  )
  @logger.info("Server certificate name updated from '#{current_name}' to
 '#{new_name}'.")
  true
end
```

```
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error updating server certificate name: #{e.message}")
  false
end

# Deletes a server certificate.
def delete_server_certificate(name)
  @iam_client.delete_server_certificate(server_certificate_name: name)
  @logger.info("Server certificate '#{name}' deleted.")
  true
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error deleting server certificate: #{e.message}")
  false
end
end
```

- Untuk API detailnya, lihat [UpdateServerCertificate](#) di AWS SDK for Ruby API Referensi.

UpdateUser

Contoh kode berikut menunjukkan cara menggunakan `UpdateUser`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
# Updates an IAM user's name
#
# @param current_name [String] The current name of the user
# @param new_name [String] The new name of the user
def update_user_name(current_name, new_name)
  @iam_client.update_user(user_name: current_name, new_user_name: new_name)
  true
rescue StandardError => e
  @logger.error("Error updating user name from '#{current_name}' to '#{new_name}':
  #{e.message}")
  false
end
```

```
end
```

- Untuk API detailnya, lihat [UpdateUser](#) di AWS SDK for Ruby API Referensi.

Skenario

Buat pengguna dan ambil peran

Contoh kode berikut menunjukkan cara membuat pengguna dan mengambil peran.

Warning

Untuk menghindari risiko keamanan, jangan gunakan IAM pengguna untuk otentikasi saat mengembangkan perangkat lunak yang dibuat khusus atau bekerja dengan data nyata. Sebaliknya, gunakan federasi dengan penyedia identitas seperti [AWS IAM Identity Center](#).

- Buat pengguna tanpa izin.
- Buat peran yang memberikan izin untuk mencantumkan bucket Amazon S3 untuk akun tersebut.
- Tambahkan kebijakan agar pengguna dapat mengambil peran tersebut.
- Asumsikan peran dan daftar bucket S3 menggunakan kredensial sementara, lalu bersihkan sumber daya.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di [Repositori Contoh Kode AWS](#).

Buat IAM pengguna dan peran yang memberikan izin untuk mencantumkan bucket Amazon S3. Pengguna hanya memiliki hak untuk mengambil peran. Setelah mengambil peran, gunakan kredensial sementara untuk membuat daftar bucket untuk akun.

```
# Wraps the scenario actions.  
class ScenarioCreateUserAssumeRole
```

```
attr_reader :iam_client

# @param [Aws::IAM::Client] iam_client: The AWS IAM client.
def initialize(iam_client, logger: Logger.new($stdout))
  @iam_client = iam_client
  @logger = logger
end

# Waits for the specified number of seconds.
#
# @param duration [Integer] The number of seconds to wait.
def wait(duration)
  puts("Give AWS time to propagate resources...")
  sleep(duration)
end

# Creates a user.
#
# @param user_name [String] The name to give the user.
# @return [Aws::IAM::User] The newly created user.
def create_user(user_name)
  user = @iam_client.create_user(user_name: user_name).user
  @logger.info("Created demo user named #{user.user_name}.")
rescue Aws::Errors::ServiceError => e
  @logger.info("Tried and failed to create demo user.")
  @logger.info("\t#{e.code}: #{e.message}")
  @logger.info("\nCan't continue the demo without a user!")
  raise
else
  user
end

# Creates an access key for a user.
#
# @param user [Aws::IAM::User] The user that owns the key.
# @return [Aws::IAM::AccessKeyPair] The newly created access key.
def create_access_key_pair(user)
  user_key = @iam_client.create_access_key(user_name: user.user_name).access_key
  @logger.info("Created accesskey pair for user #{user.user_name}.")
rescue Aws::Errors::ServiceError => e
  @logger.info("Couldn't create access keys for user #{user.user_name}.")
  @logger.info("\t#{e.code}: #{e.message}")
  raise
else
  user_key
end
```

```
    user_key
  end

  # Creates a role that can be assumed by a user.
  #
  # @param role_name [String] The name to give the role.
  # @param user [Aws::IAM::User] The user who is granted permission to assume the
  role.
  # @return [Aws::IAM::Role] The newly created role.
  def create_role(role_name, user)
    trust_policy = {
      Version: "2012-10-17",
      Statement: [{
        Effect: "Allow",
        Principal: {'AWS': user.arn},
        Action: "sts:AssumeRole"
      }]
    }.to_json
    role = @iam_client.create_role(
      role_name: role_name,
      assume_role_policy_document: trust_policy
    ).role
    @logger.info("Created role #{role.role_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't create a role for the demo. Here's why: ")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  else
    role
  end

  # Creates a policy that grants permission to list S3 buckets in the account, and
  # then attaches the policy to a role.
  #
  # @param policy_name [String] The name to give the policy.
  # @param role [Aws::IAM::Role] The role that the policy is attached to.
  # @return [Aws::IAM::Policy] The newly created policy.
  def create_and_attach_role_policy(policy_name, role)
    policy_document = {
      Version: "2012-10-17",
      Statement: [{
        Effect: "Allow",
        Action: "s3:ListAllMyBuckets",
        Resource: "arn:aws:s3:::*"
      }]
    }
```

```
    ]]
  }.to_json
  policy = @iam_client.create_policy(
    policy_name: policy_name,
    policy_document: policy_document
  ).policy
  @iam_client.attach_role_policy(
    role_name: role.role_name,
    policy_arn: policy.arn
  )
  @logger.info("Created policy #{policy.policy_name} and attached it to role
  #{role.role_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't create a policy and attach it to role #{role.role_name}.
  Here's why: ")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  end

# Creates an inline policy for a user that lets the user assume a role.
#
# @param policy_name [String] The name to give the policy.
# @param user [Aws::IAM::User] The user that owns the policy.
# @param role [Aws::IAM::Role] The role that can be assumed.
# @return [Aws::IAM::UserPolicy] The newly created policy.
def create_user_policy(policy_name, user, role)
  policy_document = {
    Version: "2012-10-17",
    Statement: [{
      Effect: "Allow",
      Action: "sts:AssumeRole",
      Resource: role.arn
    }]
  }.to_json
  @iam_client.put_user_policy(
    user_name: user.user_name,
    policy_name: policy_name,
    policy_document: policy_document
  )
  puts("Created an inline policy for #{user.user_name} that lets the user assume
  role #{role.role_name}.")
  rescue Aws::Errors::ServiceError => e
    @logger.info("Couldn't create an inline policy for user #{user.user_name}.
  Here's why: ")
```

```
@logger.info("\t#{e.code}: #{e.message}")
  raise
end

# Creates an Amazon S3 resource with specified credentials. This is separated into
a
# factory function so that it can be mocked for unit testing.
#
# @param credentials [Aws::Credentials] The credentials used by the Amazon S3
resource.
def create_s3_resource(credentials)
  Aws::S3::Resource.new(client: Aws::S3::Client.new(credentials: credentials))
end

# Lists the S3 buckets for the account, using the specified Amazon S3 resource.
# Because the resource uses credentials with limited access, it may not be able to
# list the S3 buckets.
#
# @param s3_resource [Aws::S3::Resource] An Amazon S3 resource.
def list_buckets(s3_resource)
  count = 10
  s3_resource.buckets.each do |bucket|
    @logger.info "\t#{bucket.name}"
    count -= 1
    break if count.zero?
  end
rescue Aws::Errors::ServiceError => e
  if e.code == "AccessDenied"
    puts("Attempt to list buckets with no permissions: AccessDenied.")
  else
    @logger.info("Couldn't list buckets for the account. Here's why: ")
    @logger.info("\t#{e.code}: #{e.message}")
    raise
  end
end

# Creates an AWS Security Token Service (AWS STS) client with specified
credentials.
# This is separated into a factory function so that it can be mocked for unit
testing.
#
# @param key_id [String] The ID of the access key used by the STS client.
# @param key_secret [String] The secret part of the access key used by the STS
client.
```

```
def create_sts_client(key_id, key_secret)
  Aws::STS::Client.new(access_key_id: key_id, secret_access_key: key_secret)
end

# Gets temporary credentials that can be used to assume a role.
#
# @param role_arn [String] The ARN of the role that is assumed when these
credentials
#
# are used.
# @param sts_client [AWS::STS::Client] An AWS STS client.
# @return [Aws::AssumeRoleCredentials] The credentials that can be used to assume
the role.
def assume_role(role_arn, sts_client)
  credentials = Aws::AssumeRoleCredentials.new(
    client: sts_client,
    role_arn: role_arn,
    role_session_name: "create-use-assume-role-scenario"
  )
  @logger.info("Assumed role '#{role_arn}', got temporary credentials.")
  credentials
end

# Deletes a role. If the role has policies attached, they are detached and
# deleted before the role is deleted.
#
# @param role_name [String] The name of the role to delete.
def delete_role(role_name)
  @iam_client.list_attached_role_policies(role_name:
role_name).attached_policies.each do |policy|
    @iam_client.detach_role_policy(role_name: role_name, policy_arn:
policy.policy_arn)
    @iam_client.delete_policy(policy_arn: policy.policy_arn)
    @logger.info("Detached and deleted policy #{policy.policy_name}.")
  end
  @iam_client.delete_role({ role_name: role_name })
  @logger.info("Role deleted: #{role_name}.")
rescue Aws::Errors::ServiceError => e
  @logger.info("Couldn't detach policies and delete role #{role.name}. Here's
why:")
  @logger.info("\t#{e.code}: #{e.message}")
  raise
end

# Deletes a user. If the user has inline policies or access keys, they are deleted
```



```
# before the user is deleted.
#
# @param user [Aws::IAM::User] The user to delete.
def delete_user(user_name)
  user = @iam_client.list_access_keys(user_name: user_name).access_key_metadata
  user.each do |key|
    @iam_client.delete_access_key({ access_key_id: key.access_key_id, user_name:
user_name })
    @logger.info("Deleted access key #{key.access_key_id} for user
'#{user_name}'.")
  end

  @iam_client.delete_user(user_name: user_name)
  @logger.info("Deleted user '#{user_name}'.")
rescue Aws::IAM::Errors::ServiceError => e
  @logger.error("Error deleting user '#{user_name}': #{e.message}")
end
end

# Runs the IAM create a user and assume a role scenario.
def run_scenario(scenario)
  puts("-" * 88)
  puts("Welcome to the IAM create a user and assume a role demo!")
  puts("-" * 88)
  user = scenario.create_user("doc-example-user-#{Random.uuid}")
  user_key = scenario.create_access_key_pair(user)
  scenario.wait(10)
  role = scenario.create_role("doc-example-role-#{Random.uuid}", user)
  scenario.create_and_attach_role_policy("doc-example-role-policy-#{Random.uuid}",
role)
  scenario.create_user_policy("doc-example-user-policy-#{Random.uuid}", user, role)
  scenario.wait(10)
  puts("Try to list buckets with credentials for a user who has no permissions.")
  puts("Expect AccessDenied from this call.")
  scenario.list_buckets(
    scenario.create_s3_resource(Aws::Credentials.new(user_key.access_key_id,
user_key.secret_access_key)))
  puts("Now, assume the role that grants permission.")
  temp_credentials = scenario.assume_role(
    role.arn, scenario.create_sts_client(user_key.access_key_id,
user_key.secret_access_key))
  puts("Here are your buckets:")
  scenario.list_buckets(scenario.create_s3_resource(temp_credentials))
  puts("Deleting role '#{role.role_name}' and attached policies.")
end
end
```

```
scenario.delete_role(role.role_name)
puts("Deleting user '#{user.user_name}', policies, and keys.")
scenario.delete_user(user.user_name)
puts("Thanks for watching!")
puts("-" * 88)
rescue Aws::Errors::ServiceError => e
  puts("Something went wrong with the demo.")
  puts("\t#{e.code}: #{e.message}")
end

run_scenario(ScenarioCreateUserAssumeRole.new(Aws::IAM::Client.new)) if
$PROGRAM_NAME == __FILE__
```

- Untuk API detailnya, lihat topik berikut di [AWS SDK for Ruby API Referensi](#).
 - [AttachRolePolicy](#)
 - [CreateAccessKey](#)
 - [CreatePolicy](#)
 - [CreateRole](#)
 - [CreateUser](#)
 - [DeleteAccessKey](#)
 - [DeletePolicy](#)
 - [DeleteRole](#)
 - [DeleteUser](#)
 - [DeleteUserPolicy](#)
 - [DetachRolePolicy](#)
 - [PutUserPolicy](#)

Contoh Kinesis yang digunakan SDK untuk Ruby

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan AWS SDK for Ruby Kinesis with.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Contoh nirserver](#)

Contoh nirserver

Memanggil fungsi Lambda dari pemicu Kinesis

Contoh kode berikut menunjukkan bagaimana menerapkan fungsi Lambda yang menerima peristiwa yang dipicu oleh menerima catatan dari aliran Kinesis. Fungsi mengambil payload Kinesis, mendekode dari Base64, dan mencatat konten rekaman.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di repositori [contoh Nirserver](#).

Mengonsumsi acara Kinesis dengan Lambda menggunakan Ruby.

```
# Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
# SPDX-License-Identifier: Apache-2.0
require 'aws-sdk'

def lambda_handler(event:, context:)
  event['Records'].each do |record|
    begin
      puts "Processed Kinesis Event - EventID: #{record['eventID']}"
      record_data = get_record_data_async(record['kinesis'])
      puts "Record Data: #{record_data}"
      # TODO: Do interesting work based on the new data
    rescue => err
      $stderr.puts "An error occurred #{err}"
      raise err
    end
  end
  puts "Successfully processed #{event['Records'].length} records."
end

def get_record_data_async(payload)
  data = Base64.decode64(payload['data']).force_encoding('UTF-8')
```

```
# Placeholder for actual async work
# You can use Ruby's asynchronous programming tools like async/await or fibers
here.
return data
end
```

Melaporkan kegagalan item batch untuk fungsi Lambda dengan pemicu Kinesis

Contoh kode berikut menunjukkan cara mengimplementasikan respons batch sebagian untuk fungsi Lambda yang menerima peristiwa dari aliran Kinesis. Fungsi melaporkan kegagalan item batch dalam respons, memberi sinyal ke Lambda untuk mencoba lagi pesan tersebut nanti.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan [contoh lengkapnya](#) dan pelajari cara mengatur dan menjalankannya di repositori [contoh Nirserver](#).

Melaporkan kegagalan item batch Kinesis dengan Lambda menggunakan Ruby.

```
# Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
# SPDX-License-Identifier: Apache-2.0
require 'aws-sdk'

def lambda_handler(event:, context:)
  batch_item_failures = []

  event['Records'].each do |record|
    begin
      puts "Processed Kinesis Event - EventID: #{record['eventID']}"
      record_data = get_record_data_async(record['kinesis'])
      puts "Record Data: #{record_data}"
      # TODO: Do interesting work based on the new data
    rescue StandardError => err
      puts "An error occurred #{err}"
      # Since we are working with streams, we can return the failed item
      # immediately.
      # Lambda will immediately begin to retry processing from this failed item
      # onwards.
      batch_item_failures << record
    end
  end

  {
    'batchItemFailures': batch_item_failures
  }
end
```

```
        return { batchItemFailures: [{ itemIdentifier: record['kinesis']
['sequenceNumber'] }] }
      end
    end

    puts "Successfully processed #{event['Records'].length} records."
    { batchItemFailures: batch_item_failures }
  end

  def get_record_data_async(payload)
    data = Base64.decode64(payload['data']).force_encoding('utf-8')
    # Placeholder for actual async work
    sleep(1)
    data
  end
end
```

AWS KMS contoh menggunakan SDK untuk Ruby

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan AWS SDK for Ruby with AWS KMS.

Tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

CreateKey

Contoh kode berikut menunjukkan cara menggunakan CreateKey.

SDK untuk Ruby

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
require "aws-sdk-kms" # v2: require 'aws-sdk'

# Create a AWS KMS key.
# As long we are only encrypting small amounts of data (4 KiB or less) directly,
# a KMS key is fine for our purposes.
# For larger amounts of data,
# use the KMS key to encrypt a data encryption key (DEK).

client = Aws::KMS::Client.new

resp = client.create_key({
  tags: [
    {
      tag_key: "CreatedBy",
      tag_value: "ExampleUser"
    }
  ]
})

puts resp.key_metadata.key_id
```

- Untuk API detailnya, lihat [CreateKey](#) di AWS SDK for Ruby API Referensi.

Decrypt

Contoh kode berikut menunjukkan cara menggunakan `Decrypt`.

SDK untuk Ruby

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
require "aws-sdk-kms" # v2: require 'aws-sdk'

# Decrypted blob

blob =
  "01020200785d68faeec386af1057904926253051eb2919d3c16078badf65b808b26dd057c101747cadf3593596"
blob_packed = [blob].pack("H*")

client = Aws::KMS::Client.new(region: "us-west-2")

resp = client.decrypt({
  ciphertext_blob: blob_packed
})


puts "Raw text: "
puts resp.plaintext
```

- Untuk API detailnya, lihat [Dekripsi](#) di AWS SDK for Ruby API Referensi.

Encrypt

Contoh kode berikut menunjukkan cara menggunakan `Encrypt`.

SDK untuk Ruby

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
require "aws-sdk-kms" # v2: require 'aws-sdk'

# ARN of the AWS KMS key.
#
# Replace the fictitious key ARN with a valid key ID

keyId = "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"

text = "1234567890"

client = Aws::KMS::Client.new(region: "us-west-2")

resp = client.encrypt({
  key_id: keyId,
  plaintext: text,
})

# Display a readable version of the resulting encrypted blob.
puts "Blob:"
puts resp.ciphertext_blob.unpack("H*")
```

- Untuk API detailnya, lihat [Enkripsi](#) di AWS SDK for Ruby API Referensi.

ReEncrypt

Contoh kode berikut menunjukkan cara menggunakan ReEncrypt.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
require "aws-sdk-kms" # v2: require 'aws-sdk'

# Human-readable version of the ciphertext of the data to reencrypt.
```



```
blob =
  "01020200785d68faeec386af1057904926253051eb2919d3c16078badf65b808b26dd057c101747cadf3593596"
sourceCiphertextBlob = [blob].pack("H*")

# Replace the fictitious key ARN with a valid key ID

destinationKeyId = "arn:aws:kms:us-west-2:111122223333:key/0987dcba-09fe-87dc-65ba-
ab0987654321"

client = Aws::KMS::Client.new(region: "us-west-2")

resp = client.re_encrypt({
  ciphertext_blob: sourceCiphertextBlob,
  destination_key_id: destinationKeyId
})

# Display a readable version of the resulting re-encrypted blob.
puts "Blob:"
puts resp.ciphertext_blob.unpack("H*")
```

- Untuk API detailnya, lihat [ReEncrypt](#) di AWS SDK for Ruby API Referensi.

Contoh Lambda menggunakan SDK untuk Ruby

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan Lambda AWS SDK for Ruby with.

Tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Skenario adalah contoh kode yang menunjukkan kepada Anda bagaimana menyelesaikan tugas tertentu dengan memanggil beberapa fungsi dalam layanan atau dikombinasikan dengan yang lain AWS layanan.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Memulai

Halo Lambda

Contoh kode berikut menunjukkan cara memulai menggunakan Lambda.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
require 'aws-sdk-lambda'

# Creates an AWS Lambda client using the default credentials and configuration
def lambda_client
  Aws::Lambda::Client.new
end

# Lists the Lambda functions in your AWS account, paginating the results if
# necessary
def list_lambda_functions
  lambda = lambda_client

  # Use a pagination iterator to list all functions
  functions = []
  lambda.list_functions.each_page do |page|
    functions.concat(page.functions)
  end

  # Print the name and ARN of each function
  functions.each do |function|
    puts "Function name: #{function.function_name}"
    puts "Function ARN: #{function.function_arn}"
    puts
  end

  puts "Total functions: #{functions.count}"
end

list_lambda_functions if __FILE__ == $PROGRAM_NAME
```

- Untuk API detailnya, lihat [ListFunctions](#) di AWS SDK for Ruby API Referensi.

Topik

- [Tindakan](#)
- [Skenario](#)
- [Contoh nirserver](#)

Tindakan

CreateFunction

Contoh kode berikut menunjukkan cara menggunakan `CreateFunction`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
class LambdaWrapper
  attr_accessor :lambda_client

  def initialize
    @lambda_client = Aws::Lambda::Client.new
    @logger = Logger.new($stdout)
    @logger.level = Logger::WARN
  end

  # Deploys a Lambda function.
  #
  # @param function_name: The name of the Lambda function.
  # @param handler_name: The fully qualified name of the handler function. This
  #                       must include the file name and the function name.
  # @param role_arn: The IAM role to use for the function.
  # @param deployment_package: The deployment package that contains the function
end
```

```

#           code in .zip format.
# @return: The Amazon Resource Name (ARN) of the newly created function.
def create_function(function_name, handler_name, role_arn, deployment_package)
  response = @lambda_client.create_function({
                                role: role_arn.to_s,
                                function_name: function_name,
                                handler: handler_name,
                                runtime: "ruby2.7",
                                code: {
                                  zip_file: deployment_package
                                },
                                environment: {
                                  variables: {
                                    "LOG_LEVEL" => "info"
                                  }
                                }
                              })

  @lambda_client.wait_until(:function_active_v2, { function_name: function_name})
do |w|
  w.max_attempts = 5
  w.delay = 5
end
  response
rescue Aws::Lambda::Errors::ServiceException => e
  @logger.error("There was an error creating #{function_name}:\n #{e.message}")
rescue Aws::Waiters::Errors::WaiterFailed => e
  @logger.error("Failed waiting for #{function_name} to activate:\n #{e.message}")
end

```

- Untuk API detailnya, lihat [CreateFunction](#) di AWS SDK for Ruby API Referensi.

DeleteFunction

Contoh kode berikut menunjukkan cara menggunakan `DeleteFunction`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
class LambdaWrapper
  attr_accessor :lambda_client

  def initialize
    @lambda_client = Aws::Lambda::Client.new
    @logger = Logger.new($stdout)
    @logger.level = Logger::WARN
  end

  # Deletes a Lambda function.
  # @param function_name: The name of the function to delete.
  def delete_function(function_name)
    print "Deleting function: #{function_name}..."
    @lambda_client.delete_function(
      function_name: function_name
    )
    print "Done!".green
  rescue Aws::Lambda::Errors::ServiceException => e
    @logger.error("There was an error deleting #{function_name}:\n #{e.message}")
  end
end
```

- Untuk API detailnya, lihat [DeleteFunction](#) di AWS SDK for Ruby API Referensi.

GetFunction

Contoh kode berikut menunjukkan cara menggunakan `GetFunction`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
class LambdaWrapper
  attr_accessor :lambda_client

  def initialize
    @lambda_client = Aws::Lambda::Client.new
```

```
@logger = Logger.new($stdout)
@logger.level = Logger::WARN
end

# Gets data about a Lambda function.
#
# @param function_name: The name of the function.
# @return response: The function data, or nil if no such function exists.
def get_function(function_name)
  @lambda_client.get_function(
    {
      function_name: function_name
    }
  )
rescue Aws::Lambda::Errors::ResourceNotFoundException => e
  @logger.debug("Could not find function: #{function_name}:\n #{e.message}")
  nil
end
```

- Untuk API detailnya, lihat [GetFunction](#) di AWS SDK for Ruby API Referensi.

Invoke

Contoh kode berikut menunjukkan cara menggunakan `Invoke`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
class LambdaWrapper
  attr_accessor :lambda_client

  def initialize
    @lambda_client = Aws::Lambda::Client.new
    @logger = Logger.new($stdout)
    @logger.level = Logger::WARN
  end
end
```

```
end

# Invokes a Lambda function.
# @param function_name [String] The name of the function to invoke.
# @param payload [nil] Payload containing runtime parameters.
# @return [Object] The response from the function invocation.
def invoke_function(function_name, payload = nil)
  params = { function_name: function_name }
  params[:payload] = payload unless payload.nil?
  @lambda_client.invoke(params)
rescue Aws::Lambda::Errors::ServiceException => e
  @logger.error("There was an error executing #{function_name}:\n #{e.message}")
end
```

- Untuk API detailnya, lihat [Memanggil di AWS SDK for Ruby API Referensi](#).

ListFunctions

Contoh kode berikut menunjukkan cara menggunakan `ListFunctions`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
class LambdaWrapper
  attr_accessor :lambda_client

  def initialize
    @lambda_client = Aws::Lambda::Client.new
    @logger = Logger.new($stdout)
    @logger.level = Logger::WARN
  end

  # Lists the Lambda functions for the current account.
  def list_functions
    functions = []
```

```

@lambda_client.list_functions.each do |response|
  response["functions"].each do |function|
    functions.append(function["function_name"])
  end
end
functions
rescue Aws::Lambda::Errors::ServiceException => e
  @logger.error("There was an error executing #{function_name}:\n #{e.message}")
end

```

- Untuk API detailnya, lihat [ListFunctions](#) di AWS SDK for Ruby API Referensi.

UpdateFunctionCode

Contoh kode berikut menunjukkan cara menggunakan `UpdateFunctionCode`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di [Repositori Contoh Kode AWS](#).

```

class LambdaWrapper
  attr_accessor :lambda_client

  def initialize
    @lambda_client = Aws::Lambda::Client.new
    @logger = Logger.new($stdout)
    @logger.level = Logger::WARN
  end

  # Updates the code for a Lambda function by submitting a .zip archive that
  # contains
  # the code for the function.

  # @param function_name: The name of the function to update.
  # @param deployment_package: The function code to update, packaged as bytes in
  #                               .zip format.
  # @return: Data about the update, including the status.

```



```
def update_function_code(function_name, deployment_package)
  @lambda_client.update_function_code(
    function_name: function_name,
    zip_file: deployment_package
  )
  @lambda_client.wait_until(:function_updated_v2, { function_name: function_name})
do |w|
  w.max_attempts = 5
  w.delay = 5
end
rescue Aws::Lambda::Errors::ServiceException => e
  @logger.error("There was an error updating function code for: #{function_name}:
\n #{e.message}")
  nil
rescue Aws::Waiters::Errors::WaiterFailed => e
  @logger.error("Failed waiting for #{function_name} to update:\n #{e.message}")
end
```

- Untuk API detailnya, lihat [UpdateFunctionCode](#) di AWS SDK for Ruby API Referensi.

UpdateFunctionConfiguration

Contoh kode berikut menunjukkan cara menggunakan `UpdateFunctionConfiguration`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
class LambdaWrapper
  attr_accessor :lambda_client

  def initialize
    @lambda_client = Aws::Lambda::Client.new
    @logger = Logger.new($stdout)
    @logger.level = Logger::WARN
  end
end
```

```
# Updates the environment variables for a Lambda function.
# @param function_name: The name of the function to update.
# @param log_level: The log level of the function.
# @return: Data about the update, including the status.
def update_function_configuration(function_name, log_level)
  @lambda_client.update_function_configuration({
    function_name: function_name,
    environment: {
      variables: {
        "LOG_LEVEL" => log_level
      }
    }
  })

  @lambda_client.wait_until(:function_updated_v2, { function_name: function_name})
do |w|
  w.max_attempts = 5
  w.delay = 5
end
rescue Aws::Lambda::Errors::ServiceException => e
  @logger.error("There was an error updating configurations for #{function_name}:
\n #{e.message}")
rescue Aws::Waiters::Errors::WaiterFailed => e
  @logger.error("Failed waiting for #{function_name} to activate:\n #{e.message}")
end
```

- Untuk API detailnya, lihat [UpdateFunctionConfiguration](#) di AWS SDK for Ruby API Referensi.

Skenario

Buat aplikasi untuk menganalisis umpan balik pelanggan

Contoh kode berikut menunjukkan cara membuat aplikasi yang menganalisis kartu komentar pelanggan, menerjemahkannya dari bahasa aslinya, menentukan sentimen mereka, dan menghasilkan file audio dari teks yang diterjemahkan.

SDK untuk Ruby

Aplikasi contoh ini menganalisis dan menyimpan kartu umpan balik pelanggan. Secara khusus, ini memenuhi kebutuhan hotel fiktif di New York City. Hotel menerima umpan balik dari para tamu dalam berbagai bahasa dalam bentuk kartu komentar fisik. Umpan balik itu diunggah ke aplikasi melalui klien web. Setelah gambar kartu komentar diunggah, langkah-langkah berikut terjadi:

- Teks diekstraksi dari gambar menggunakan Amazon Textract.
- Amazon Comprehend menentukan sentimen teks yang diekstraksi dan bahasanya.
- Teks yang diekstraksi diterjemahkan ke bahasa Inggris menggunakan Amazon Translate.
- Amazon Polly mensintesis file audio dari teks yang diekstraksi.

Aplikasi lengkap dapat digunakan dengan AWS CDK Untuk kode sumber dan petunjuk penerapan, lihat proyek di [GitHub](#).

Layanan yang digunakan dalam contoh ini

- Amazon Comprehend
- Lambda
- Amazon Polly
- Amazon Textract
- Amazon Translate


Memulai dengan fungsi

Contoh kode berikut ini menunjukkan cara:

- Buat IAM peran dan fungsi Lambda, lalu unggah kode handler.
- Panggil fungsi dengan satu parameter dan dapatkan hasil.
- Perbarui kode fungsi dan konfigurasi dengan variabel lingkungan.
- Panggil fungsi dengan parameter baru dan dapatkan hasil. Tampilkan log eksekusi yang dikembalikan.
- Buat daftar fungsi untuk akun Anda, lalu bersihkan sumber daya.

Untuk informasi selengkapnya, lihat [Membuat fungsi Lambda dengan konsol](#).

SDK untuk Ruby

 Note

Ada lebih banyak tentang GitHub. Temukan contoh selengkapnya dan pelajari cara pengaturan dan menjalankannya di [Repositori Contoh Kode AWS](#).

Siapkan IAM izin prasyarat untuk fungsi Lambda yang mampu menulis log.

```
# Get an AWS Identity and Access Management (IAM) role.
#
# @param iam_role_name: The name of the role to retrieve.
# @param action: Whether to create or destroy the IAM apparatus.
# @return: The IAM role.
def manage_iam(iam_role_name, action)
  role_policy = {
    'Version': "2012-10-17",
    'Statement': [
      {
        'Effect': "Allow",
        'Principal': {
          'Service': "lambda.amazonaws.com"
        },
        'Action': "sts:AssumeRole"
      }
    ]
  }
  case action
  when "create"
    role = $iam_client.create_role(
      role_name: iam_role_name,
      assume_role_policy_document: role_policy.to_json
    )
    $iam_client.attach_role_policy(
      {
        policy_arn: "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole",
        role_name: iam_role_name
      }
    )
    $iam_client.wait_until(:role_exists, { role_name: iam_role_name }) do |w|
      w.max_attempts = 5
      w.delay = 5
    end
    @logger.debug("Successfully created IAM role: #{role['role']['arn']}")
    @logger.debug("Enforcing a 10-second sleep to allow IAM role to activate
fully.")
    sleep(10)
    return role, role_policy.to_json
  when "destroy"
    $iam_client.detach_role_policy(
```

```
    {
      policy_arn: "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole",
      role_name: iam_role_name
    }
  )
  $iam_client.delete_role(
    role_name: iam_role_name
  )
  @logger.debug("Detached policy & deleted IAM role: #{iam_role_name}")
else
  raise "Incorrect action provided. Must provide 'create' or 'destroy'"
end
rescue Aws::Lambda::Errors::ServiceException => e
  @logger.error("There was an error creating role or attaching policy:\n
#{e.message}")
end
```

Tentukan handler Lambda yang menambah angka yang disediakan sebagai parameter pemanggilan.

```
require "logger"

# A function that increments a whole number by one (1) and logs the result.
# Requires a manually-provided runtime parameter, 'number', which must be Int
#
# @param event [Hash] Parameters sent when the function is invoked
# @param context [Hash] Methods and properties that provide information
# about the invocation, function, and execution environment.
# @return incremented_number [String] The incremented number.
def lambda_handler(event:, context:)
  logger = Logger.new($stdout)
  log_level = ENV["LOG_LEVEL"]
  logger.level = case log_level
                 when "debug"
                   Logger::DEBUG
                 when "info"
                   Logger::INFO
                 else
                   Logger::ERROR
                 end

  logger.debug("This is a debug log message.")
```

```

logger.info("This is an info log message. Code executed successfully!")
number = event["number"].to_i
incremented_number = number + 1
logger.info("You provided #{number.round} and it was incremented to
#{incremented_number.round}")
incremented_number.round.to_s
end

```

Zip fungsi Lambda Anda ke dalam paket penerapan.

```

# Creates a Lambda deployment package in .zip format.
# This zip can be passed directly as a string to Lambda when creating the
function.
#
# @param source_file: The name of the object, without suffix, for the Lambda file
and zip.
# @return: The deployment package.
def create_deployment_package(source_file)
  Dir.chdir(File.dirname(__FILE__))
  if File.exist?("lambda_function.zip")
    File.delete("lambda_function.zip")
    @logger.debug("Deleting old zip: lambda_function.zip")
  end
  Zip::File.open("lambda_function.zip", create: true) {
    |zipfile|
    zipfile.add("lambda_function.rb", "#{source_file}.rb")
  }
  @logger.debug("Zipping #{source_file}.rb into: lambda_function.zip.")
  File.read("lambda_function.zip").to_s
rescue StandardError => e
  @logger.error("There was an error creating deployment package:\n #{e.message}")
end

```

Buat fungsi Lambda baru.

```

# Deploys a Lambda function.
#
# @param function_name: The name of the Lambda function.
# @param handler_name: The fully qualified name of the handler function. This
#                       must include the file name and the function name.
# @param role_arn: The IAM role to use for the function.

```

```

# @param deployment_package: The deployment package that contains the function
#                               code in .zip format.
# @return: The Amazon Resource Name (ARN) of the newly created function.
def create_function(function_name, handler_name, role_arn, deployment_package)
  response = @lambda_client.create_function({
    role: role_arn.to_s,
    function_name: function_name,
    handler: handler_name,
    runtime: "ruby2.7",
    code: {
      zip_file: deployment_package
    },
    environment: {
      variables: {
        "LOG_LEVEL" => "info"
      }
    }
  })

  @lambda_client.wait_until(:function_active_v2, { function_name: function_name})
do |w|
  w.max_attempts = 5
  w.delay = 5
end
  response
rescue Aws::Lambda::Errors::ServiceException => e
  @logger.error("There was an error creating #{function_name}:\n #{e.message}")
rescue Aws::Waiters::Errors::WaiterFailed => e
  @logger.error("Failed waiting for #{function_name} to activate:\n #{e.message}")
end

```

Panggil fungsi Lambda Anda dengan parameter runtime opsional.

```

# Invokes a Lambda function.
# @param function_name [String] The name of the function to invoke.
# @param payload [nil] Payload containing runtime parameters.
# @return [Object] The response from the function invocation.
def invoke_function(function_name, payload = nil)
  params = { function_name: function_name }
  params[:payload] = payload unless payload.nil?
  @lambda_client.invoke(params)
rescue Aws::Lambda::Errors::ServiceException => e
  @logger.error("There was an error executing #{function_name}:\n #{e.message}")
end

```

```
end
```

Perbarui konfigurasi fungsi Lambda Anda untuk menyuntikkan variabel lingkungan baru.

```
# Updates the environment variables for a Lambda function.
# @param function_name: The name of the function to update.
# @param log_level: The log level of the function.
# @return: Data about the update, including the status.
def update_function_configuration(function_name, log_level)
  @lambda_client.update_function_configuration({
    function_name: function_name,
    environment: {
      variables: {
        "LOG_LEVEL" => log_level
      }
    }
  })
  @lambda_client.wait_until(:function_updated_v2, { function_name: function_name})
do |w|
  w.max_attempts = 5
  w.delay = 5
end
rescue Aws::Lambda::Errors::ServiceException => e
  @logger.error("There was an error updating configurations for #{function_name}:
\n #{e.message}")
rescue Aws::Waiters::Errors::WaiterFailed => e
  @logger.error("Failed waiting for #{function_name} to activate:\n #{e.message}")
end
```

Perbarui kode fungsi Lambda Anda dengan paket penerapan berbeda yang berisi kode berbeda.

```
# Updates the code for a Lambda function by submitting a .zip archive that
contains
# the code for the function.

# @param function_name: The name of the function to update.
# @param deployment_package: The function code to update, packaged as bytes in
#                               .zip format.
# @return: Data about the update, including the status.
def update_function_code(function_name, deployment_package)
  @lambda_client.update_function_code(
```



```

    function_name: function_name,
    zip_file: deployment_package
  )
  @lambda_client.wait_until(:function_updated_v2, { function_name: function_name})
do |w|
  w.max_attempts = 5
  w.delay = 5
end
rescue Aws::Lambda::Errors::ServiceException => e
  @logger.error("There was an error updating function code for: #{function_name}:
\n #{e.message}")
  nil
rescue Aws::Waiters::Errors::WaiterFailed => e
  @logger.error("Failed waiting for #{function_name} to update:\n #{e.message}")
end

```

Daftar semua fungsi Lambda yang ada menggunakan paginator bawaan.

```

# Lists the Lambda functions for the current account.
def list_functions
  functions = []
  @lambda_client.list_functions.each do |response|
    response["functions"].each do |function|
      functions.append(function["function_name"])
    end
  end
  functions
rescue Aws::Lambda::Errors::ServiceException => e
  @logger.error("There was an error executing #{function_name}:\n #{e.message}")
end

```

Hapus fungsi Lambda tertentu.

```

# Deletes a Lambda function.
# @param function_name: The name of the function to delete.
def delete_function(function_name)
  print "Deleting function: #{function_name}..."
  @lambda_client.delete_function(
    function_name: function_name
  )
  print "Done!".green

```

```
rescue Aws::Lambda::Errors::ServiceException => e
  @logger.error("There was an error deleting #{function_name}:\n #{e.message}")
end
```

- Untuk API detailnya, lihat topik berikut di AWS SDK for Ruby API Referensi.
 - [CreateFunction](#)
 - [DeleteFunction](#)
 - [GetFunction](#)
 - [Memohon](#)
 - [ListFunctions](#)
 - [UpdateFunctionCode](#)
 - [UpdateFunctionConfiguration](#)

Contoh nirserver

Menghubungkan ke RDS database Amazon dalam fungsi Lambda

Contoh kode berikut menunjukkan bagaimana menerapkan fungsi Lambda yang menghubungkan ke database. RDS Fungsi membuat permintaan database sederhana dan mengembalikan hasilnya.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di repositori [contoh Nirserver](#).

Menghubungkan ke RDS database Amazon dalam fungsi Lambda menggunakan Ruby.

```
# Ruby code here.

require 'aws-sdk-rds'
require 'json'
require 'mysql2'

def lambda_handler(event:, context:)
```

```
endpoint = ENV['DBEndpoint'] # Add the endpoint without https"
port = ENV['Port']           # 3306
user = ENV['DBUser']
region = ENV['DBRegion']     # 'us-east-1'
db_name = ENV['DBName']

credentials = Aws::Credentials.new(
  ENV['AWS_ACCESS_KEY_ID'],
  ENV['AWS_SECRET_ACCESS_KEY'],
  ENV['AWS_SESSION_TOKEN']
)
rds_client = Aws::RDS::AuthTokenGenerator.new(
  region: region,
  credentials: credentials
)

token = rds_client.auth_token(
  endpoint: endpoint+ ':' + port,
  user_name: user,
  region: region
)

begin
  conn = Mysql2::Client.new(
    host: endpoint,
    username: user,
    password: token,
    port: port,
    database: db_name,
    sslca: '/var/task/global-bundle.pem',
    sslverify: true,
    enable_clear_text_plugin: true
  )
  a = 3
  b = 2
  result = conn.query("SELECT #{a} + #{b} AS sum").first['sum']
  puts result
  conn.close
  {
    statusCode: 200,
    body: result.to_json
  }
rescue => e
  puts "Database connection failed due to #{e}"
```

```
end
end
```

Memanggil fungsi Lambda dari pemicu Kinesis

Contoh kode berikut menunjukkan bagaimana menerapkan fungsi Lambda yang menerima peristiwa yang dipicu oleh menerima catatan dari aliran Kinesis. Fungsi mengambil payload Kinesis, mendekode dari Base64, dan mencatat konten rekaman.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di repositori [contoh Nirserver](#).

Mengonsumsi acara Kinesis dengan Lambda menggunakan Ruby.

```
# Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
# SPDX-License-Identifier: Apache-2.0
require 'aws-sdk'

def lambda_handler(event:, context:)
  event['Records'].each do |record|
    begin
      puts "Processed Kinesis Event - EventID: #{record['eventID']}"
      record_data = get_record_data_async(record['kinesis'])
      puts "Record Data: #{record_data}"
      # TODO: Do interesting work based on the new data
    rescue => err
      $stderr.puts "An error occurred #{err}"
      raise err
    end
  end
  puts "Successfully processed #{event['Records'].length} records."
end

def get_record_data_async(payload)
  data = Base64.decode64(payload['data']).force_encoding('UTF-8')
  # Placeholder for actual async work
```

```
# You can use Ruby's asynchronous programming tools like async/await or fibers
here.
  return data
end
```

Memanggil fungsi Lambda dari pemicu DynamoDB

Contoh kode berikut menunjukkan bagaimana menerapkan fungsi Lambda yang menerima peristiwa yang dipicu oleh menerima catatan dari aliran DynamoDB. Fungsi mengambil payload DynamoDB dan mencatat isi catatan.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di repositori [contoh Nirserver](#).

Mengonsumsi acara DynamoDB dengan Lambda menggunakan Ruby.

```
def lambda_handler(event:, context:)
  return 'received empty event' if event['Records'].empty?

  event['Records'].each do |record|
    log_dynamodb_record(record)
  end

  "Records processed: #{event['Records'].length}"
end

def log_dynamodb_record(record)
  puts record['eventID']
  puts record['eventName']
  puts "DynamoDB Record: #{JSON.generate(record['dynamodb'])}"
end
```

Memanggil fungsi Lambda dari pemicu Amazon DocumentDB

Contoh kode berikut menunjukkan cara menerapkan fungsi Lambda yang menerima peristiwa yang dipicu dengan menerima catatan dari aliran perubahan DocumentDB. Fungsi mengambil payload DocumentDB dan mencatat isi catatan.

SDKuntuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di repositori [contoh Nirserver](#).

Mengonsumsi acara Amazon DocumentDB dengan Lambda menggunakan Ruby.

```
require 'json'

def lambda_handler(event:, context:)
  event['events'].each do |record|
    log_document_db_event(record)
  end
  'OK'
end

def log_document_db_event(record)
  event_data = record['event'] || {}
  operation_type = event_data['operationType'] || 'Unknown'
  db = event_data.dig('ns', 'db') || 'Unknown'
  collection = event_data.dig('ns', 'coll') || 'Unknown'
  full_document = event_data['fullDocument'] || {}

  puts "Operation type: #{operation_type}"
  puts "db: #{db}"
  puts "collection: #{collection}"
  puts "Full document: #{JSON.pretty_generate(full_document)}"
end
```

Memanggil fungsi Lambda dari pemicu Amazon MSK

Contoh kode berikut menunjukkan cara menerapkan fungsi Lambda yang menerima peristiwa yang dipicu dengan menerima catatan dari kluster Amazon MSK. Fungsi mengambil MSK payload dan mencatat isi catatan.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di repositori [contoh Nirserver](#).

Mengonsumsi MSK acara Amazon dengan Lambda menggunakan Ruby.

```
require 'base64'

def lambda_handler(event:, context:)
  # Iterate through keys
  event['records'].each do |key, records|
    puts "Key: #{key}"

    # Iterate through records
    records.each do |record|
      puts "Record: #{record}"

      # Decode base64
      msg = Base64.decode64(record['value'])
      puts "Message: #{msg}"
    end
  end
end
```

Menginvokasi fungsi Lambda dari pemicu Amazon S3

Contoh kode berikut menunjukkan cara mengimplementasikan fungsi Lambda yang menerima peristiwa yang dipicu dengan mengunggah objek ke bucket S3. Fungsi ini mengambil nama bucket S3 dan kunci objek dari parameter event dan memanggil Amazon API S3 untuk mengambil dan mencatat jenis konten objek.

SDKuntuk Ruby

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di repositori [contoh Nirserver](#).

Mengonsumsi acara S3 dengan Lambda menggunakan Ruby.

```
require 'json'
require 'uri'
require 'aws-sdk'

puts 'Loading function'

def lambda_handler(event:, context:)
  s3 = Aws::S3::Client.new(region: 'region') # Your AWS region
  # puts "Received event: #{JSON.dump(event)}"

  # Get the object from the event and show its content type
  bucket = event['Records'][0]['s3']['bucket']['name']
  key = URI.decode_www_form_component(event['Records'][0]['s3']['object']['key'],
  Encoding::UTF_8)
  begin
    response = s3.get_object(bucket: bucket, key: key)
    puts "CONTENT TYPE: #{response.content_type}"
    return response.content_type
  rescue StandardError => e
    puts e.message
    puts "Error getting object #{key} from bucket #{bucket}. Make sure they exist
    and your bucket is in the same region as this function."
    raise e
  end
end
```


Memanggil fungsi Lambda dari pemicu Amazon SNS

Contoh kode berikut menunjukkan cara menerapkan fungsi Lambda yang menerima peristiwa yang dipicu dengan menerima pesan dari suatu SNS topik. Fungsi mengambil pesan dari parameter peristiwa dan mencatat konten setiap pesan.

SDKuntuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di repositori [contoh Nirserver](#).

Mengonsumsi SNS acara dengan Lambda menggunakan Ruby.

```
# Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
# SPDX-License-Identifier: Apache-2.0
def lambda_handler(event:, context:)
  event['Records'].map { |record| process_message(record) }
end

def process_message(record)
  message = record['Sns']['Message']
  puts("Processing message: #{message}")
rescue StandardError => e
  puts("Error processing message: #{e}")
  raise
end
```

Memanggil fungsi Lambda dari pemicu Amazon SQS

Contoh kode berikut menunjukkan cara menerapkan fungsi Lambda yang menerima peristiwa yang dipicu oleh menerima pesan dari antrianSQS. Fungsi mengambil pesan dari parameter peristiwa dan mencatat konten setiap pesan.

SDK untuk Ruby

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di repositori [contoh Nirserver](#).

Mengonsumsi SQS secara acak dengan Lambda menggunakan Ruby.

```
# Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
# SPDX-License-Identifier: Apache-2.0
def lambda_handler(event:, context:)
  event['Records'].each do |message|
    process_message(message)
  end
  puts "done"
end

def process_message(message)
  begin
    puts "Processed message #{message['body']}"
    # TODO: Do interesting work based on the new message
  rescue StandardError => err
    puts "An error occurred"
    raise err
  end
end
```

Melaporkan kegagalan item batch untuk fungsi Lambda dengan pemicu Kinesis

Contoh kode berikut menunjukkan cara mengimplementasikan respons batch sebagian untuk fungsi Lambda yang menerima peristiwa dari aliran Kinesis. Fungsi melaporkan kegagalan item batch dalam respons, memberi sinyal ke Lambda untuk mencoba lagi pesan tersebut nanti.

SDK untuk Ruby

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di repositori [contoh Nirserver](#).

Melaporkan kegagalan item batch Kinesis dengan Lambda menggunakan Ruby.

```
# Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
# SPDX-License-Identifier: Apache-2.0
require 'aws-sdk'

def lambda_handler(event:, context:)
  batch_item_failures = []

  event['Records'].each do |record|
    begin
      puts "Processed Kinesis Event - EventID: #{record['eventID']}"
      record_data = get_record_data_async(record['kinesis'])
      puts "Record Data: #{record_data}"
      # TODO: Do interesting work based on the new data
    rescue StandardError => err
      puts "An error occurred #{err}"
      # Since we are working with streams, we can return the failed item
      # immediately.
      # Lambda will immediately begin to retry processing from this failed item
      # onwards.
      return { batchItemFailures: [{ itemIdentifier: record['kinesis']
['sequenceNumber'] }] }
    end
  end

  puts "Successfully processed #{event['Records'].length} records."
  { batchItemFailures: batch_item_failures }
end

def get_record_data_async(payload)
  data = Base64.decode64(payload['data']).force_encoding('utf-8')
  # Placeholder for actual async work
  sleep(1)
  data
end
```

```
end
```

Melaporkan kegagalan item batch untuk fungsi Lambda dengan pemicu DynamoDB

Contoh kode berikut menunjukkan cara mengimplementasikan respons batch sebagian untuk fungsi Lambda yang menerima peristiwa dari aliran DynamoDB. Fungsi melaporkan kegagalan item batch dalam respons, memberi sinyal ke Lambda untuk mencoba lagi pesan tersebut nanti.

SDKuntuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di repositori [contoh Nirserver](#).

Melaporkan kegagalan item batch DynamoDB dengan Lambda menggunakan Ruby.

```
def lambda_handler(event:, context:)
  records = event["Records"]
  cur_record_sequence_number = ""

  records.each do |record|
    begin
      # Process your record
      cur_record_sequence_number = record["dynamodb"]["SequenceNumber"]
      rescue StandardError => e
        # Return failed record's sequence number
        return {"batchItemFailures" => [{"itemIdentifier" =>
cur_record_sequence_number}]}
    end
  end

  {"batchItemFailures" => []}
end
```

Melaporkan kegagalan item batch untuk fungsi Lambda dengan pemicu Amazon SQS

Contoh kode berikut menunjukkan cara mengimplementasikan respons batch sebagian untuk fungsi Lambda yang menerima peristiwa dari antrian SQS. Fungsi melaporkan kegagalan item batch dalam respons, memberi sinyal ke Lambda untuk mencoba lagi pesan tersebut nanti.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di repositori [contoh Nirserver](#).

Melaporkan kegagalan item SQS batch dengan Lambda menggunakan Ruby.

```
# Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
# SPDX-License-Identifier: Apache-2.0
require 'json'

def lambda_handler(event:, context:)
  if event
    batch_item_failures = []
    sqs_batch_response = {}

    event["Records"].each do |record|
      begin
        # process message
        rescue StandardError => e
          batch_item_failures << {"itemIdentifier" => record['messageId']}
        end
      end

      sqs_batch_response["batchItemFailures"] = batch_item_failures
      return sqs_batch_response
    end
  end
end
```

MSK Contoh Amazon menggunakan SDK untuk Ruby

Contoh kode berikut menunjukkan kepada Anda cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan AWS SDK for Ruby With AmazonMSK.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Contoh nirserver](#)

Contoh nirserver

Memanggil fungsi Lambda dari pemicu Amazon MSK

Contoh kode berikut menunjukkan cara menerapkan fungsi Lambda yang menerima peristiwa yang dipicu dengan menerima catatan dari kluster AmazonMSK. Fungsi mengambil MSK payload dan mencatat isi catatan.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di repositori [contoh Nirserver](#).

Mengonsumsi MSK acara Amazon dengan Lambda menggunakan Ruby.

```
require 'base64'

def lambda_handler(event:, context:)
  # Iterate through keys
  event['records'].each do |key, records|
    puts "Key: #{key}"

    # Iterate through records
    records.each do |record|
```

```
puts "Record: #{record}"

# Decode base64
msg = Base64.decode64(record['value'])
puts "Message: #{msg}"
end
end
end
```

Contoh Amazon Polly menggunakan Ruby SDK

Contoh kode berikut menunjukkan cara melakukan tindakan dan menerapkan skenario umum dengan menggunakan Amazon Polly. AWS SDK for Ruby

Tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Skenario adalah contoh kode yang menunjukkan kepada Anda bagaimana menyelesaikan tugas tertentu dengan memanggil beberapa fungsi dalam layanan atau dikombinasikan dengan yang lain AWS layanan.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)
- [Skenario](#)

Tindakan

DescribeVoices

Contoh kode berikut menunjukkan cara menggunakan `DescribeVoices`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
require "aws-sdk-polly" # In v2: require 'aws-sdk'

begin
  # Create an Amazon Polly client using
  # credentials from the shared credentials file ~/.aws/credentials
  # and the configuration (region) from the shared configuration file ~/.aws/config
  polly = Aws::Polly::Client.new

  # Get US English voices
  resp = polly.describe_voices(language_code: "en-US")

  resp.voices.each do |v|
    puts v.name
    puts "  " + v.gender
    puts
  end
rescue StandardError => ex
  puts "Could not get voices"
  puts "Error message:"
  puts ex.message
end
```

- Untuk API detailnya, lihat [DescribeVoices](#) di AWS SDK for Ruby API Referensi.

ListLexicons

Contoh kode berikut menunjukkan cara menggunakan `ListLexicons`.

SDK untuk Ruby

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
require "aws-sdk-polly" # In v2: require 'aws-sdk'

begin
  # Create an Amazon Polly client using
  # credentials from the shared credentials file ~/.aws/credentials
  # and the configuration (region) from the shared configuration file ~/.aws/config
  polly = Aws::Polly::Client.new

  resp = polly.list_lexicons

  resp.lexicons.each do |l|
    puts l.name
    puts "  Alphabet:" + l.attributes.alphabet
    puts "  Language:" + l.attributes.language
    puts
  end
rescue StandardError => ex
  puts "Could not get lexicons"
  puts "Error message:"
  puts ex.message
end
```

- Untuk API detailnya, lihat [ListLexicons](#) di AWS SDK for Ruby API Referensi.

SynthesizeSpeech

Contoh kode berikut menunjukkan cara menggunakan `SynthesizeSpeech`.

SDK untuk Ruby

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
require "aws-sdk-polly" # In v2: require 'aws-sdk'

begin
  # Get the filename from the command line
  if ARGV.empty?
    puts "You must supply a filename"
    exit 1
  end

  filename = ARGV[0]

  # Open file and get the contents as a string
  if File.exist?(filename)
    contents = IO.read(filename)
  else
    puts "No such file: " + filename
    exit 1
  end

  # Create an Amazon Polly client using
  # credentials from the shared credentials file ~/.aws/credentials
  # and the configuration (region) from the shared configuration file ~/.aws/config
  polly = Aws::Polly::Client.new

  resp = polly.synthesize_speech({
    output_format: "mp3",
    text: contents,
    voice_id: "Joanna",
  })

  # Save output
  # Get just the file name
  # abc/xyz.txt -> xyx.txt
```

```
name = File.basename(filename)

# Split up name so we get just the xyz part
parts = name.split(".")
first_part = parts[0]
mp3_file = first_part + ".mp3"

IO.copy_stream(resp.audio_stream, mp3_file)

puts "Wrote MP3 content to: " + mp3_file
rescue StandardError => ex
  puts "Got error:"
  puts "Error message:"
  puts ex.message
end
```

- Untuk API detailnya, lihat [SynthesizeSpeech](#) di AWS SDK for Ruby API Referensi.

Skenario

Buat aplikasi untuk menganalisis umpan balik pelanggan

Contoh kode berikut menunjukkan cara membuat aplikasi yang menganalisis kartu komentar pelanggan, menerjemahkannya dari bahasa aslinya, menentukan sentimen mereka, dan menghasilkan file audio dari teks yang diterjemahkan.

SDK untuk Ruby

Aplikasi contoh ini menganalisis dan menyimpan kartu umpan balik pelanggan. Secara khusus, ini memenuhi kebutuhan hotel fiktif di New York City. Hotel menerima umpan balik dari para tamu dalam berbagai bahasa dalam bentuk kartu komentar fisik. Umpan balik itu diunggah ke aplikasi melalui klien web. Setelah gambar kartu komentar diunggah, langkah-langkah berikut terjadi:

- Teks diekstraksi dari gambar menggunakan Amazon Textract.
- Amazon Comprehend menentukan sentimen teks yang diekstraksi dan bahasanya.
- Teks yang diekstraksi diterjemahkan ke bahasa Inggris menggunakan Amazon Translate.
- Amazon Polly mensintesis file audio dari teks yang diekstraksi.

Aplikasi lengkap dapat digunakan dengan AWS CDK Untuk kode sumber dan petunjuk penerapan, lihat proyek di [GitHub](#).

Layanan yang digunakan dalam contoh ini

- Amazon Comprehend
- Lambda
- Amazon Polly
- Amazon Textract
- Amazon Translate

RDS Contoh Amazon menggunakan SDK untuk Ruby

Contoh kode berikut menunjukkan kepada Anda cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan AWS SDK for Ruby With AmazonRDS.

Tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Memulai

Halo Amazon RDS

Contoh kode berikut menunjukkan cara memulai menggunakan AmazonRDS.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
require 'aws-sdk-rds'
require 'logger'

# RDSManager is a class responsible for managing RDS operations
# such as listing all RDS DB instances in the current AWS account.
```

```
class RDSManager
  def initialize(client)
    @client = client
    @logger = Logger.new($stdout)
  end

  # Lists and prints all RDS DB instances in the current AWS account.
  def list_db_instances
    @logger.info('Listing RDS DB instances')

    paginator = @client.describe_db_instances
    instances = []

    paginator.each_page do |page|
      instances.concat(page.db_instances)
    end

    if instances.empty?
      @logger.info('No instances found.')
    else
      @logger.info("Found #{instances.count} instance(s):")
      instances.each do |instance|
        @logger.info(" * #{instance.db_instance_identifier}
          (#{instance.db_instance_status})")
      end
    end
  end

  end

  end

  if $PROGRAM_NAME == __FILE__
    rds_client = Aws::RDS::Client.new(region: 'us-west-2')
    manager = RDSManager.new(rds_client)
    manager.list_db_instances
  end
end
```

- Untuk API detailnya, lihat [DescribeDBInstances](#) di AWS SDK for Ruby API Referensi.

Topik

- [Tindakan](#)
- [Contoh nirservers](#)

Tindakan

CreateDBSnapshot

Contoh kode berikut menunjukkan cara menggunakan `CreateDBSnapshot`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
require "aws-sdk-rds" # v2: require 'aws-sdk'


# Create a snapshot for an Amazon Relational Database Service (Amazon RDS)
# DB instance.
#
# @param rds_resource [Aws::RDS::Resource] The resource containing SDK logic.
# @param db_instance_name [String] The name of the Amazon RDS DB instance.
# @return [Aws::RDS::DBSnapshot, nil] The snapshot created, or nil if error.
def create_snapshot(rds_resource, db_instance_name)
  id = "snapshot-#{rand(10**6)}"
  db_instance = rds_resource.db_instance(db_instance_name)
  db_instance.create_snapshot({
    db_snapshot_identifier: id
  })
rescue Aws::Errors::ServiceError => e
  puts "Couldn't create DB instance snapshot #{id}: \n #{e.message}"
end
```

- Untuk API detailnya, lihat [CreateDBSnapshot](#) di AWS SDK for Ruby API Referensi.

DescribeDBInstances

Contoh kode berikut menunjukkan cara menggunakan `DescribeDBInstances`.

SDK untuk Ruby

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
require "aws-sdk-rds" # v2: require 'aws-sdk'

# List all Amazon Relational Database Service (Amazon RDS) DB instances.
#
# @param rds_resource [Aws::RDS::Resource] An SDK for Ruby Amazon RDS resource.
# @return [Array, nil] List of all DB instances, or nil if error.
def list_instances(rds_resource)
  db_instances = []
  rds_resource.db_instances.each do |i|
    db_instances.append({
      "name": i.id,
      "status": i.db_instance_status
    })
  end
  db_instances
rescue Aws::Errors::ServiceError => e
  puts "Couldn't list instances:\n#{e.message}"
end
```

- Untuk API detailnya, lihat [DescribeDBInstances](#) di AWS SDK for Ruby API Referensi.

DescribeDBParameterGroups

Contoh kode berikut menunjukkan cara menggunakan `DescribeDBParameterGroups`.

SDK untuk Ruby

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
require "aws-sdk-rds" # v2: require 'aws-sdk'

# List all Amazon Relational Database Service (Amazon RDS) parameter groups.
#
# @param rds_resource [Aws::RDS::Resource] An SDK for Ruby Amazon RDS resource.
# @return [Array, nil] List of all parameter groups, or nil if error.
def list_parameter_groups(rds_resource)
  parameter_groups = []
  rds_resource.db_parameter_groups.each do |p|
    parameter_groups.append({
      "name": p.db_parameter_group_name,
      "description": p.description
    })
  end
  parameter_groups
rescue Aws::Errors::ServiceError => e
  puts "Couldn't list parameter groups:\n #{e.message}"
end
```

- Untuk API detailnya, lihat [describeDBParameterGroup](#) dalam AWS SDK for Ruby API Referensi.

DescribeDBParameters

Contoh kode berikut menunjukkan cara menggunakan `DescribeDBParameters`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
require "aws-sdk-rds" # v2: require 'aws-sdk'

# List all Amazon Relational Database Service (Amazon RDS) parameter groups.
#
# @param rds_resource [Aws::RDS::Resource] An SDK for Ruby Amazon RDS resource.
# @return [Array, nil] List of all parameter groups, or nil if error.
def list_parameter_groups(rds_resource)
```



```

parameter_groups = []
rds_resource.db_parameter_groups.each do |p|
  parameter_groups.append({
    "name": p.db_parameter_group_name,
    "description": p.description
  })
end
parameter_groups
rescue Aws::Errors::ServiceError => e
  puts "Couldn't list parameter groups:\n #{e.message}"
end

```

- Untuk API detailnya, lihat [DescribeDBParameters](#) di AWS SDK for Ruby API Referensi.

DescribeDBSnapshots

Contoh kode berikut menunjukkan cara menggunakan `DescribeDBSnapshots`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```

require "aws-sdk-rds" # v2: require 'aws-sdk'

# List all Amazon Relational Database Service (Amazon RDS) DB instance
# snapshots.
#
# @param rds_resource [Aws::RDS::Resource] An SDK for Ruby Amazon RDS resource.
# @return instance_snapshots [Array, nil] All instance snapshots, or nil if error.
def list_instance_snapshots(rds_resource)
  instance_snapshots = []
  rds_resource.db_snapshots.each do |s|
    instance_snapshots.append({
      "id": s.snapshot_id,
      "status": s.status
    })
  end
end

```

```
instance_snapshots
rescue Aws::Errors::ServiceError => e
  puts "Couldn't list instance snapshots:\n #{e.message}"
end
```

- Untuk API detailnya, lihat [DescribeDBSnapshots](#) di AWS SDK for Ruby API Referensi.

Contoh nirserver

Menghubungkan ke RDS database Amazon dalam fungsi Lambda

Contoh kode berikut menunjukkan bagaimana menerapkan fungsi Lambda yang menghubungkan ke database. RDS Fungsi membuat permintaan database sederhana dan mengembalikan hasilnya.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di repositori [contoh Nirserver](#).

Menghubungkan ke RDS database Amazon dalam fungsi Lambda menggunakan Ruby.

```
# Ruby code here.

require 'aws-sdk-rds'
require 'json'
require 'mysql2'

def lambda_handler(event:, context:)
  endpoint = ENV['DBEndpoint'] # Add the endpoint without https"
  port = ENV['Port']           # 3306
  user = ENV['DBUser']
  region = ENV['DBRegion']     # 'us-east-1'
  db_name = ENV['DBName']

  credentials = Aws::Credentials.new(
    ENV['AWS_ACCESS_KEY_ID'],
    ENV['AWS_SECRET_ACCESS_KEY'],
```

```
ENV['AWS_SESSION_TOKEN']
)
rds_client = Aws::RDS::AuthTokenGenerator.new(
  region: region,
  credentials: credentials
)

token = rds_client.auth_token(
  endpoint: endpoint+ ':' + port,
  user_name: user,
  region: region
)

begin
  conn = Mysql2::Client.new(
    host: endpoint,
    username: user,
    password: token,
    port: port,
    database: db_name,
    sslca: '/var/task/global-bundle.pem',
    sslverify: true,
    enable_clear_text_plugin: true
  )
  a = 3
  b = 2
  result = conn.query("SELECT #{a} + #{b} AS sum").first['sum']
  puts result
  conn.close
  {
    statusCode: 200,
    body: result.to_json
  }
rescue => e
  puts "Database connection failed due to #{e}"
end
end
```

Contoh Amazon S3 menggunakan SDK untuk Ruby

Contoh kode berikut menunjukkan cara melakukan tindakan dan menerapkan skenario umum dengan menggunakan Amazon S3. AWS SDK for Ruby

Dasar-dasar adalah contoh kode yang menunjukkan kepada Anda bagaimana melakukan operasi penting dalam suatu layanan.

Tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Skenario adalah contoh kode yang menunjukkan kepada Anda bagaimana menyelesaikan tugas tertentu dengan memanggil beberapa fungsi dalam layanan atau dikombinasikan dengan yang lain AWS layanan.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Memulai

Halo Amazon S3

Contoh kode berikut ini menunjukkan cara memulai menggunakan Amazon S3.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
# frozen_string_literal: true

# S3Manager is a class responsible for managing S3 operations
# such as listing all S3 buckets in the current AWS account.
class S3Manager
  def initialize(client)
    @client = client
    @logger = Logger.new($stdout)
  end

  # Lists and prints all S3 buckets in the current AWS account.
  def list_buckets
```

```
@logger.info('Here are the buckets in your account:')

response = @client.list_buckets

if response.buckets.empty?
  @logger.info("You don't have any S3 buckets yet.")
else
  response.buckets.each do |bucket|
    @logger.info("- #{bucket.name}")
  end
end

rescue Aws::Errors::ServiceError => e
  @logger.error("Encountered an error while listing buckets: #{e.message}")
end

end

if $PROGRAM_NAME == __FILE__
  s3_client = Aws::S3::Client.new
  manager = S3Manager.new(s3_client)
  manager.list_buckets
end
```

- Untuk API detailnya, lihat [ListBuckets](#) di AWS SDK for Ruby API Referensi.

Topik

- [Hal-hal mendasar](#)
- [Tindakan](#)
- [Skenario](#)
- [Contoh nirservers](#)

Hal-hal mendasar

Pelajari dasar-dasarnya

Contoh kode berikut ini menunjukkan cara:

- Membuat bucket dan mengunggah file ke dalamnya.
- Mengunduh objek dari bucket.

- Menyalin objek ke subfolder di bucket.
- Membuat daftar objek dalam bucket.
- Menghapus objek bucket dan bucket tersebut.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
require "aws-sdk-s3"

# Wraps the getting started scenario actions.
class ScenarioGettingStarted
  attr_reader :s3_resource

  # @param s3_resource [Aws::S3::Resource] An Amazon S3 resource.
  def initialize(s3_resource)
    @s3_resource = s3_resource
  end

  # Creates a bucket with a random name in the currently configured account and
  # AWS Region.
  #
  # @return [Aws::S3::Bucket] The newly created bucket.
  def create_bucket
    bucket = @s3_resource.create_bucket(
      bucket: "doc-example-bucket-#{Random.uuid}",
      create_bucket_configuration: {
        location_constraint: "us-east-1" # Note: only certain regions permitted
      }
    )
    puts("Created demo bucket named #{bucket.name}.")
  rescue Aws::Errors::ServiceError => e
    puts("Tried and failed to create demo bucket.")
    puts("\t#{e.code}: #{e.message}")
    puts("\nCan't continue the demo without a bucket!")
    raise
  else
  end
end
```

```
    bucket
  end

  # Requests a file name from the user.
  #
  # @return The name of the file.
  def create_file
    File.open("demo.txt", w) { |f| f.write("This is a demo file.") }
  end

  # Uploads a file to an Amazon S3 bucket.
  #
  # @param bucket [Aws::S3::Bucket] The bucket object representing the upload
  destination
  # @return [Aws::S3::Object] The Amazon S3 object that contains the uploaded file.
  def upload_file(bucket)
    File.open("demo.txt", "w+") { |f| f.write("This is a demo file.") }
    s3_object = bucket.object(File.basename("demo.txt"))
    s3_object.upload_file("demo.txt")
    puts("Uploaded file demo.txt into bucket #{bucket.name} with key
  #{s3_object.key}.")
    rescue Aws::Errors::ServiceError => e
      puts("Couldn't upload file demo.txt to #{bucket.name}.")
      puts("\t#{e.code}: #{e.message}")
      raise
    else
      s3_object
    end

  # Downloads an Amazon S3 object to a file.
  #
  # @param s3_object [Aws::S3::Object] The object to download.
  def download_file(s3_object)
    puts("\nDo you want to download #{s3_object.key} to a local file (y/n)? ")
    answer = gets.chomp.downcase
    if answer == "y"
      puts("Enter a name for the downloaded file: ")
      file_name = gets.chomp
      s3_object.download_file(file_name)
      puts("Object #{s3_object.key} successfully downloaded to #{file_name}.")
    end
    rescue Aws::Errors::ServiceError => e
      puts("Couldn't download #{s3_object.key}.")
      puts("\t#{e.code}: #{e.message}")
    end
  end
end
```

```
    raise
  end

  # Copies an Amazon S3 object to a subfolder within the same bucket.
  #
  # @param source_object [Aws::S3::Object] The source object to copy.
  # @return [Aws::S3::Object, nil] The destination object.
  def copy_object(source_object)
    dest_object = nil
    puts("\nDo you want to copy #{source_object.key} to a subfolder in your bucket
(y/n)? ")
    answer = gets.chomp.downcase
    if answer == "y"
      dest_object = source_object.bucket.object("demo-folder/#{source_object.key}")
      dest_object.copy_from(source_object)
      puts("Copied #{source_object.key} to #{dest_object.key}.")
    end
  rescue Aws::Errors::ServiceError => e
    puts("Couldn't copy #{source_object.key}.")
    puts("\t#{e.code}: #{e.message}")
    raise
  else
    dest_object
  end

  # Lists the objects in an Amazon S3 bucket.
  #
  # @param bucket [Aws::S3::Bucket] The bucket to query.
  def list_objects(bucket)
    puts("\nYour bucket contains the following objects:")
    bucket.objects.each do |obj|
      puts("\t#{obj.key}")
    end
  rescue Aws::Errors::ServiceError => e
    puts("Couldn't list the objects in bucket #{bucket.name}.")
    puts("\t#{e.code}: #{e.message}")
    raise
  end

  # Deletes the objects in an Amazon S3 bucket and deletes the bucket.
  #
  # @param bucket [Aws::S3::Bucket] The bucket to empty and delete.
  def delete_bucket(bucket)
    puts("\nDo you want to delete all of the objects as well as the bucket (y/n)? ")
  end
end
```



```
    answer = gets.chomp.downcase
    if answer == "y"
      bucket.objects.batch_delete!
      bucket.delete
      puts("Emptied and deleted bucket #{bucket.name}.\n")
    end
  rescue Aws::Errors::ServiceError => e
    puts("Couldn't empty and delete bucket #{bucket.name}.")
    puts("\t#{e.code}: #{e.message}")
    raise
  end
end

# Runs the Amazon S3 getting started scenario.
def run_scenario(scenario)
  puts("-" * 88)
  puts("Welcome to the Amazon S3 getting started demo!")
  puts("-" * 88)

  bucket = scenario.create_bucket
  s3_object = scenario.upload_file(bucket)
  scenario.download_file(s3_object)
  scenario.copy_object(s3_object)
  scenario.list_objects(bucket)
  scenario.delete_bucket(bucket)

  puts("Thanks for watching!")
  puts("-" * 88)
rescue Aws::Errors::ServiceError
  puts("Something went wrong with the demo!")
end

run_scenario(ScenarioGettingStarted.new(Aws::S3::Resource.new)) if $PROGRAM_NAME ==
__FILE__
```

- Untuk API detailnya, lihat topik berikut di AWS SDK for Ruby API Referensi.
 - [CopyObject](#)
 - [CreateBucket](#)
 - [DeleteBucket](#)
 - [DeleteObjects](#)

- [GetObject](#)
- [ListObjectsV2](#)
- [PutObject](#)

Tindakan

CopyObject

Contoh kode berikut menunjukkan cara menggunakan `CopyObject`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

Salin objek.

```
require "aws-sdk-s3"

# Wraps Amazon S3 object actions.
class ObjectCopyWrapper
  attr_reader :source_object

  # @param source_object [Aws::S3::Object] An existing Amazon S3 object. This is
  # used as the source object for
  #                               copy actions.
  def initialize(source_object)
    @source_object = source_object
  end

  # Copy the source object to the specified target bucket and rename it with the
  # target key.
  # @param target_bucket [Aws::S3::Bucket] An existing Amazon S3 bucket where the
  # object is copied.
  # @param target_object_key [String] The key to give the copy of the object.
  # @return [Aws::S3::Object, nil] The copied object when successful; otherwise,
  nil.
```

```

def copy_object(target_bucket, target_object_key)
  @source_object.copy_to(bucket: target_bucket.name, key: target_object_key)
  target_bucket.object(target_object_key)
rescue Aws::Errors::ServiceError => e
  puts "Couldn't copy #{@source_object.key} to #{target_object_key}. Here's why:
#{e.message}"
end
end

# Example usage:
def run_demo
  source_bucket_name = "doc-example-bucket1"
  source_key = "my-source-file.txt"
  target_bucket_name = "doc-example-bucket2"
  target_key = "my-target-file.txt"

  source_bucket = Aws::S3::Bucket.new(source_bucket_name)
  wrapper = ObjectCopyWrapper.new(source_bucket.object(source_key))
  target_bucket = Aws::S3::Bucket.new(target_bucket_name)
  target_object = wrapper.copy_object(target_bucket, target_key)
  return unless target_object

  puts "Copied #{source_key} from #{source_bucket_name} to
#{target_object.bucket_name}:#{target_object.key}."
end

run_demo if $PROGRAM_NAME == __FILE__

```

Salin objek dan tambahkan enkripsi di sisi server ke objek tujuan.

```

require "aws-sdk-s3"

# Wraps Amazon S3 object actions.
class ObjectCopyEncryptWrapper
  attr_reader :source_object

  # @param source_object [Aws::S3::Object] An existing Amazon S3 object. This is
  # used as the source object for
  #           copy actions.
  def initialize(source_object)
    @source_object = source_object
  end
end

```

```
# Copy the source object to the specified target bucket, rename it with the target
key, and encrypt it.
#
# @param target_bucket [Aws::S3::Bucket] An existing Amazon S3 bucket where the
object is copied.
# @param target_object_key [String] The key to give the copy of the object.
# @return [Aws::S3::Object, nil] The copied object when successful; otherwise,
nil.
def copy_object(target_bucket, target_object_key, encryption)
  @source_object.copy_to(bucket: target_bucket.name, key: target_object_key,
server_side_encryption: encryption)
  target_bucket.object(target_object_key)
rescue Aws::Errors::ServiceError => e
  puts "Couldn't copy #{@source_object.key} to #{target_object_key}. Here's why:
#{e.message}"
end
end

# Example usage:
def run_demo
  source_bucket_name = "doc-example-bucket1"
  source_key = "my-source-file.txt"
  target_bucket_name = "doc-example-bucket2"
  target_key = "my-target-file.txt"
  target_encryption = "AES256"

  source_bucket = Aws::S3::Bucket.new(source_bucket_name)
  wrapper = ObjectCopyEncryptWrapper.new(source_bucket.object(source_key))
  target_bucket = Aws::S3::Bucket.new(target_bucket_name)
  target_object = wrapper.copy_object(target_bucket, target_key, target_encryption)
  return unless target_object

  puts "Copied #{source_key} from #{source_bucket_name} to
#{target_object.bucket_name}:#{target_object.key} and "\
    "encrypted the target with #{target_object.server_side_encryption}
encryption."
end

run_demo if $PROGRAM_NAME == __FILE__
```

- Untuk API detailnya, lihat [CopyObject](#) di AWS SDK for Ruby API Referensi.

CreateBucket

Contoh kode berikut menunjukkan cara menggunakan `CreateBucket`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
require "aws-sdk-s3"

# Wraps Amazon S3 bucket actions.
class BucketCreateWrapper
  attr_reader :bucket

  # @param bucket [Aws::S3::Bucket] An Amazon S3 bucket initialized with a name.
  # This is a client-side object until
  #
  #
  # create is called.
  def initialize(bucket)
    @bucket = bucket
  end

  # Creates an Amazon S3 bucket in the specified AWS Region.
  #
  # @param region [String] The Region where the bucket is created.
  # @return [Boolean] True when the bucket is created; otherwise, false.
  def create?(region)
    @bucket.create(create_bucket_configuration: { location_constraint: region })
    true
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't create bucket. Here's why: #{e.message}"
    false
  end

  # Gets the Region where the bucket is located.
  #
  # @return [String] The location of the bucket.
  def location
    if @bucket.nil?
      "None. You must create a bucket before you can get its location!"
    end
  end
end
```

```

    else
      @bucket.client.get_bucket_location(bucket: @bucket.name).location_constraint
    end
  rescue Aws::Errors::ServiceError => e
    "Couldn't get the location of #{@bucket.name}. Here's why: #{e.message}"
  end
end

# Example usage:
def run_demo
  region = "us-west-2"
  wrapper = BucketCreateWrapper.new(Aws::S3::Bucket.new("doc-example-bucket-
#{Random.uuid}"))
  return unless wrapper.create?(region)

  puts "Created bucket #{wrapper.bucket.name}."
  puts "Your bucket's region is: #{wrapper.location}"
end

run_demo if $PROGRAM_NAME == __FILE__

```

- Untuk API detailnya, lihat [CreateBucket](#) di AWS SDK for Ruby API Referensi.

DeleteBucket

Contoh kode berikut menunjukkan cara menggunakan `DeleteBucket`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```

# Deletes the objects in an Amazon S3 bucket and deletes the bucket.
#
# @param bucket [Aws::S3::Bucket] The bucket to empty and delete.
def delete_bucket(bucket)
  puts("\nDo you want to delete all of the objects as well as the bucket (y/n)? ")

```

```

    answer = gets.chomp.downcase
    if answer == "y"
      bucket.objects.batch_delete!
      bucket.delete
      puts("Emptied and deleted bucket #{bucket.name}.\n")
    end
  rescue Aws::Errors::ServiceError => e
    puts("Couldn't empty and delete bucket #{bucket.name}.")
    puts("\t#{e.code}: #{e.message}")
    raise
  end
end

```

- Untuk API detailnya, lihat [DeleteBucket](#) di AWS SDK for Ruby API Referensi.

DeleteBucketCors

Contoh kode berikut menunjukkan cara menggunakan `DeleteBucketCors`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```

require "aws-sdk-s3"

# Wraps Amazon S3 bucket CORS configuration.
class BucketCorsWrapper
  attr_reader :bucket_cors

  # @param bucket_cors [Aws::S3::BucketCors] A bucket CORS object configured with an
  # existing bucket.
  def initialize(bucket_cors)
    @bucket_cors = bucket_cors
  end

  # Deletes the CORS configuration of a bucket.
  #

```

```
# @return [Boolean] True if the CORS rules were deleted; otherwise, false.
def delete_cors
  @bucket_cors.delete
  true
rescue Aws::Errors::ServiceError => e
  puts "Couldn't delete CORS rules for #{@bucket_cors.bucket.name}. Here's why:
#{e.message}"
  false
end
end
```

- Untuk API detailnya, lihat [DeleteBucketCors](#) di AWS SDK for Ruby API Referensi.

DeleteBucketPolicy

Contoh kode berikut menunjukkan cara menggunakan `DeleteBucketPolicy`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
# Wraps an Amazon S3 bucket policy.
class BucketPolicyWrapper
  attr_reader :bucket_policy

  # @param bucket_policy [Aws::S3::BucketPolicy] A bucket policy object configured
  # with an existing bucket.
  def initialize(bucket_policy)
    @bucket_policy = bucket_policy
  end

  def delete_policy
    @bucket_policy.delete
    true
  rescue Aws::Errors::ServiceError => e
```



```

    puts "Couldn't delete the policy from #{@bucket_policy.bucket.name}. Here's why:
#{e.message}"
    false
  end
end
end

```

- Untuk API detailnya, lihat [DeleteBucketPolicy](#) di AWS SDK for Ruby API Referensi.

Delete Objects

Contoh kode berikut menunjukkan cara menggunakan `DeleteObjects`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```

# Deletes the objects in an Amazon S3 bucket and deletes the bucket.
#
# @param bucket [Aws::S3::Bucket] The bucket to empty and delete.
def delete_bucket(bucket)
  puts("\nDo you want to delete all of the objects as well as the bucket (y/n)? ")
  answer = gets.chomp.downcase
  if answer == "y"
    bucket.objects.batch_delete!
    bucket.delete
    puts("Emptied and deleted bucket #{bucket.name}.\n")
  end
rescue Aws::Errors::ServiceError => e
  puts("Couldn't empty and delete bucket #{bucket.name}.")
  puts("\t#{e.code}: #{e.message}")
  raise
end

```

- Untuk API detailnya, lihat [DeleteObjects](#) di AWS SDK for Ruby API Referensi.

GetBucketCors

Contoh kode berikut menunjukkan cara menggunakan `GetBucketCors`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
require "aws-sdk-s3"

# Wraps Amazon S3 bucket CORS configuration.
class BucketCorsWrapper
  attr_reader :bucket_cors

  # @param bucket_cors [Aws::S3::BucketCors] A bucket CORS object configured with an
  # existing bucket.
  def initialize(bucket_cors)
    @bucket_cors = bucket_cors
  end

  # Gets the CORS configuration of a bucket.
  #
  # @return [Aws::S3::Type::GetBucketCorsOutput, nil] The current CORS configuration
  # for the bucket.
  def get_cors
    @bucket_cors.data
    rescue Aws::Errors::ServiceError => e
      puts "Couldn't get CORS configuration for #{@bucket_cors.bucket.name}. Here's
  why: #{e.message}"
      nil
    end
  end
end
```

- Untuk API detailnya, lihat [GetBucketCors](#) di AWS SDK for Ruby API Referensi.

GetBucketPolicy

Contoh kode berikut menunjukkan cara menggunakan `GetBucketPolicy`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
# Wraps an Amazon S3 bucket policy.
class BucketPolicyWrapper
  attr_reader :bucket_policy

  # @param bucket_policy [Aws::S3::BucketPolicy] A bucket policy object configured
  # with an existing bucket.
  def initialize(bucket_policy)
    @bucket_policy = bucket_policy
  end

  # Gets the policy of a bucket.
  #
  # @return [Aws::S3::GetBucketPolicyOutput, nil] The current bucket policy.
  def get_policy
    policy = @bucket_policy.data.policy
    policy.respond_to?(:read) ? policy.read : policy
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't get the policy for #{@bucket_policy.bucket.name}. Here's why:
    #{e.message}"
    nil
  end
end
```

- Untuk API detailnya, lihat [GetBucketPolicy](#) di AWS SDK for Ruby API Referensi.

GetObject

Contoh kode berikut menunjukkan cara menggunakan `GetObject`.

SDK untuk Ruby

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

Dapatkan objek.

```
require "aws-sdk-s3"

# Wraps Amazon S3 object actions.
class ObjectGetWrapper
  attr_reader :object

  # @param object [Aws::S3::Object] An existing Amazon S3 object.
  def initialize(object)
    @object = object
  end

  # Gets the object directly to a file.
  #
  # @param target_path [String] The path to the file where the object is downloaded.
  # @return [Aws::S3::Types::GetObjectOutput, nil] The retrieved object data if
  # successful; otherwise nil.
  def get_object(target_path)
    @object.get(response_target: target_path)
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't get object #{@object.key}. Here's why: #{e.message}"
  end
end

# Example usage:
def run_demo
  bucket_name = "doc-example-bucket"
  object_key = "my-object.txt"
  target_path = "my-object-as-file.txt"

  wrapper = ObjectGetWrapper.new(Aws::S3::Object.new(bucket_name, object_key))
  obj_data = wrapper.get_object(target_path)
  return unless obj_data
end
```

```
puts "Object #{object_key} (#{obj_data.content_length} bytes) downloaded to  
#{target_path}."  
end  
  
run_demo if $PROGRAM_NAME == __FILE__
```

Dapatkan objek dan laporkan status enkripsi di sisi servernya.

```
require "aws-sdk-s3"  
  
# Wraps Amazon S3 object actions.  
class ObjectGetEncryptionWrapper  
  attr_reader :object  
  
  # @param object [Aws::S3::Object] An existing Amazon S3 object.  
  def initialize(object)  
    @object = object  
  end  
  
  # Gets the object into memory.  
  #  
  # @return [Aws::S3::Types::GetObjectOutput, nil] The retrieved object data if  
  # successful; otherwise nil.  
  def get_object  
    @object.get  
    rescue Aws::Errors::ServiceError => e  
      puts "Couldn't get object #{@object.key}. Here's why: #{e.message}"  
    end  
  end  
end  
  
# Example usage:  
def run_demo  
  bucket_name = "doc-example-bucket"  
  object_key = "my-object.txt"  
  
  wrapper = ObjectGetEncryptionWrapper.new(Aws::S3::Object.new(bucket_name,  
    object_key))  
  obj_data = wrapper.get_object  
  return unless obj_data  
  
  encryption = obj_data.server_side_encryption.nil? ? "no" :  
  obj_data.server_side_encryption
```

```
puts "Object #{object_key} uses #{encryption} encryption."
end

run_demo if $PROGRAM_NAME == __FILE__
```

- Untuk API detailnya, lihat [GetObject](#) di AWS SDK for Ruby API Referensi.

HeadObject

Contoh kode berikut menunjukkan cara menggunakan `HeadObject`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
require "aws-sdk-s3"

# Wraps Amazon S3 object actions.
class ObjectExistsWrapper
  attr_reader :object

  # @param object [Aws::S3::Object] An Amazon S3 object.
  def initialize(object)
    @object = object
  end

  # Checks whether the object exists.
  #
  # @return [Boolean] True if the object exists; otherwise false.
  def exists?
    @object.exists?
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't check existence of object #{@object.bucket.name}:#{@object.key}.
    Here's why: #{e.message}"
    false
  end
end
```

```
# Example usage:
def run_demo
  bucket_name = "doc-example-bucket"
  object_key = "my-object.txt"

  wrapper = ObjectExistsWrapper.new(Aws::S3::Object.new(bucket_name, object_key))
  exists = wrapper.exists?

  puts "Object #{object_key} #{exists ? 'does' : 'does not'} exist."
end

run_demo if $PROGRAM_NAME == __FILE__
```

- Untuk API detailnya, lihat [HeadObject](#) di AWS SDK for Ruby API Referensi.

ListBuckets

Contoh kode berikut menunjukkan cara menggunakan `ListBuckets`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
require "aws-sdk-s3"

# Wraps Amazon S3 resource actions.
class BucketListWrapper
  attr_reader :s3_resource

  # @param s3_resource [Aws::S3::Resource] An Amazon S3 resource.
  def initialize(s3_resource)
    @s3_resource = s3_resource
  end

  # Lists buckets for the current account.
  #
```

```
# @param count [Integer] The maximum number of buckets to list.
def list_buckets(count)
  puts "Found these buckets:"
  @s3_resource.buckets.each do |bucket|
    puts "\t#{bucket.name}"
    count -= 1
    break if count.zero?
  end
  true
rescue Aws::Errors::ServiceError => e
  puts "Couldn't list buckets. Here's why: #{e.message}"
  false
end
end

# Example usage:
def run_demo
  wrapper = BucketListWrapper.new(Aws::S3::Resource.new)
  wrapper.list_buckets(25)
end

run_demo if $PROGRAM_NAME == __FILE__
```

- Untuk API detailnya, lihat [ListBuckets](#) di AWS SDK for Ruby API Referensi.

ListObjectsV2

Contoh kode berikut menunjukkan cara menggunakan ListObjectsV2.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
require "aws-sdk-s3"

# Wraps Amazon S3 bucket actions.
class BucketListObjectsWrapper
```



```
attr_reader :bucket

# @param bucket [Aws::S3::Bucket] An existing Amazon S3 bucket.
def initialize(bucket)
  @bucket = bucket
end

# Lists object in a bucket.
#
# @param max_objects [Integer] The maximum number of objects to list.
# @return [Integer] The number of objects listed.
def list_objects(max_objects)
  count = 0
  puts "The objects in #{@bucket.name} are:"
  @bucket.objects.each do |obj|
    puts "\t#{obj.key}"
    count += 1
    break if count == max_objects
  end
  count
rescue Aws::Errors::ServiceError => e
  puts "Couldn't list objects in bucket #{bucket.name}. Here's why: #{e.message}"
  0
end

# Example usage:
def run_demo
  bucket_name = "doc-example-bucket"

  wrapper = BucketListObjectsWrapper.new(Aws::S3::Bucket.new(bucket_name))
  count = wrapper.list_objects(25)
  puts "Listed #{count} objects."
end

run_demo if $PROGRAM_NAME == __FILE__
```

- Untuk API detailnya, lihat [ListObjectsV2](#) di AWS SDK for Ruby API Referensi.

PutBucketCors

Contoh kode berikut menunjukkan cara menggunakan PutBucketCors.

SDK untuk Ruby

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
require "aws-sdk-s3"

# Wraps Amazon S3 bucket CORS configuration.
class BucketCorsWrapper
  attr_reader :bucket_cors

  # @param bucket_cors [Aws::S3::BucketCors] A bucket CORS object configured with an
  # existing bucket.
  def initialize(bucket_cors)
    @bucket_cors = bucket_cors
  end

  # Sets CORS rules on a bucket.
  #
  # @param allowed_methods [Array<String>] The types of HTTP requests to allow.
  # @param allowed_origins [Array<String>] The origins to allow.
  # @returns [Boolean] True if the CORS rules were set; otherwise, false.
  def set_cors(allowed_methods, allowed_origins)
    @bucket_cors.put(
      cors_configuration: {
        cors_rules: [
          {
            allowed_methods: allowed_methods,
            allowed_origins: allowed_origins,
            allowed_headers: %w[*],
            max_age_seconds: 3600
          }
        ]
      }
    )
    true
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't set CORS rules for #{@bucket_cors.bucket.name}. Here's why:
    #{e.message}"
  end
end
```

```
    false
  end

end
```

- Untuk API detailnya, lihat [PutBucketCors](#) di AWS SDK for Ruby API Referensi.

PutBucketPolicy

Contoh kode berikut menunjukkan cara menggunakan `PutBucketPolicy`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
# Wraps an Amazon S3 bucket policy.
class BucketPolicyWrapper
  attr_reader :bucket_policy

  # @param bucket_policy [Aws::S3::BucketPolicy] A bucket policy object configured
  # with an existing bucket.
  def initialize(bucket_policy)
    @bucket_policy = bucket_policy
  end

  # Sets a policy on a bucket.
  #
  def set_policy(policy)
    @bucket_policy.put(policy: policy)
    true
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't set the policy for #{@bucket_policy.bucket.name}. Here's why:
    #{e.message}"
    false
  end
end
```

- Untuk API detailnya, lihat [PutBucketPolicy](#) di AWS SDK for Ruby API Referensi.

PutBucketWebsite

Contoh kode berikut menunjukkan cara menggunakan `PutBucketWebsite`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
require "aws-sdk-s3"

# Wraps Amazon S3 bucket website actions.
class BucketWebsiteWrapper
  attr_reader :bucket_website

  # @param bucket_website [Aws::S3::BucketWebsite] A bucket website object
  # configured with an existing bucket.
  def initialize(bucket_website)
    @bucket_website = bucket_website
  end

  # Sets a bucket as a static website.
  #
  # @param index_document [String] The name of the index document for the website.
  # @param error_document [String] The name of the error document to show for 4XX
  # errors.
  # @return [Boolean] True when the bucket is configured as a website; otherwise,
  # false.
  def set_website(index_document, error_document)
    @bucket_website.put(
      website_configuration: {
        index_document: { suffix: index_document },
        error_document: { key: error_document }
      }
    )
  end
end
```

```
    true
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't configure #{@bucket_website.bucket.name} as a website. Here's
why: #{e.message}"
    false
  end
end
end

# Example usage:
def run_demo
  bucket_name = "doc-example-bucket"
  index_document = "index.html"
  error_document = "404.html"

  wrapper = BucketWebsiteWrapper.new(Aws::S3::BucketWebsite.new(bucket_name))
  return unless wrapper.set_website(index_document, error_document)

  puts "Successfully configured bucket #{bucket_name} as a static website."
end

run_demo if $PROGRAM_NAME == __FILE__
```

- Untuk API detailnya, lihat [PutBucketWebsite](#) di AWS SDK for Ruby API Referensi.

PutObject

Contoh kode berikut menunjukkan cara menggunakan `PutObject`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

Unggah file menggunakan pengunggah terkelola (`Object.upload_file`).

```
require "aws-sdk-s3"

# Wraps Amazon S3 object actions.
```

```
class ObjectUploadFileWrapper
  attr_reader :object

  # @param object [Aws::S3::Object] An existing Amazon S3 object.
  def initialize(object)
    @object = object
  end

  # Uploads a file to an Amazon S3 object by using a managed uploader.
  #
  # @param file_path [String] The path to the file to upload.
  # @return [Boolean] True when the file is uploaded; otherwise false.
  def upload_file(file_path)
    @object.upload_file(file_path)
    true
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't upload file #{file_path} to #{@object.key}. Here's why:
#{e.message}"
    false
  end
end

# Example usage:
def run_demo
  bucket_name = "doc-example-bucket"
  object_key = "my-uploaded-file"
  file_path = "object_upload_file.rb"

  wrapper = ObjectUploadFileWrapper.new(Aws::S3::Object.new(bucket_name,
object_key))
  return unless wrapper.upload_file(file_path)

  puts "File #{file_path} successfully uploaded to #{bucket_name}:#{object_key}."
end

run_demo if $PROGRAM_NAME == __FILE__
```

Unggah file menggunakan Object.put.

```
require "aws-sdk-s3"

# Wraps Amazon S3 object actions.
```

```

class ObjectPutWrapper
  attr_reader :object

  # @param object [Aws::S3::Object] An existing Amazon S3 object.
  def initialize(object)
    @object = object
  end

  def put_object(source_file_path)
    File.open(source_file_path, "rb") do |file|
      @object.put(body: file)
    end
    true
  rescue Aws::Errors::ServiceError => e
    puts "Couldn't put #{source_file_path} to #{object.key}. Here's why:
#{e.message}"
    false
  end
end

# Example usage:
def run_demo
  bucket_name = "doc-example-bucket"
  object_key = "my-object-key"
  file_path = "my-local-file.txt"

  wrapper = ObjectPutWrapper.new(Aws::S3::Object.new(bucket_name, object_key))
  success = wrapper.put_object(file_path)
  return unless success

  puts "Put file #{file_path} into #{object_key} in #{bucket_name}."
end

run_demo if $PROGRAM_NAME == __FILE__

```

Unggah file menggunakan `Object.put` dan tambahkan enkripsi di sisi server.

```

require "aws-sdk-s3"

# Wraps Amazon S3 object actions.
class ObjectPutSseWrapper
  attr_reader :object

```

```
# @param object [Aws::S3::Object] An existing Amazon S3 object.
def initialize(object)
  @object = object
end

def put_object_encrypted(object_content, encryption)
  @object.put(body: object_content, server_side_encryption: encryption)
  true
rescue Aws::Errors::ServiceError => e
  puts "Couldn't put your content to #{object.key}. Here's why: #{e.message}"
  false
end

# Example usage:
def run_demo
  bucket_name = "doc-example-bucket"
  object_key = "my-encrypted-content"
  object_content = "This is my super-secret content."
  encryption = "AES256"

  wrapper = ObjectPutSseWrapper.new(Aws::S3::Object.new(bucket_name,
object_content))
  return unless wrapper.put_object_encrypted(object_content, encryption)

  puts "Put your content into #{bucket_name}:#{object_key} and encrypted it with
#{encryption}."
end

run_demo if $PROGRAM_NAME == __FILE__
```

- Untuk API detailnya, lihat [PutObject](#) di AWS SDK for Ruby API Referensi.

Skenario

Buat presigned URL

Contoh kode berikut menunjukkan cara membuat presigned URL untuk Amazon S3 dan mengunggah objek.

SDK untuk Ruby

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
require "aws-sdk-s3"
require "net/http"

# Creates a presigned URL that can be used to upload content to an object.
#
# @param bucket [Aws::S3::Bucket] An existing Amazon S3 bucket.
# @param object_key [String] The key to give the uploaded object.
# @return [URI, nil] The parsed URI if successful; otherwise nil.
def get_presigned_url(bucket, object_key)
  url = bucket.object(object_key).presigned_url(:put)
  puts "Created presigned URL: #{url}"
  URI(url)
rescue Aws::Errors::ServiceError => e
  puts "Couldn't create presigned URL for #{bucket.name}:#{object_key}. Here's why:
  #{e.message}"
end

# Example usage:
def run_demo
  bucket_name = "doc-example-bucket"
  object_key = "my-file.txt"
  object_content = "This is the content of my-file.txt."

  bucket = Aws::S3::Bucket.new(bucket_name)
  presigned_url = get_presigned_url(bucket, object_key)
  return unless presigned_url

  response = Net::HTTP.start(presigned_url.host) do |http|
    http.send_request("PUT", presigned_url.request_uri, object_content,
"content_type" => "")
  end

  case response
  when Net::HTTPSuccess
```

```
    puts "Content uploaded!"
  else
    puts response.value
  end
end
end

run_demo if $PROGRAM_NAME == __FILE__
```

Contoh nirserver

Menginvokasi fungsi Lambda dari pemicu Amazon S3

Contoh kode berikut menunjukkan cara mengimplementasikan fungsi Lambda yang menerima peristiwa yang dipicu dengan mengunggah objek ke bucket S3. Fungsi ini mengambil nama bucket S3 dan kunci objek dari parameter event dan memanggil Amazon API S3 untuk mengambil dan mencatat jenis konten objek.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di repositori [contoh Nirserver](#).

Mengonsumsi acara S3 dengan Lambda menggunakan Ruby.

```
require 'json'
require 'uri'
require 'aws-sdk'

puts 'Loading function'

def lambda_handler(event:, context:)
  s3 = Aws::S3::Client.new(region: 'region') # Your AWS region
  # puts "Received event: #{JSON.dump(event)}"

  # Get the object from the event and show its content type
  bucket = event['Records'][0]['s3']['bucket']['name']
```

```
key = URI.decode_www_form_component(event['Records'][0]['s3']['object']['key'],
Encoding::UTF_8)
begin
  response = s3.get_object(bucket: bucket, key: key)
  puts "CONTENT TYPE: #{response.content_type}"
  return response.content_type
rescue StandardError => e
  puts e.message
  puts "Error getting object #{key} from bucket #{bucket}. Make sure they exist
and your bucket is in the same region as this function."
  raise e
end
end
```

SESContoh Amazon menggunakan SDK untuk Ruby

Contoh kode berikut menunjukkan kepada Anda cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan AWS SDK for Ruby With AmazonSES.

Tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

GetIdentityVerificationAttributes

Contoh kode berikut menunjukkan cara menggunakan `GetIdentityVerificationAttributes`.

SDK untuk Ruby

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
require "aws-sdk-ses" # v2: require 'aws-sdk'

# Create client in us-west-2 region
# Replace us-west-2 with the AWS Region you're using for Amazon SES.
client = Aws::SES::Client.new(region: "us-west-2")

# Get up to 1000 identities
ids = client.list_identities({
  identity_type: "EmailAddress"
})

ids.identities.each do |email|
  attrs = client.get_identity_verification_attributes({
    identities: [email]
  })

  status = attrs.verification_attributes[email].verification_status

  # Display email addresses that have been verified
  if status == "Success"
    puts email
  end
end
```

- Untuk API detailnya, lihat [GetIdentityVerificationAttributes](#) di AWS SDK for Ruby API Referensi.

ListIdentities

Contoh kode berikut menunjukkan cara menggunakan `ListIdentities`.

SDK untuk Ruby

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
require "aws-sdk-ses" # v2: require 'aws-sdk'

# Create client in us-west-2 region
# Replace us-west-2 with the AWS Region you're using for Amazon SES.
client = Aws::SES::Client.new(region: "us-west-2")

# Get up to 1000 identities
ids = client.list_identities({
  identity_type: "EmailAddress"
})

ids.identities.each do |email|
  attrs = client.get_identity_verification_attributes({
    identities: [email]
  })

  status = attrs.verification_attributes[email].verification_status

  # Display email addresses that have been verified
  if status == "Success"
    puts email
  end
end
```

- Untuk API detailnya, lihat [ListIdentities](#) di AWS SDK for Ruby API Referensi.

SendEmail

Contoh kode berikut menunjukkan cara menggunakan `SendEmail`.

SDK untuk Ruby

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
require "aws-sdk-ses" # v2: require 'aws-sdk'

# Replace sender@example.com with your "From" address.
# This address must be verified with Amazon SES.
sender = "sender@example.com"

# Replace recipient@example.com with a "To" address. If your account
# is still in the sandbox, this address must be verified.
recipient = "recipient@example.com"

# Specify a configuration set. To use a configuration
# set, uncomment the next line and line 74.
# configsetname = "ConfigSet"

# The subject line for the email.
subject = "Amazon SES test (AWS SDK for Ruby)"

# The HTML body of the email.
htmlbody =
  "<h1>Amazon SES test (AWS SDK for Ruby)</h1>\"
  '<p>This email was sent with <a href="https://aws.amazon.com/ses/">\'
  'Amazon SES</a> using the <a href="https://aws.amazon.com/sdk-for-ruby/">\'
  "AWS SDK for Ruby</a>."

# The email body for recipients with non-HTML email clients.
textbody = "This email was sent with Amazon SES using the AWS SDK for Ruby."

# Specify the text encoding scheme.
encoding = "UTF-8"

# Create a new SES client in the us-west-2 region.
# Replace us-west-2 with the AWS Region you're using for Amazon SES.
ses = Aws::SES::Client.new(region: "us-west-2")
```

```
# Try to send the email.
begin
  # Provide the contents of the email.
  ses.send_email(
    destination: {
      to_addresses: [
        recipient
      ]
    },
    message: {
      body: {
        html: {
          charset: encoding,
          data: htmlbody
        },
        text: {
          charset: encoding,
          data: textbody
        }
      },
      subject: {
        charset: encoding,
        data: subject
      }
    },
    source: sender,
    # Uncomment the following line to use a configuration set.
    # configuration_set_name: configsetname,
  )

  puts "Email sent to " + recipient

# If something goes wrong, display an error message.
rescue Aws::SES::Errors::ServiceError => error
  puts "Email not sent. Error message: #{error}"
end
```

- Untuk API detailnya, lihat [SendEmail](#) di AWS SDK for Ruby API Referensi.

VerifyEmailIdentity

Contoh kode berikut menunjukkan cara menggunakan `VerifyEmailIdentity`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
require "aws-sdk-ses" # v2: require 'aws-sdk'

# Replace recipient@example.com with a "To" address.
recipient = "recipient@example.com"

# Create a new SES resource in the us-west-2 region.
# Replace us-west-2 with the AWS Region you're using for Amazon SES.
ses = Aws::SES::Client.new(region: "us-west-2")

# Try to verify email address.
begin
  ses.verify_email_identity({
    email_address: recipient
  })

  puts "Email sent to " + recipient

# If something goes wrong, display an error message.
rescue Aws::SES::Errors::ServiceError => error
  puts "Email not sent. Error message: #{error}"
end
```

- Untuk API detailnya, lihat [VerifyEmailIdentity](#) di AWS SDK for Ruby API Referensi.

Amazon SES API v2 contoh menggunakan SDK untuk Ruby

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan AWS SDK for Ruby with Amazon SES API v2.

Tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

SendEmail

Contoh kode berikut menunjukkan cara menggunakan `SendEmail`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
require "aws-sdk-sesv2"
require_relative "config" # Recipient and sender email addresses.

# Set up the SESv2 client.
client = Aws::SESV2::Client.new(region: AWS_REGION)

def send_email(client, sender_email, recipient_email)
  response = client.send_email(
    {
      from_email_address: sender_email,
```

```
destination: {
  to_addresses: [recipient_email]
},
content: {
  simple: {
    subject: {
      data: "Test email subject"
    },
    body: {
      text: {
        data: "Test email body"
      }
    }
  }
}
)
puts "Email sent from #{SENDER_EMAIL} to #{RECIPIENT_EMAIL} with message ID:
#{response.message_id}"
end

send_email(client, SENDER_EMAIL, RECIPIENT_EMAIL)
```

- Untuk API detailnya, lihat [SendEmail](#) di AWS SDK for Ruby API Referensi.

SNSSContoh Amazon menggunakan SDK untuk Ruby

Contoh kode berikut menunjukkan kepada Anda cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan AWS SDK for Ruby With AmazonSNS.

Tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)
- [Contoh nirserver](#)

Tindakan

CreateTopic

Contoh kode berikut menunjukkan cara menggunakan `CreateTopic`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
# This class demonstrates how to create an Amazon Simple Notification Service (SNS)
# topic.
class SNSTopicCreator
  # Initializes an SNS client.
  #
  # Utilizes the default AWS configuration for region and credentials.
  def initialize
    @sns_client = Aws::SNS::Client.new
  end

  # Attempts to create an SNS topic with the specified name.
  #
  # @param topic_name [String] The name of the SNS topic to create.
  # @return [Boolean] true if the topic was successfully created, false otherwise.
  def create_topic(topic_name)
    @sns_client.create_topic(name: topic_name)
    puts "The topic '#{topic_name}' was successfully created."
    true
  rescue Aws::SNS::Errors::ServiceError => e
    # Handles SNS service errors gracefully.
    puts "Error while creating the topic named '#{topic_name}': #{e.message}"
    false
  end
end

# Example usage:
if $PROGRAM_NAME == __FILE__
  topic_name = "YourTopicName" # Replace with your topic name
end
```

```
sns_topic_creator = SNS::TopicCreator.new

puts "Creating the topic '#{topic_name}'..."
unless sns_topic_creator.create_topic(topic_name)
  puts "The topic was not created. Stopping program."
  exit 1
end
end
```

- Untuk informasi selengkapnya, silakan lihat [Panduan Developer AWS SDK for Ruby](#).
- Untuk API detailnya, lihat [CreateTopic](#) di AWS SDK for Ruby API Referensi.

ListSubscriptions

Contoh kode berikut menunjukkan cara menggunakan `ListSubscriptions`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
# This class demonstrates how to list subscriptions to an Amazon Simple Notification
Service (SNS) topic
class SnsSubscriptionLister
  def initialize(sns_client)
    @sns_client = sns_client
    @logger = Logger.new($stdout)
  end

  # Lists subscriptions for a given SNS topic
  # @param topic_arn [String] The ARN of the SNS topic
  # @return [Types::ListSubscriptionsResponse] subscriptions: The response object
  def list_subscriptions(topic_arn)
    @logger.info("Listing subscriptions for topic: #{topic_arn}")
    subscriptions = @sns_client.list_subscriptions_by_topic(topic_arn: topic_arn)
    subscriptions.subscriptions.each do |subscription|
      @logger.info("Subscription endpoint: #{subscription.endpoint}")
    end
  end
end
```

```
    end
    subscriptions
  rescue Aws::SNS::Errors::ServiceError => e
    @logger.error("Error listing subscriptions: #{e.message}")
    raise
  end
end
end

# Example usage:
if $PROGRAM_NAME == __FILE__
  sns_client = Aws::SNS::Client.new
  topic_arn = "SNS_TOPIC_ARN" # Replace with your SNS topic ARN
  lister = SnsSubscriptionLister.new(sns_client)

  begin
    lister.list_subscriptions(topic_arn)
  rescue StandardError => e
    puts "Failed to list subscriptions: #{e.message}"
    exit 1
  end
end
end
```

- Untuk informasi selengkapnya, silakan lihat [Panduan Developer AWS SDK for Ruby](#).
- Untuk API detailnya, lihat [ListSubscriptions](#) di AWS SDK for Ruby API Referensi.

ListTopics

Contoh kode berikut menunjukkan cara menggunakan `ListTopics`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
require "aws-sdk-sns" # v2: require 'aws-sdk'
```

```
def list_topics?(sns_client)
  sns_client.topics.each do |topic|
    puts topic.arn
  rescue StandardError => e
    puts "Error while listing the topics: #{e.message}"
  end
end

def run_me

  region = "REGION"
  sns_client = Aws::SNS::Resource.new(region: region)

  puts "Listing the topics."

  if list_topics?(sns_client)
  else
    puts "The bucket was not created. Stopping program."
    exit 1
  end
end

# Example usage:
run_me if $PROGRAM_NAME == __FILE__
```

- Untuk informasi selengkapnya, silakan lihat [Panduan Developer AWS SDK for Ruby](#).
- Untuk API detailnya, lihat [ListTopics](#) di AWS SDK for Ruby API Referensi.

Publish

Contoh kode berikut menunjukkan cara menggunakan `Publish`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
# Service class for sending messages using Amazon Simple Notification Service (SNS)
class SnsMessageSender
  # Initializes the SnsMessageSender with an SNS client
  #
  # @param sns_client [Aws::SNS::Client] The SNS client
  def initialize(sns_client)
    @sns_client = sns_client
    @logger = Logger.new($stdout)
  end

  # Sends a message to a specified SNS topic
  #
  # @param topic_arn [String] The ARN of the SNS topic
  # @param message [String] The message to send
  # @return [Boolean] true if message was successfully sent, false otherwise
  def send_message(topic_arn, message)
    @sns_client.publish(topic_arn: topic_arn, message: message)
    @logger.info("Message sent successfully to #{topic_arn}.")
    true
  rescue Aws::SNS::Errors::ServiceError => e
    @logger.error("Error while sending the message: #{e.message}")
    false
  end
end

# Example usage:
if $PROGRAM_NAME == __FILE__
  topic_arn = "SNS_TOPIC_ARN" # Should be replaced with a real topic ARN
  message = "MESSAGE"        # Should be replaced with the actual message content

  sns_client = Aws::SNS::Client.new
  message_sender = SnsMessageSender.new(sns_client)

  @logger.info("Sending message.")
  unless message_sender.send_message(topic_arn, message)
    @logger.error("Message sending failed. Stopping program.")
    exit 1
  end
end
```

- Untuk informasi selengkapnya, silakan lihat [Panduan Developer AWS SDK for Ruby](#).
- Untuk API detailnya, lihat [Publikasikan](#) di AWS SDK for Ruby API Referensi.

SetTopicAttributes

Contoh kode berikut menunjukkan cara menggunakan `SetTopicAttributes`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
# Service class to enable an SNS resource with a specified policy
class SnsResourceEnabler
  # Initializes the SnsResourceEnabler with an SNS resource client
  #
  # @param sns_resource [Aws::SNS::Resource] The SNS resource client
  def initialize(sns_resource)
    @sns_resource = sns_resource
    @logger = Logger.new($stdout)
  end

  # Sets a policy on a specified SNS topic
  #
  # @param topic_arn [String] The ARN of the SNS topic
  # @param resource_arn [String] The ARN of the resource to include in the policy
  # @param policy_name [String] The name of the policy attribute to set
  def enable_resource(topic_arn, resource_arn, policy_name)
    policy = generate_policy(topic_arn, resource_arn)
    topic = @sns_resource.topic(topic_arn)

    topic.set_attributes({
      attribute_name: policy_name,
      attribute_value: policy
    })
    @logger.info("Policy #{policy_name} set successfully for topic #{topic_arn}.")
  rescue Aws::SNS::Errors::ServiceError => e
    @logger.error("Failed to set policy: #{e.message}")
  end

  private

  # Generates a policy string with dynamic resource ARNs
```



```
#
# @param topic_arn [String] The ARN of the SNS topic
# @param resource_arn [String] The ARN of the resource
# @return [String] The policy as a JSON string
def generate_policy(topic_arn, resource_arn)
  {
    Version: "2008-10-17",
    Id: "__default_policy_ID",
    Statement: [{
      Sid: "__default_statement_ID",
      Effect: "Allow",
      Principal: { "AWS": "*" },
      Action: ["SNS:Publish"],
      Resource: topic_arn,
      Condition: {
        ArnEquals: {
          "AWS:SourceArn": resource_arn
        }
      }
    }]
  }.to_json
end

# Example usage:
if $PROGRAM_NAME == __FILE__
  topic_arn = "MY_TOPIC_ARN" # Should be replaced with a real topic ARN
  resource_arn = "MY_RESOURCE_ARN" # Should be replaced with a real resource ARN
  policy_name = "POLICY_NAME" # Typically, this is "Policy"

  sns_resource = Aws::SNS::Resource.new
  enabler = SnsResourceEnabler.new(sns_resource)

  enabler.enable_resource(topic_arn, resource_arn, policy_name)
end
```

- Untuk informasi selengkapnya, silakan lihat [Panduan Developer AWS SDK for Ruby](#).
- Untuk API detailnya, lihat [SetTopicAttributes](#) di AWS SDK for Ruby API Referensi.

Subscribe

Contoh kode berikut menunjukkan cara menggunakan `Subscribe`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

Berlangganan alamat email ke suatu topik.

```
require "aws-sdk-sns"
require "logger"

# Represents a service for creating subscriptions in Amazon Simple Notification
# Service (SNS)
class SubscriptionService
  # Initializes the SubscriptionService with an SNS client
  #
  # @param sns_client [Aws::SNS::Client] The SNS client
  def initialize(sns_client)
    @sns_client = sns_client
    @logger = Logger.new($stdout)
  end

  # Attempts to create a subscription to a topic
  #
  # @param topic_arn [String] The ARN of the SNS topic
  # @param protocol [String] The subscription protocol (e.g., email)
  # @param endpoint [String] The endpoint that receives the notifications (email
  # address)
  # @return [Boolean] true if subscription was successfully created, false otherwise
  def create_subscription(topic_arn, protocol, endpoint)
    @sns_client.subscribe(topic_arn: topic_arn, protocol: protocol, endpoint:
    endpoint)
    @logger.info("Subscription created successfully.")
    true
  rescue Aws::SNS::Errors::ServiceError => e
    @logger.error("Error while creating the subscription: #{e.message}")
    false
  end
end
```

```
end
end

# Main execution if the script is run directly
if $PROGRAM_NAME == __FILE__
  protocol = "email"
  endpoint = "EMAIL_ADDRESS" # Should be replaced with a real email address
  topic_arn = "TOPIC_ARN"    # Should be replaced with a real topic ARN

  sns_client = Aws::SNS::Client.new
  subscription_service = SubscriptionService.new(sns_client)

  @logger.info("Creating the subscription.")
  unless subscription_service.create_subscription(topic_arn, protocol, endpoint)
    @logger.error("Subscription creation failed. Stopping program.")
    exit 1
  end
end
end
```

- Untuk informasi selengkapnya, lihat [AWS SDK for Ruby Panduan Developer](#).
- Untuk API detailnya, lihat [Berlangganan](#) di AWS SDK for Ruby API Referensi.

Contoh nirserver

Memanggil fungsi Lambda dari pemicu Amazon SNS

Contoh kode berikut menunjukkan cara menerapkan fungsi Lambda yang menerima peristiwa yang dipicu dengan menerima pesan dari suatu SNS topik. Fungsi mengambil pesan dari parameter peristiwa dan mencatat konten setiap pesan.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di repositori [contoh Nirserver](#).

Mengonsumsi SNS acara dengan Lambda menggunakan Ruby.

```
# Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
# SPDX-License-Identifier: Apache-2.0
def lambda_handler(event:, context:)
  event['Records'].map { |record| process_message(record) }
end

def process_message(record)
  message = record['Sns']['Message']
  puts("Processing message: #{message}")
rescue StandardError => e
  puts("Error processing message: #{e}")
  raise
end
```

SQSContoh Amazon menggunakan SDK untuk Ruby

Contoh kode berikut menunjukkan kepada Anda cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan AWS SDK for Ruby With AmazonSQS.

Tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)
- [Contoh nirserver](#)

Tindakan

ChangeMessageVisibility

Contoh kode berikut menunjukkan cara menggunakan `ChangeMessageVisibility`.

SDK untuk Ruby

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
require "aws-sdk-sqs" # v2: require 'aws-sdk'
# Replace us-west-2 with the AWS Region you're using for Amazon SQS.
sqs = Aws::SQS::Client.new(region: "us-west-2")

begin
  queue_name = "my-queue"
  queue_url = sqs.get_queue_url(queue_name: queue_name).queue_url

  receive_message_result_before = sqs.receive_message({
    queue_url: queue_url,
    max_number_of_messages: 10 # Receive up to 10 messages, if there are that many.
  })

  puts "Before attempting to change message visibility timeout: received
#{receive_message_result_before.messages.count} message(s)."

  receive_message_result_before.messages.each do |message|
    sqs.change_message_visibility({
      queue_url: queue_url,
      receipt_handle: message.receipt_handle,
      visibility_timeout: 30 # This message will not be visible for 30 seconds after
first receipt.
    })
  end

  # Try to retrieve the original messages after setting their visibility timeout.
  receive_message_result_after = sqs.receive_message({
    queue_url: queue_url,
    max_number_of_messages: 10
  })

  puts "\nAfter attempting to change message visibility timeout: received
#{receive_message_result_after.messages.count} message(s)."
```

```
rescue Aws::SQS::Errors::NonExistentQueue
  puts "Cannot receive messages for a queue named '#{receive_queue_name}', as it
  does not exist."
end
```

- Untuk API detailnya, lihat [ChangeMessageVisibility](#) di AWS SDK for Ruby API Referensi.

CreateQueue

Contoh kode berikut menunjukkan cara menggunakan `CreateQueue`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
# This code example demonstrates how to create a queue in Amazon Simple Queue
Service (Amazon SQS).

require "aws-sdk-sqs"

# @param sqs_client [Aws::SQS::Client] An initialized Amazon SQS client.
# @param queue_name [String] The name of the queue.
# @return [Boolean] true if the queue was created; otherwise, false.
# @example
#   exit 1 unless queue_created?(
#     Aws::SQS::Client.new(region: 'us-west-2'),
#     'my-queue'
#   )
def queue_created?(sqs_client, queue_name)
  sqs_client.create_queue(queue_name: queue_name)
  true
rescue StandardError => e
  puts "Error creating queue: #{e.message}"
  false
end
```

```
# Full example call:
# Replace us-west-2 with the AWS Region you're using for Amazon SQS.
def run_me
  region = "us-west-2"
  queue_name = "my-queue"
  sqs_client = Aws::SQS::Client.new(region: region)

  puts "Creating the queue named '#{queue_name}'..."

  if queue_created?(sqs_client, queue_name)
    puts "Queue created."
  else
    puts "Queue not created."
  end
end

# Example usage:
run_me if $PROGRAM_NAME == __FILE__
```

- Untuk API detailnya, lihat [CreateQueue](#) di AWS SDK for Ruby API Referensi.

DeleteQueue

Contoh kode berikut menunjukkan cara menggunakan `DeleteQueue`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
require "aws-sdk-sqs" # v2: require 'aws-sdk'
# Replace us-west-2 with the AWS Region you're using for Amazon SQS.
sqs = Aws::SQS::Client.new(region: "us-west-2")

sqs.delete_queue(queue_url: URL)
```

- Untuk API detailnya, lihat [DeleteQueue](#) di AWS SDK for Ruby API Referensi.

ListQueues

Contoh kode berikut menunjukkan cara menggunakan `ListQueues`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
require "aws-sdk-sqs"
require "aws-sdk-sts"

# @param sqs_client [Aws::SQS::Client] An initialized Amazon SQS client.
# @example
#   list_queue_urls(Aws::SQS::Client.new(region: 'us-west-2'))
def list_queue_urls(sqs_client)
  queues = sqs_client.list_queues

  queues.queue_urls.each do |url|
    puts url
  end
rescue StandardError => e
  puts "Error listing queue URLs: #{e.message}"
end

# Lists the attributes of a queue in Amazon Simple Queue Service (Amazon SQS).
#
# @param sqs_client [Aws::SQS::Client] An initialized Amazon SQS client.
# @param queue_url [String] The URL of the queue.
# @example
#   list_queue_attributes(
#     Aws::SQS::Client.new(region: 'us-west-2'),
#     'https://sqs.us-west-2.amazonaws.com/111111111111/my-queue'
#   )
def list_queue_attributes(sqs_client, queue_url)
  attributes = sqs_client.get_queue_attributes(
```



```
    queue_url: queue_url,
    attribute_names: ["All"]
  )

  attributes.attributes.each do |key, value|
    puts "#{key}: #{value}"
  end

rescue StandardError => e
  puts "Error getting queue attributes: #{e.message}"
end

# Full example call:
# Replace us-west-2 with the AWS Region you're using for Amazon SQS.
def run_me
  region = "us-west-2"
  queue_name = "my-queue"

  sqs_client = Aws::SQS::Client.new(region: region)

  puts "Listing available queue URLs..."
  list_queue_urls(sqs_client)

  sts_client = Aws::STS::Client.new(region: region)

  # For example:
  # 'https://sqs.us-west-2.amazonaws.com/111111111111/my-queue'
  queue_url = "https://sqs." + region + ".amazonaws.com/" +
    sts_client.get_caller_identity.account + "/" + queue_name

  puts "\nGetting information about queue '#{queue_name}'..."
  list_queue_attributes(sqs_client, queue_url)
end
```

- Untuk API detailnya, lihat [ListQueues](#) di AWS SDK for Ruby API Referensi.

ReceiveMessage

Contoh kode berikut menunjukkan cara menggunakan `ReceiveMessage`.

SDK untuk Ruby

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
require "aws-sdk-sqs"
require "aws-sdk-sts"

# Receives messages in a queue in Amazon Simple Queue Service (Amazon SQS).
#
# @param sqs_client [Aws::SQS::Client] An initialized Amazon SQS client.
# @param queue_url [String] The URL of the queue.
# @param max_number_of_messages [Integer] The maximum number of messages
#   to receive. This number must be 10 or less. The default is 10.
# @example
#   receive_messages(
#     Aws::SQS::Client.new(region: 'us-west-2'),
#     'https://sqs.us-west-2.amazonaws.com/111111111111/my-queue',
#     10
#   )
def receive_messages(sqs_client, queue_url, max_number_of_messages = 10)

  if max_number_of_messages > 10
    puts "Maximum number of messages to receive must be 10 or less. " \
      "Stopping program."
    return
  end

  response = sqs_client.receive_message(
    queue_url: queue_url,
    max_number_of_messages: max_number_of_messages
  )

  if response.messages.count.zero?
    puts "No messages to receive, or all messages have already " \
      "been previously received."
    return
  end
end
```

```
response.messages.each do |message|
  puts "-" * 20
  puts "Message body: #{message.body}"
  puts "Message ID:  #{message.message_id}"
end

rescue StandardError => e
  puts "Error receiving messages: #{e.message}"
end

# Full example call:
# Replace us-west-2 with the AWS Region you're using for Amazon SQS.
def run_me
  region = "us-west-2"
  queue_name = "my-queue"
  max_number_of_messages = 10

  sts_client = Aws::STS::Client.new(region: region)

  # For example:
  # 'https://sqs.us-west-2.amazonaws.com/111111111111/my-queue'
  queue_url = "https://sqs." + region + ".amazonaws.com/" +
    sts_client.get_caller_identity.account + "/" + queue_name

  sqs_client = Aws::SQS::Client.new(region: region)

  puts "Receiving messages from queue '#{queue_name}'..."

  receive_messages(sqs_client, queue_url, max_number_of_messages)
end


# Example usage:
run_me if $PROGRAM_NAME == __FILE__
```

- Untuk API detailnya, lihat [ReceiveMessage](#) di AWS SDK for Ruby API Referensi.

SendMessage

Contoh kode berikut menunjukkan cara menggunakan `SendMessage`.

SDK untuk Ruby

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
require "aws-sdk-sqs"
require "aws-sdk-sts"

# @param sqs_client [Aws::SQS::Client] An initialized Amazon SQS client.
# @param queue_url [String] The URL of the queue.
# @param message_body [String] The contents of the message to be sent.
# @return [Boolean] true if the message was sent; otherwise, false.
# @example
#   exit 1 unless message_sent?(
#     Aws::SQS::Client.new(region: 'us-west-2'),
#     'https://sqs.us-west-2.amazonaws.com/111111111111/my-queue',
#     'This is my message.'
#   )
def message_sent?(sqs_client, queue_url, message_body)
  sqs_client.send_message(
    queue_url: queue_url,
    message_body: message_body
  )
  true
rescue StandardError => e
  puts "Error sending message: #{e.message}"
  false
end

# Full example call:
# Replace us-west-2 with the AWS Region you're using for Amazon SQS.
def run_me
  region = "us-west-2"
  queue_name = "my-queue"
  message_body = "This is my message."

  sts_client = Aws::STS::Client.new(region: region)
```

```

# For example:
# 'https://sqs.us-west-2.amazonaws.com/111111111111/my-queue'
queue_url = "https://sqs." + region + ".amazonaws.com/" +
  sts_client.get_caller_identity.account + "/" + queue_name

sqs_client = Aws::SQS::Client.new(region: region)

puts "Sending a message to the queue named '#{queue_name}'..."

if message_sent?(sqs_client, queue_url, message_body)
  puts "Message sent."
else
  puts "Message not sent."
end
end

# Example usage:
run_me if $PROGRAM_NAME == __FILE__

```

- Untuk API detailnya, lihat [SendMessage](#) di AWS SDK for Ruby API Referensi.

SendMessageBatch

Contoh kode berikut menunjukkan cara menggunakan `SendMessageBatch`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```

require "aws-sdk-sqs"
require "aws-sdk-sts"

#
# @param sqs_client [Aws::SQS::Client] An initialized Amazon SQS client.
# @param queue_url [String] The URL of the queue.
# @param entries [Hash] The contents of the messages to be sent,

```

```
# in the correct format.
# @return [Boolean] true if the messages were sent; otherwise, false.
# @example
#   exit 1 unless messages_sent?(
#     Aws::SQS::Client.new(region: 'us-west-2'),
#     'https://sqs.us-west-2.amazonaws.com/111111111111/my-queue',
#     [
#       {
#         id: 'Message1',
#         message_body: 'This is the first message.'
#       },
#       {
#         id: 'Message2',
#         message_body: 'This is the second message.'
#       }
#     ]
#   )
def messages_sent?(sqs_client, queue_url, entries)
  sqs_client.send_message_batch(
    queue_url: queue_url,
    entries: entries
  )
  true
rescue StandardError => e
  puts "Error sending messages: #{e.message}"
  false
end

# Full example call:
# Replace us-west-2 with the AWS Region you're using for Amazon SQS.
def run_me
  region = "us-west-2"
  queue_name = "my-queue"
  entries = [
    {
      id: "Message1",
      message_body: "This is the first message."
    },
    {
      id: "Message2",
      message_body: "This is the second message."
    }
  ]
]
```

```
sts_client = Aws::STS::Client.new(region: region)

# For example:
# 'https://sqs.us-west-2.amazonaws.com/111111111111/my-queue'
queue_url = "https://sqs." + region + ".amazonaws.com/" +
  sts_client.get_caller_identity.account + "/" + queue_name

sqs_client = Aws::SQS::Client.new(region: region)

puts "Sending messages to the queue named '#{queue_name}'..."

if messages_sent?(sqs_client, queue_url, entries)
  puts "Messages sent."
else
  puts "Messages not sent."
end
end
```

- Untuk API detailnya, lihat [SendMessageBatch](#) di AWS SDK for Ruby API Referensi.

Contoh nirserver

Memanggil fungsi Lambda dari pemicu Amazon SQS

Contoh kode berikut menunjukkan cara menerapkan fungsi Lambda yang menerima peristiwa yang dipicu oleh menerima pesan dari antrian SQS. Fungsi mengambil pesan dari parameter peristiwa dan mencatat konten setiap pesan.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara mengatur dan menjalankannya di repositori [contoh Nirserver](#).

Mengonsumsi SQS acara dengan Lambda menggunakan Ruby.

```
# Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
```

```
# SPDX-License-Identifier: Apache-2.0
def lambda_handler(event:, context:)
  event['Records'].each do |message|
    process_message(message)
  end
  puts "done"
end

def process_message(message)
  begin
    puts "Processed message #{message['body']}"
    # TODO: Do interesting work based on the new message
  rescue StandardError => err
    puts "An error occurred"
    raise err
  end
end
```

Melaporkan kegagalan item batch untuk fungsi Lambda dengan pemicu Amazon SQS

Contoh kode berikut menunjukkan cara mengimplementasikan respons batch sebagian untuk fungsi Lambda yang menerima peristiwa dari antrian SQS. Fungsi melaporkan kegagalan item batch dalam respons, memberi sinyal ke Lambda untuk mencoba lagi pesan tersebut nanti.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan [contoh lengkapnya](#) dan pelajari cara mengatur dan menjalankannya di repositori [contoh Nirserver](#).

Melaporkan kegagalan item SQS batch dengan Lambda menggunakan Ruby.

```
# Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
# SPDX-License-Identifier: Apache-2.0
require 'json'

def lambda_handler(event:, context:)
  if event
    batch_item_failures = []
  end
end
```



```
sqs_batch_response = {}

event["Records"].each do |record|
  begin
    # process message
    rescue StandardError => e
      batch_item_failures << {"itemIdentifier" => record['messageId']}
    end
  end

  sqs_batch_response["batchItemFailures"] = batch_item_failures
  return sqs_batch_response
end
end
```

AWS STS contoh menggunakan SDK untuk Ruby

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan AWS SDK for Ruby with AWS STS.

Tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

AssumeRole

Contoh kode berikut menunjukkan cara menggunakan `AssumeRole`.

SDK untuk Ruby

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
# Creates an AWS Security Token Service (AWS STS) client with specified
credentials.
# This is separated into a factory function so that it can be mocked for unit
testing.
#
# @param key_id [String] The ID of the access key used by the STS client.
# @param key_secret [String] The secret part of the access key used by the STS
client.
def create_sts_client(key_id, key_secret)
  Aws::STS::Client.new(access_key_id: key_id, secret_access_key: key_secret)
end

# Gets temporary credentials that can be used to assume a role.
#
# @param role_arn [String] The ARN of the role that is assumed when these
credentials
#           are used.
# @param sts_client [Aws::STS::Client] An AWS STS client.
# @return [Aws::AssumeRoleCredentials] The credentials that can be used to assume
the role.
def assume_role(role_arn, sts_client)
  credentials = Aws::AssumeRoleCredentials.new(
    client: sts_client,
    role_arn: role_arn,
    role_session_name: "create-use-assume-role-scenario"
  )
  @logger.info("Assumed role '#{role_arn}', got temporary credentials.")
  credentials
end
```

- Untuk API detailnya, lihat [AssumeRole](#) di AWS SDK for Ruby API Referensi.

Contoh Amazon Texttract yang digunakan SDK untuk Ruby

Contoh kode berikut menunjukkan cara melakukan tindakan dan menerapkan skenario umum dengan menggunakan Amazon Texttract. AWS SDK for Ruby

Skenario adalah contoh kode yang menunjukkan kepada Anda bagaimana menyelesaikan tugas tertentu dengan memanggil beberapa fungsi dalam layanan atau dikombinasikan dengan yang lain AWS layanan.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Skenario](#)

Skenario

Buat aplikasi untuk menganalisis umpan balik pelanggan

Contoh kode berikut menunjukkan cara membuat aplikasi yang menganalisis kartu komentar pelanggan, menerjemahkannya dari bahasa aslinya, menentukan sentimen mereka, dan menghasilkan file audio dari teks yang diterjemahkan.

SDK untuk Ruby

Aplikasi contoh ini menganalisis dan menyimpan kartu umpan balik pelanggan. Secara khusus, ini memenuhi kebutuhan hotel fiktif di New York City. Hotel menerima umpan balik dari para tamu dalam berbagai bahasa dalam bentuk kartu komentar fisik. Umpan balik itu diunggah ke aplikasi melalui klien web. Setelah gambar kartu komentar diunggah, langkah-langkah berikut terjadi:

- Teks diekstraksi dari gambar menggunakan Amazon Texttract.
- Amazon Comprehend menentukan sentimen teks yang diekstraksi dan bahasanya.
- Teks yang diekstraksi diterjemahkan ke bahasa Inggris menggunakan Amazon Translate.
- Amazon Polly mensintesis file audio dari teks yang diekstraksi.

Aplikasi lengkap dapat digunakan dengan AWS CDK Untuk kode sumber dan petunjuk penerapan, lihat proyek di [GitHub](#).

Layanan yang digunakan dalam contoh ini

- Amazon Comprehend
- Lambda
- Amazon Polly
- Amazon Textract
- Amazon Translate

Contoh Amazon Translate menggunakan SDK Ruby

Contoh kode berikut menunjukkan cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan AWS SDK for Ruby with Amazon Translate.

Skenario adalah contoh kode yang menunjukkan kepada Anda bagaimana menyelesaikan tugas tertentu dengan memanggil beberapa fungsi dalam layanan atau dikombinasikan dengan yang lain AWS layanan.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Skenario](#)

Skenario

Buat aplikasi untuk menganalisis umpan balik pelanggan

Contoh kode berikut menunjukkan cara membuat aplikasi yang menganalisis kartu komentar pelanggan, menerjemahkannya dari bahasa aslinya, menentukan sentimen mereka, dan menghasilkan file audio dari teks yang diterjemahkan.

SDK untuk Ruby

Aplikasi contoh ini menganalisis dan menyimpan kartu umpan balik pelanggan. Secara khusus, ini memenuhi kebutuhan hotel fiktif di New York City. Hotel menerima umpan balik dari para tamu dalam berbagai bahasa dalam bentuk kartu komentar fisik. Umpan balik itu diunggah ke aplikasi melalui klien web. Setelah gambar kartu komentar diunggah, langkah-langkah berikut terjadi:

- Teks diekstraksi dari gambar menggunakan Amazon Textract.
- Amazon Comprehend menentukan sentimen teks yang diekstraksi dan bahasanya.
- Teks yang diekstraksi diterjemahkan ke bahasa Inggris menggunakan Amazon Translate.
- Amazon Polly mensintesis file audio dari teks yang diekstraksi.

Aplikasi lengkap dapat digunakan dengan AWS CDK Untuk kode sumber dan petunjuk penerapan, lihat proyek di [GitHub](#).

Layanan yang digunakan dalam contoh ini

- Amazon Comprehend
- Lambda
- Amazon Polly
- Amazon Textract
- Amazon Translate

WorkDocs Contoh Amazon menggunakan SDK untuk Ruby

Contoh kode berikut menunjukkan kepada Anda cara melakukan tindakan dan mengimplementasikan skenario umum dengan menggunakan AWS SDK for Ruby With Amazon WorkDocs.

Tindakan adalah kutipan kode dari program yang lebih besar dan harus dijalankan dalam konteks. Sementara tindakan menunjukkan cara memanggil fungsi layanan individual, Anda dapat melihat tindakan dalam konteks dalam skenario terkait.

Setiap contoh menyertakan tautan ke kode sumber lengkap, di mana Anda dapat menemukan instruksi tentang cara mengatur dan menjalankan kode dalam konteks.

Topik

- [Tindakan](#)

Tindakan

DescribeRootFolders

Contoh kode berikut menunjukkan cara menggunakan `DescribeRootFolders`.

SDK untuk Ruby

 Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturannya dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
# Retrieves the root folder for a user by email
# @param users [Array<Types::User>] A list of users selected from API response
# @param user_email [String] The email of the user.
def get_user_folder(users, user_email)
  user = users.find { |user| user.email_address == user_email }
  if user
    user.root_folder_id
  else
    @logger.error "Could not get root folder for user with email address
#{user_email}"
    exit(1)
  end
end

# Describes the contents of a folder
# @param [String] folder_id - The Id of the folder to describe.
def describe_folder_contents(folder_id)
  resp = @client.describe_folder_contents({
    folder_id: folder_id, # required
    sort: "NAME", # accepts DATE, NAME
    order: "ASCENDING", # accepts
ASCENDING, DESCENDING
  })

  resp.documents.each do |doc|
    md = doc.latest_version_metadata
    @logger.info "Name:          #{md.name}"
    @logger.info "Size (bytes):  #{md.size}"
    @logger.info "Last modified: #{doc.modified_timestamp}"
    @logger.info "Doc ID:       #{doc.id}"
    @logger.info "Version ID:   #{md.id}"
    @logger.info ""
  end
rescue Aws::WorkDocs::Errors::ServiceError => e
  @logger.error "Error listing folder contents: #{e.message}"
end
```

```
    exit(1)
  end
```

- Untuk API detailnya, lihat [DescribeRootFolders](#) di AWS SDK for Ruby API Referensi.

DescribeUsers

Contoh kode berikut menunjukkan cara menggunakan `DescribeUsers`.

SDK untuk Ruby

Note

Ada lebih banyak tentang GitHub. Temukan contoh lengkapnya dan pelajari cara pengaturan dan menjalankannya di [Repositori Contoh Kode AWS](#).

```
# Describes users within an organization
# @param [String] org_id: The ID of the org.
def describe_users(org_id)
  resp = @client.describe_users({
    organization_id: org_id,
    include: "ALL", # accepts ALL, ACTIVE_PENDING
    order: "ASCENDING", # accepts ASCENDING,
DESCENDING
    sort: "USER_NAME", # accepts USER_NAME,
FULL_NAME, STORAGE_LIMIT, USER_STATUS, STORAGE_USED
  })

  resp.users.each do |user|
    @logger.info "First name: #{user.given_name}"
    @logger.info "Last name:  #{user.surname}"
    @logger.info "Email:      #{user.email_address}"
    @logger.info "Root folder: #{user.root_folder_id}"
    @logger.info ""
  end
  resp.users
rescue Aws::WorkDocs::Errors::ServiceError => e
  @logger.error "AWS WorkDocs Service Error: #{e.message}"
  exit(1)
end
```

- Untuk API detailnya, lihat [DescribeUsers](#) di AWS SDK for Ruby API Referensi.

Keamanan AWS SDK untuk Ruby

Keamanan cloud di Amazon Web Services (AWS) merupakan prioritas tertinggi. Sebagai seorang pelanggan AWS, Anda mendapatkan manfaat dari pusat data dan arsitektur jaringan yang dibangun untuk memenuhi persyaratan dari organisasi yang paling sensitif terhadap keamanan. Keamanan adalah tanggung jawab bersama antara Anda AWS dan Anda. [Model Tanggung Jawab Bersama](#) menggambarkan ini sebagai Keamanan dari Cloud dan Keamanan dalam Cloud.

Security of the Cloud - AWS bertanggung jawab untuk melindungi infrastruktur yang menjalankan semua layanan yang ditawarkan di AWS Cloud dan memberi Anda layanan yang dapat Anda gunakan dengan aman. Tanggung jawab keamanan kami adalah prioritas tertinggi di AWS, dan efektivitas keamanan kami secara teratur diuji dan diverifikasi oleh auditor pihak ketiga sebagai bagian dari [Program AWS Kepatuhan](#).

Keamanan di Cloud — Tanggung jawab Anda ditentukan oleh yang AWS layanan Anda gunakan, dan faktor-faktor lain termasuk sensitivitas data Anda, persyaratan organisasi Anda, dan hukum dan peraturan yang berlaku.

Topik

- [Perlindungan Data AWS SDK untuk Ruby](#)
- [Identity and Access Management AWS SDK untuk Ruby](#)
- [Validasi Kepatuhan AWS SDK untuk Ruby](#)
- [Ketahanan untuk AWS SDK Ruby](#)
- [Keamanan Infrastruktur AWS SDK untuk Ruby](#)
- [Menegakkan TLS versi minimum di AWS SDK untuk Ruby](#)
- [Migrasi Klien Enkripsi Amazon S3](#)

Perlindungan Data AWS SDK untuk Ruby

[Model tanggung jawab AWS bersama model](#) berlaku untuk perlindungan data di. Seperti yang dijelaskan dalam model AWS ini, bertanggung jawab untuk melindungi infrastruktur global yang menjalankan semua AWS Cloud. Anda bertanggung jawab untuk mempertahankan kendali atas konten yang di-host pada infrastruktur ini. Anda juga bertanggung jawab atas tugas-tugas konfigurasi dan manajemen keamanan untuk AWS layanan yang Anda gunakan. Untuk informasi selengkapnya

tentang privasi data, lihat [Privasi Data FAQ](#). Untuk informasi tentang perlindungan data di Eropa, lihat [Model Tanggung Jawab AWS Bersama dan](#) posting GDPR blog di Blog AWS Keamanan.

Untuk tujuan perlindungan data, kami menyarankan Anda melindungi Akun AWS kredensial dan mengatur pengguna individu dengan AWS IAM Identity Center atau AWS Identity and Access Management (IAM). Dengan cara itu, setiap pengguna hanya diberi izin yang diperlukan untuk memenuhi tanggung jawab tugasnya. Kami juga menyarankan supaya Anda mengamankan data dengan cara-cara berikut:

- Gunakan otentikasi multi-faktor (MFA) dengan setiap akun.
- Gunakan SSL/TLS untuk berkomunikasi dengan AWS sumber daya. Kami membutuhkan TLS 1.2 dan merekomendasikan TLS 1.3.
- Siapkan API dan log aktivitas pengguna dengan AWS CloudTrail.
- Gunakan solusi AWS enkripsi, bersama dengan semua kontrol keamanan default di dalamnya AWS layanan.
- Gunakan layanan keamanan terkelola lanjut seperti Amazon Macie, yang membantu menemukan dan mengamankan data sensitif yang disimpan di Amazon S3.
- Jika Anda memerlukan FIPS 140-3 modul kriptografi yang divalidasi saat mengakses AWS melalui antarmuka baris perintah atau, gunakan titik akhir. API FIPS Untuk informasi selengkapnya tentang FIPS titik akhir yang tersedia, lihat [Federal Information Processing Standard \(FIPS\) 140-3](#).

Kami sangat merekomendasikan agar Anda tidak pernah memasukkan informasi identifikasi yang sensitif, seperti nomor rekening pelanggan Anda, ke dalam tanda atau bidang isian bebas seperti bidang Nama. Ini termasuk ketika Anda bekerja dengan atau lainnya AWS layanan menggunakan konsol, API, AWS CLI, atau AWS SDKs. Data apa pun yang Anda masukkan ke dalam tanda atau bidang isian bebas yang digunakan untuk nama dapat digunakan untuk log penagihan atau log diagnostik. Jika Anda memberikan URL ke server eksternal, kami sangat menyarankan agar Anda tidak menyertakan informasi kredensial dalam URL untuk memvalidasi permintaan Anda ke server tersebut.

Identity and Access Management AWS SDK untuk Ruby

AWS Identity and Access Management (IAM) adalah layanan Amazon Web Services (AWS) yang membantu administrator mengontrol akses ke AWS sumber daya dengan aman. IAM administrator mengontrol siapa yang dapat diautentikasi (masuk) dan diberi wewenang (memiliki izin) untuk

menggunakan sumber daya. AWS layanan IAM adalah AWS layanan yang dapat Anda gunakan tanpa biaya tambahan.

Untuk menggunakan Ruby AWS SDK untuk mengakses AWS, Anda memerlukan AWS akun dan AWS kredensial. Untuk meningkatkan keamanan AWS akun Anda, kami sarankan Anda menggunakan IAM pengguna untuk memberikan kredensi akses alih-alih menggunakan kredensi AWS akun Anda.

Untuk detail tentang bekerja dengan IAM, lihat [IAM](#).

Untuk ikhtisar IAM pengguna dan mengapa mereka penting untuk keamanan akun Anda, lihat [Kredensial AWS Keamanan](#) di Referensi [Umum Amazon Web Services](#).

AWS SDK untuk Ruby mengikuti [model tanggung jawab bersama](#) melalui layanan Amazon Web Services (AWS) tertentu yang didukungnya. Untuk informasi AWS layanan keamanan, lihat [halaman dokumentasi AWS layanan keamanan](#) dan [AWS layanan yang berada dalam lingkup upaya AWS kepatuhan oleh program kepatuhan](#).

Validasi Kepatuhan AWS SDK untuk Ruby

AWS SDK untuk Ruby mengikuti [model tanggung jawab bersama](#) melalui layanan Amazon Web Services (AWS) tertentu yang didukungnya. Untuk informasi AWS layanan keamanan, lihat [halaman dokumentasi AWS layanan keamanan](#) dan [AWS layanan yang berada dalam lingkup upaya AWS kepatuhan oleh program kepatuhan](#).

Keamanan dan kepatuhan layanan Amazon Web Services (AWS) dinilai oleh auditor pihak ketiga sebagai bagian dari beberapa program AWS kepatuhan. Ini termasuk SOC, PCI, Fed RAMP, HIPAA, dan lainnya. AWS menyediakan daftar yang sering diperbarui AWS layanan dalam lingkup program kepatuhan khusus di [AWS Layanan dalam Lingkup oleh Program Kepatuhan](#).

Laporan audit pihak ketiga tersedia untuk Anda unduh AWS Artifact. Untuk informasi selengkapnya, lihat [Mengunduh Laporan di AWS Artifak](#).

Untuk informasi selengkapnya tentang program AWS kepatuhan, lihat [Program AWS Kepatuhan](#).

Tanggung jawab kepatuhan Anda saat menggunakan Ruby AWS SDK untuk mengakses suatu AWS layanan ditentukan oleh sensitivitas data Anda, tujuan kepatuhan organisasi Anda, dan hukum dan peraturan yang berlaku. Jika penggunaan Anda tunduk pada kepatuhan terhadap standar seperti HIPAA, atau Fed RAMP, AWS menyediakan sumber daya untuk membantu: AWS layanan

- [Panduan Memulai Cepat Keamanan dan Kepatuhan — Panduan](#) penerapan yang membahas pertimbangan arsitektur dan memberikan langkah-langkah untuk menerapkan lingkungan dasar yang berfokus pada keamanan dan berfokus pada kepatuhan. AWS
- [Architecting for HIPAA Security and Compliance Whitepaper](#) — Whitepaper yang menjelaskan bagaimana perusahaan dapat menggunakan AWS untuk membuat aplikasi yang sesuai. HIPAA
- [AWS Sumber Daya Kepatuhan](#) — Kumpulan buku kerja dan panduan yang mungkin berlaku untuk industri dan lokasi Anda.
- [AWS Config](#) — Layanan yang menilai seberapa baik konfigurasi sumber daya Anda mematuhi praktik internal, pedoman industri, dan peraturan.
- [AWS Security Hub](#) — Pandangan komprehensif tentang status keamanan Anda di dalamnya AWS yang membantu Anda memeriksa kepatuhan Anda terhadap standar industri keamanan dan praktik terbaik.

Ketahanan untuk AWS SDK Ruby

Infrastruktur global Amazon Web Services (AWS) dibangun di sekitar Wilayah AWS dan Availability Zones.

Wilayah AWS menyediakan beberapa Availability Zone yang terpisah secara fisik dan terisolasi, yang terhubung dengan latensi rendah, throughput tinggi, dan jaringan yang sangat redundan.

Dengan Zona Ketersediaan, Anda dapat merancang serta mengoperasikan aplikasi dan basis data yang secara otomatis melakukan failover di antara Zona Ketersediaan tanpa gangguan. Zona Ketersediaan memiliki ketersediaan dan toleransi kesalahan yang lebih baik, dan dapat diskalakan dibandingkan infrastruktur biasa yang terdiri dari satu atau beberapa pusat data.

Untuk informasi selengkapnya tentang Wilayah AWS dan Availability Zone, lihat [Infrastruktur AWS Global](#).

AWS SDK untuk Ruby mengikuti [model tanggung jawab bersama](#) melalui layanan Amazon Web Services (AWS) tertentu yang didukungnya. Untuk informasi AWS layanan keamanan, lihat [halaman dokumentasi AWS layanan keamanan](#) dan [AWS layanan yang berada dalam lingkup upaya AWS kepatuhan oleh program kepatuhan](#).

Keamanan Infrastruktur AWS SDK untuk Ruby

AWS SDK untuk Ruby mengikuti [model tanggung jawab bersama](#) melalui layanan Amazon Web Services (AWS) tertentu yang didukungnya. Untuk informasi AWS layanan keamanan, lihat [halaman dokumentasi AWS layanan keamanan](#) dan [AWS layanan yang berada dalam lingkup upaya AWS kepatuhan oleh program kepatuhan](#).

Untuk informasi tentang proses AWS keamanan, lihat [AWS whitepaper: Ringkasan Proses Keamanan](#).

Menegakkan TLS versi minimum di AWS SDK untuk Ruby

Komunikasi antara Ruby dan AWS diamankan menggunakan Secure Sockets Layer (SSL) atau Transport Layer Security (TLS). AWS SDK Semua versi SSL, dan versi yang TLS lebih awal dari 1.2, memiliki kerentanan yang dapat membahayakan keamanan komunikasi Anda. AWS Untuk alasan ini, Anda harus memastikan bahwa Anda menggunakan AWS SDK untuk Ruby dengan versi Ruby yang mendukung TLS versi 1.2 atau yang lebih baru.

Ruby menggunakan SSL perpustakaan Terbuka untuk mengamankan HTTP koneksi. Versi Ruby yang didukung (1.9.3 dan yang lebih baru) diinstal melalui [manajer paket](#) sistem (yum,, dan lainnya) apt, [installer resmi](#), atau [manajer](#) Ruby (rbenv,, dan lainnya) biasanya menggabungkan Open SSL 1.0.1 atau yang lebih baru RVM, yang mendukung 1.2. TLS

Ketika digunakan dengan versi Ruby yang didukung dengan Open SSL 1.0.1 atau yang lebih baru, AWS SDK untuk Ruby lebih memilih TLS 1.2, dan menggunakan versi terbaru SSL atau TLS didukung oleh klien dan server. Ini selalu setidaknya TLS 1,2 untuk AWS layanan. (SDK menggunakan `Net::HTTP` kelas Ruby dengan `use_ssl=true`.)

Memeriksa SSL versi Terbuka

Untuk memastikan instalasi Ruby Anda menggunakan Open SSL 1.0.1 atau yang lebih baru, masukkan perintah berikut.

```
ruby -r openssl -e 'puts OpenSSL::OPENSSL_VERSION'
```

Cara alternatif untuk mendapatkan SSL versi Open adalah dengan menanyakan `openssl` executable secara langsung. Pertama, cari executable yang sesuai menggunakan perintah berikut.

```
ruby -r rbconfig -e 'puts RbConfig::CONFIG["configure_args"]'
```

Output harus `--with-openssl-dir=/path/to/openssl` menunjukkan lokasi SSL instalasi Terbuka. Catat jalan ini. Untuk memeriksa versi BukaSSL, masukkan perintah berikut.

```
cd /path/to/openssl
bin/openssl version
```

Metode terakhir ini mungkin tidak bekerja dengan semua instalasi Ruby.

Meningkatkan dukungan TLS

[Jika versi Open yang SSL digunakan oleh instalasi Ruby Anda lebih awal dari 1.0.1, tingkatkan SSL instalasi Ruby atau Open Anda menggunakan manajer paket sistem Anda, penginstal Ruby, atau manajer Ruby, seperti yang dijelaskan dalam panduan instalasi Ruby.](#) Jika Anda menginstal Ruby [dari sumber](#), instal [Open terbaru SSL](#) terlebih dahulu, lalu lulus `--with-openssl-dir=/path/to/upgraded/openssl` saat menjalankan `./configure`.

Migrasi Klien Enkripsi Amazon S3

Topik ini menunjukkan cara memigrasikan aplikasi Anda dari klien enkripsi Amazon Simple Storage Service (Amazon S3) ke Versi 1 (V1) Amazon Simple Storage Service (Amazon S3) ke Versi 2 (V2), dan memastikan ketersediaan aplikasi selama proses migrasi.

Ikhtisar Migrasi

Migrasi ini terjadi dalam dua fase:

1. Perbarui klien yang ada untuk membaca format baru. Pertama, gunakan versi terbaru dari Ruby AWS SDK untuk aplikasi Anda. Ini akan memungkinkan klien enkripsi V1 yang ada untuk mendekripsi objek yang ditulis oleh klien V2 baru. Jika aplikasi Anda menggunakan beberapa AWS SDKs, Anda harus memutakhirkan masing-masing SDK secara terpisah.
2. Migrasikan enkripsi dan dekripsi klien ke V2. Setelah semua klien enkripsi V1 Anda dapat membaca format baru, Anda dapat memigrasikan klien enkripsi dan dekripsi yang ada ke versi V2 masing-masing.

Perbarui Klien yang Ada untuk Membaca Format Baru

Klien enkripsi V2 menggunakan algoritma enkripsi yang tidak didukung oleh versi klien yang lebih lama. Langkah pertama dalam migrasi adalah memperbarui klien dekripsi V1 Anda ke rilis terbaru. SDK Setelah menyelesaikan langkah ini, klien V1 aplikasi Anda akan dapat mendekripsi objek yang

dienkripsi oleh klien enkripsi V2. Lihat detail di bawah untuk setiap versi utama AWS SDK untuk Ruby.

Update AWS SDK untuk Ruby Versi 3

Versi 3 adalah versi terbaru dari AWS SDK For Ruby. Untuk menyelesaikan migrasi ini, Anda perlu menggunakan permata versi 1.76.0 atau yang lebih baru. `aws-sdk-s3`

Menginstal dari baris perintah

Untuk proyek yang menginstal `aws-sdk-s3` permata, gunakan opsi versi untuk memverifikasi bahwa versi minimum 1.76.0 diinstal.

```
gem install aws-sdk-s3 -v '>= 1.76.0'
```

Menggunakan Gemfiles

Untuk proyek yang menggunakan Gemfile untuk mengelola dependensi, atur versi minimum `aws-sdk-s3` permata ke 1.76.0. Sebagai contoh:

```
gem 'aws-sdk-s3', '>= 1.76.0'
```

1. Ubah Gemfile Anda.
2. Jalankan `bundle update aws-sdk-s3`. Untuk memverifikasi versi Anda, jalankan `bundle info aws-sdk-s3`.

Update AWS SDK untuk Ruby Versi 2

Versi 2 AWS SDK untuk Ruby akan memasuki [mode pemeliharaan](#) pada 21 November 2021. Untuk menyelesaikan migrasi ini, Anda perlu menggunakan permata `aws-sdk` versi 2.11.562 atau yang lebih baru.

Menginstal dari baris perintah

Untuk proyek yang menginstal `aws-sdk` permata, dari baris perintah, gunakan opsi versi untuk memverifikasi bahwa versi minimum 2.11.562 diinstal.

```
gem install aws-sdk -v '>= 2.11.562'
```

Menggunakan Gemfiles

Untuk proyek yang menggunakan Gemfile untuk mengelola dependensi, atur versi minimum `aws-sdk` permata ke 2.11.562. Sebagai contoh:

```
gem 'aws-sdk', '>= 2.11.562'
```

1. Ubah Gemfile Anda. Jika Anda memiliki file `Gemfile.lock`, hapus atau perbarui.
2. Jalankan `bundle update aws-sdk`. Untuk memverifikasi versi Anda, jalankan `bundle info aws-sdk`.

Migrasi Klien Enkripsi dan Dekripsi ke V2

Setelah memperbarui klien Anda untuk membaca format enkripsi baru, Anda dapat memperbarui aplikasi Anda ke klien enkripsi dan dekripsi V2. Langkah-langkah berikut menunjukkan cara berhasil memigrasikan kode Anda dari V1 ke V2.

Sebelum memperbarui kode Anda untuk menggunakan klien enkripsi V2, pastikan Anda telah mengikuti langkah-langkah sebelumnya dan menggunakan `aws-sdk-s3` permata versi 2.11.562 atau yang lebih baru.

Note

Saat mendekripsi dengan AES-GCM, baca seluruh objek sampai akhir sebelum Anda mulai menggunakan data yang didekripsi. Ini untuk memverifikasi bahwa objek belum dimodifikasi sejak dienkripsi.

Mengkonfigurasi Klien Enkripsi V2

`EncryptionV2::Client` membutuhkan konfigurasi tambahan. Untuk informasi konfigurasi terperinci, lihat [dokumentasi EncryptionV2::Client](#) atau contoh yang diberikan nanti dalam topik ini.

1. Metode pembungkus kunci dan algoritma enkripsi konten harus ditentukan pada konstruksi klien. Saat membuat yang baru `EncryptionV2::Client`, Anda perlu memberikan nilai untuk `key_wrap_schema` dan `content_encryption_schema`.

`key_wrap_schema`- Jika Anda menggunakan AWS KMS, ini harus diatur ke `:kms_context`. Jika Anda menggunakan kunci simetris (AES), itu harus disetel ke `:aes_gcm`. Jika Anda menggunakan kunci asimetris (RSA), itu harus diatur ke `:rsa_oaep_sha1`.

`content_encryption_schema`- Ini harus diatur ke:`aes_gcm_no_padding`.

2. `security_profile` harus ditentukan pada konstruksi klien. Saat membuat yang baru `EncryptionV2::Client`, Anda perlu memberikan nilai untuk `security_profile`. Parameter `security_profile` menentukan dukungan untuk membaca objek yang ditulis menggunakan V1 yang lebih lama. `Encryption::Client` Ada dua nilai `:v2` dan `:v2_and_legacy`. Untuk mendukung migrasi, atur `security_profile` ke `:v2_and_legacy`. Gunakan: `v2` hanya untuk pengembangan aplikasi baru.

3. AWS KMS key ID diberlakukan secara default. Di `V1Encryption::Client`, yang `kms_key_id` digunakan untuk membuat klien tidak diberikan kepada `AWS KMS Decrypt` call `AWS KMS` bisa mendapatkan informasi ini dari metadata dan menambahkannya ke gumpalan ciphertext simetris. Dalam `V2, EncryptionV2::Client`, `kms_key_id` diteruskan ke panggilan `AWS KMS` Dekripsi, dan panggilan gagal jika tidak cocok dengan kunci yang digunakan untuk mengenkripsi objek. Jika kode Anda sebelumnya mengandalkan tidak menyetel yang spesifik `kms_key_id`, setel `kms_key_id: :kms_allow_decrypt_with_any_cmk` pada pembuatan klien atau setel `kms_allow_decrypt_with_any_cmk: true` `get_object` panggilan.

Contoh: Menggunakan Tombol Symmetric (AES)

Pra-migrasi

```
client = Aws::S3::Encryption::Client.new(encryption_key: aes_key)
client.put_object(bucket: bucket, key: key, body: secret_data)
resp = client.get_object(bucket: bucket, key: key)
```

Pasca-migrasi

```
client = Aws::S3::EncryptionV2::Client.new(
  encryption_key: rsa_key,
  key_wrap_schema: :rsa_oaep_sha1, # the key_wrap_schema must be rsa_oaep_sha1 for
  asymmetric keys
  content_encryption_schema: :aes_gcm_no_padding,
  security_profile: :v2_and_legacy # to allow reading/decrypting objects encrypted by
  the V1 encryption client
)
client.put_object(bucket: bucket, key: key, body: secret_data) # No changes
resp = client.get_object(bucket: bucket, key: key) # No changes
```

Contoh: Menggunakan AWS KMS dengan kms_key_id

Pra-migrasi

```
client = Aws::S3::Encryption::Client.new(kms_key_id: kms_key_id)
client.put_object(bucket: bucket, key: key, body: secret_data)
resp = client.get_object(bucket: bucket, key: key)
```

Pasca-migrasi

```
client = Aws::S3::EncryptionV2::Client.new(
  kms_key_id: kms_key_id,
  key_wrap_schema: :kms_context, # the key_wrap_schema must be kms_context for KMS keys
  content_encryption_schema: :aes_gcm_no_padding,
  security_profile: :v2_and_legacy # to allow reading/decrypting objects encrypted by
  the V1 encryption client
)
client.put_object(bucket: bucket, key: key, body: secret_data) # No changes
resp = client.get_object(bucket: bucket, key: key) # No change
```

Contoh: Menggunakan AWS KMS tanpa kms_key_id

Pra-migrasi

```
client = Aws::S3::Encryption::Client.new(kms_key_id: kms_key_id)
client.put_object(bucket: bucket, key: key, body: secret_data)
resp = client.get_object(bucket: bucket, key: key)
```

Pasca-migrasi

```
client = Aws::S3::EncryptionV2::Client.new(
  kms_key_id: kms_key_id,
  key_wrap_schema: :kms_context, # the key_wrap_schema must be kms_context for KMS keys
  content_encryption_schema: :aes_gcm_no_padding,
  security_profile: :v2_and_legacy # to allow reading/decrypting objects encrypted by
  the V1 encryption client
)
client.put_object(bucket: bucket, key: key, body: secret_data) # No changes
resp = client.get_object(bucket: bucket, key: key, kms_allow_decrypt_with_any_cmk:
  true) # To allow decrypting with any cmk
```

Alternatif Pasca Migrasi

Jika Anda hanya membaca dan mendekripsi (tidak pernah menulis dan mengenkripsi) objek menggunakan klien enkripsi S2, gunakan kode ini.

```
client = Aws::S3::EncryptionV2::Client.new(  
  kms_key_id: :kms_allow_decrypt_with_any_cmk, # set kms_key_id to allow all get_object  
  requests to use any cmk  
  key_wrap_schema: :kms_context, # the key_wrap_schema must be kms_context for KMS keys  
  content_encryption_schema: :aes_gcm_no_padding,  
  security_profile: :v2_and_legacy # to allow reading/decrypting objects encrypted by  
  the V1 encryption client  
)  
resp = client.get_object(bucket: bucket, key: key) # No change
```

Riwayat Dokumen

Tabel berikut menjelaskan perubahan penting dalam panduan ini. Untuk pemberitahuan tentang pembaruan dokumentasi ini, Anda dapat berlangganan [RSSumpam](#).

Perubahan	Deskripsi	Tanggal
Daftar isi dan contoh terpandu	Contoh terpandu yang dihapus untuk tunduk pada Repositori Contoh Kode yang lebih komprehensif.	Juli 10, 2024
Daftar isi	Daftar isi yang diperbarui untuk membuat contoh kode lebih mudah diakses.	1 Juni 2023
IAM pembaruan praktik terbaik	Panduan yang diperbarui untuk menyelaraskan dengan praktik IAM terbaik. Untuk informasi selengkapnya, lihat Praktik terbaik keamanan di IAM . Pembaruan untuk Memulai.	8 Mei 2023
Pembaruan umum	Memperbarui halaman selamat datang untuk sumber daya eksternal yang relevan. Juga memperbarui versi Ruby minimum yang diperlukan untuk v2.3. AWS Key Management Service Bagian yang diperbarui untuk mencerminkan pembaruan terminologi. Informasi penggunaan yang diperbarui tentang REPL utilitas untuk kejelasan.	8 Agustus 2022

[Memperbaiki tautan yang rusak](#)

Contoh tautan yang rusak tetap. Menghapus halaman Tips dan Trik redundan; mengarahkan ke konten contoh AmazonEC2. Termasuk daftar contoh kode yang tersedia di GitHub dalam repositori Contoh Kode.

3 Agustus 2022

[Metrik-metrik SDK](#)

Menghapus informasi tentang mengaktifkan SDK Metrik untuk Dukungan Perusahaan, yang telah tidak digunakan lagi.

28 Januari 2022

Terjemahan disediakan oleh mesin penerjemah. Jika konten terjemahan yang diberikan bertentangan dengan versi bahasa Inggris aslinya, utamakan versi bahasa Inggris.