aws

開発者ガイド

Amazon Polly

Amazon Polly: 開発者ガイド

Table of Contents

Amazon Polly とは

Amazon Polly はテキストを肉声に近い音声に変換するクラウドサービスです。Amazon Polly を使用して、エンゲージメントやアクセシビリティを高めるアプリケーションを開発できます。Amazon Polly は複数の言語をサポートし、リアルな音声を多数備えています。Amazon Polly を使用すると、複数の場所で動作し、顧客に最適な音声を使用する音声対応アプリケーションを構築できます。また、お支払いいただくのは合成したテキストのみです。また、追加コストなしで、Amazon Polly が生成した音声をキャッシュして再生できます。

Amazon Polly には、生成、ロングフォーム、ニューラル、標準 text-to-speech (TTS) オプションなど、多くの音声オプションが用意されています。これらの音声は、可能な限り最も自然で人間のような text-to-speech 音声を提供する新しい機械学習テクノロジーを使用して、音声品質の画期的改善を実現します。ニューラル TTS テクノロジーは、ニュースナレーションのユースケースに合わせたニュースキャスターの話し方もサポートしています。

Amazon Polly の一般的なユースケースには、ニュースリーダー、ゲーム、eLearningプラットフォームなどのモバイルアプリケーション、視覚障害者向けのアクセシビリティアプリケーション、モノのインターネット (IoT) の急成長セグメントなどがあります。

Amazon Polly は、HIPAA (Health Insurance Portability and Accountability Act of 1996) および Payment Card Industry Data Security Standard (PCI DSS) の規制されたワークロードでの使用が認定されています。

利点

Amazon Polly を使用する利点のいくつかを以下に示します。

- 高品質 — Amazon Polly は、高性能の生成音声、ロングフォーム音声、ニューラル音声、高品質音声 (TTS) text-to-speechを提供します。これらのテクノロジーは、自然な音声を高い発音精度で合成します (略語、頭字語の拡張、日付/時刻の解釈、ホモグラフのあいまいさの排除など)。
- 低レイテンシー – Amazon Polly は高速応答を実現するため、ダイアログシステムなどの低レイテンシーのユースケースで実行可能なオプションになります。
- 言語と音声の大規模なポートフォリオのサポート – Amazon Polly は多数の音声と言語をサポートしており、ほとんどの言語に男性と女性の音声オプションを提供します。さらに多くのニューラル音声が提供されるにつれて、この数は増え続ける予定です。米国英語の音声 Matthew と Joanna では、プロのニュースアンカーのようなニューラルニュースキャスターの話し方も使用できます。

- 費用対効果が高い — Amazon Polly の pay-per-use モデルでは、セットアップコストは発生しません。小規模から始めて、アプリケーションの拡大に合わせてスケールアップします。
- クラウドベースのソリューション – デバイス上の TTS ソリューションは、膨大なコンピューティングリソース、特に CPU パワー、RAM、ディスク容量を必要とします。これにより、タブレットやスマートフォンなどのデバイスでの開発コストが高くなり、電力消費量が高くなる可能性があります。対照的に、 で行われた TTS 変換は、ローカルリソースの要件 AWS クラウド を大幅に削減します。これにより、利用可能なすべての言語と音声を、優れた品質でサポートできます。さらに、音声の改善はすべてのエンドユーザーがすぐに利用でき、デバイスの追加更新は必要ありません。

> **Note**
>
> ブラウザで Amazon Polly 音声の例を聞くには、Amazon Polly 製品の概要」を参照してください。

を初めてお使いになる方向けの情報

Amazon Polly を初めて使用する場合は、以下のセクションをリストされた順序で読むことをお勧めします。

1. Amazon Polly の仕組み – このセクションでは、シンプルなエクスペリエンスを作成するために使用できるさまざまな Amazon Polly 入力とオプションを紹介します。
2. Amazon Polly の開始方法 – このセクションでは、アカウントをセットアップして Amazon Polly の音声合成をテストします。
3. サンプルアプリケーション – このセクションでは、Amazon Polly を試すときに役立つその他の例を示します。

Amazon Polly の仕組み

Amazon Polly は入力テキストを肉声に近い音声に変換します。Amazon Polly 音声を使用するには、音声エンジン を選択し、音声合成メソッドを呼び出し、合成するテキストを指定してから、音声出力形式を指定します。Amazon Polly により、入力されたテキストが高品質のスピーチ音声ストリームに合成されます。

- 入力テキスト – 合成するテキストを入力します。Amazon Polly によって音声ストリームが返されます。入力は、プレーンテキストまたは音声合成マークアップ言語 (SSML) 形式で指定できます。SSML を使用すると、発音、ボリューム、ピッチ、話す速度など、音声のさまざまな要素を制御できます。詳細については、「SSML ドキュメントからの音声の生成」を参照してください。
- 使用できる音声 – Amazon Polly には、バイリンガル音声 (英語とヒンディー語の両方) を含む言語と音声のポートフォリオが用意されています。ほとんどの言語で、男女両方の複数の音声から選択できます。音声合成タスクを起動するときに、音声 ID を指定すると、Amazon Polly がその音声を使用してテキストを音声に変換します。Amazon Polly は翻訳サービスではありません。合成音声はテキストと同じ言語になります。数字で表される数値 (例えば、53、53 ではない) は、テキストではなく音声の言語で合成されます。詳細については、Amazon Polly の音声を参照してください。
- 出力形式 – Amazon Polly は合成音声を複数の形式で提供できます。必要に応じて音声形式を選択できます。たとえば、ウェブやモバイルアプリケーション用に、MP3 や Ogg Vorbis 形式の音声をリクエストすることがあるかもしれません。または、 AWS IoT デバイスやテレフォニーソリューションで使用する PCM 出力形式をリクエストすることもできます。

> **Note**
>
> ブラウザで Amazon Polly 音声の例を聞くには、Amazon Polly 製品の概要」を参照してください。

を初めてお使いになる方向けの情報

Amazon Polly を初めて使用する場合は、次のトピックを順番に読むことをお勧めします。

- Amazon Polly の開始方法
- サンプルアプリケーション

- Amazon Polly のクォータ

Amazon Polly の開始方法

Amazon Polly には、既存のアプリケーションと簡単に統合できるいくつかの API オペレーションが用意されています。サポートされているオペレーションのリストについては、アクションを参照してください。次のオプションのいずれかを使用できます。

- AWS SDKs – SDKs、Amazon Polly へのリクエストは、指定した認証情報を使用して自動的に署名および認証されます。これは、アプリケーション構築に推奨される選択肢です。
- AWS CLI – を使用して AWS CLI 、コードを記述せずに Amazon Polly を使用できます。

以下のセクションで、Amazon Polly の使用を開始する方法を説明します。

トピック

Amazon Polly のセットアップ

Amazon Polly を初めて使用する場合は、事前に にサインアップする必要があります AWS。Amazon Web Services (AWS) にサインアップすると AWS、Amazon Polly を含む のすべてのサービスに AWS アカウントが自動的にサインアップされます。実際に使用した分のサービスとリソースについてのみ請求されます。初めてのお客様は AWS 、Amazon Polly を無料で使い始めることができます。詳細については、「AWS 無料利用枠」を参照してください。

AWS アカウントを既にお持ちの場合は、次のいずれかのアクティビティに進むことができます。

- コンソールでの Amazon Polly の使用
- での Amazon Polly の使用 AWS CLI

にサインアップする AWS アカウント

がない場合は AWS アカウント、次の手順を実行して作成します。

にサインアップするには AWS アカウント

1. https://portal.aws.amazon.com/billing/signup を開きます。
2. オンラインの手順に従います。

   サインアップ手順の一環として、通話呼び出しを受け取り、電話キーパッドで検証コードを入力するように求められます。

   にサインアップすると AWS アカウント、 AWS アカウントのルートユーザーが作成されます。ルートユーザーには、アカウントのすべての AWS のサービス とリソースへのアクセス権があります。セキュリティのベストプラクティスとして、ユーザーに管理アクセスを割り当て、ルートユーザーのみを使用してルートユーザーアクセスが必要なタスクを実行してください。

AWS サインアッププロセスが完了すると、 から確認メールが送信されます。https://aws.amazon.com/ の [マイアカウント] を選んで、いつでもアカウントの現在のアクティビティを表示し、アカウントを管理できます。

管理アクセスを持つユーザーを作成する

にサインアップしたら AWS アカウント、 を保護し AWS アカウントのルートユーザー、 を有効にして AWS IAM Identity Center、日常的なタスクにルートユーザーを使用しないように管理ユーザーを作成します。

のセキュリティ保護 AWS アカウントのルートユーザー

1. ルートユーザーを選択し、 AWS アカウント E メールアドレスを入力して、アカウント所有者AWS Management Consoleとして にサインインします。次のページでパスワードを入力します。

   ルートユーザーを使用してサインインする方法については、AWS サインイン ユーザーガイドの「ルートユーザーとしてサインインする」を参照してください。
2. ルートユーザーの多要素認証 (MFA) を有効にします。

   手順については、「IAM ユーザーガイド」の AWS アカウント 「ルートユーザーの仮想 MFA デバイスを有効にする (コンソール)」を参照してください。

管理アクセスを持つユーザーを作成する

1. IAM アイデンティティセンターを有効にします。

手順については、「AWS IAM Identity Center ユーザーガイド」の「AWS IAM Identity Centerの有効化」を参照してください。

2. IAM アイデンティティセンターで、ユーザーに管理アクセスを付与します。

   を ID ソース IAM アイデンティティセンターディレクトリ として使用する方法のチュートリアルについては、「 ユーザーガイド」の「デフォルト でユーザーアクセス IAM アイデンティティセンターディレクトリを設定するAWS IAM Identity Center 」を参照してください。

管理アクセス権を持つユーザーとしてサインインする

- IAM アイデンティティセンターのユーザーとしてサインインするには、IAM アイデンティティセンターのユーザーの作成時に E メールアドレスに送信されたサインイン URL を使用します。

  IAM Identity Center ユーザーを使用してサインインする方法については、「 AWS サインイン ユーザーガイド」の AWS 「 アクセスポータルにサインインする」を参照してください。

追加のユーザーにアクセス権を割り当てる

1. IAM アイデンティティセンターで、最小特権のアクセス許可を適用するというベストプラクティスに従ったアクセス許可セットを作成します。

   手順については、「AWS IAM Identity Center ユーザーガイド」の「権限設定を作成する」を参照してください。

2. グループにユーザーを割り当て、そのグループにシングルサインオンアクセス権を割り当てます。

   手順については、「AWS IAM Identity Center ユーザーガイド」の「グループの参加」を参照してください。

IAM の詳細については、以下を参照してください。

- AWS Identity and Access Management (IAM)
- IAM の使用開始
- IAM ユーザーガイド

> **Note**
>
> AWS アカウント ID を書き留めます。次のステップで必要になります。

コンソールでの Amazon Polly の使用

Amazon Polly コンソールから、Amazon Polly の音声合成機能のテストと使用をすばやく開始できます。Amazon Polly コンソールは、プレーンテキストまたは SSML 入力からの音声の合成をサポートしています。

トピック

ステップ 1.1: コンソールで音声クイックスタートを合成する

コンソールから、Amazon Polly 音声合成の音声品質をすばやくテストできます。

コンソールで Amazon Polly の音声を聞くには

1. にサインイン AWS Management Console し、https://console.aws.amazon.com/polly/ で Amazon Polly コンソールを開きます。
2. [Text-to-Speech (テキスト読み上げ機能)] タブを選択します。テキストフィールドにはサンプルテキストが読み込まれるので、Amazon Polly をすぐに試すことができます。
3. [SSML] を無効にします。
4. エンジン で、生成 、ロングフォーム 、ニューラル 、または標準 を選択します。
5. 言語と AWS リージョンを選択し、音声を選択します。(エンジン にニューラル を選択した場合、NTTS をサポートする言語と音声のみを使用できます。 標準音声とロングフォーム音声はすべて無効になっています)。
6. [聴く] を選択します。

詳細なテストについては、以下のトピックを参照してください。

- ステップ 1.2: コンソールで音声をプレーンテキスト入力と合成する

- コンソールでの SSML の使用
- コンソールでのレキシコンの適用 (音声合成)

ステップ 1.2: コンソールで音声をプレーンテキスト入力と合成する

次の手順では、プレーンテキスト入力を使用して音声を合成します。 (「W3C」と日付「10/3」 (10 月 3 日) がどのように合成されるかに注意してください。)

コンソールでプレーンテキスト入力を使用して音声を合成するには

1. Amazon Polly コンソールにログオンしたら、Amazon Polly」を選択します。次に、テキスト読み上げタブを選択します。
2. [SSML] を無効にします。
3. このテキストを入力ボックスに入力するか貼り付けます。

```
He was caught up in the game.
In the middle of the 10/3/2014 W3C meeting
he shouted, "Score!" quite loudly.
```

4. エンジン で、生成 、ロングフォーム 、ニューラル 、または標準 を選択します。
5. 言語と AWS リージョンを選択し、音声を選択します。 (エンジン にニューラル を選択した場合、NTTS をサポートする言語と音声のみを使用できます。 標準音声とロングフォーム音声はすべて無効になっています) 。
6. 音声をすぐに聞くには、[聴く] を選択します。
7. 音声をファイルに保存するには、以下のいずれかを行います。

   a. [ダウンロード] を選択します。

   b. 別のファイル形式に変更するには、[詳細設定] を展開して [音声ファイル形式の設定] を有効にし、目的のファイル形式を選択してから、[ダウンロード] を選択します。

詳細な例については、以下のトピックを参照してください。

- コンソールでのレキシコンの適用 (音声合成)
- コンソールでの SSML の使用

での Amazon Polly の使用 AWS CLI

Amazon Polly コンソールと では、ほぼすべての同じオペレーションを実行できます AWS CLI。ただし、 で合成された音声を聞くことはできません AWS CLI。でオーディオを操作するには AWS CLI、テキストを ファイルに保存します。次に、選択したオーディオアプリケーションで ファイルを開きます。

トピック

ステップ 2.1: をセットアップする AWS CLI

Amazon Polly で動作する AWS CLI ように をダウンロードして設定するには、次の手順に従います。

> **Important**
>
> この演習のステップを実行する AWS CLI ために は必要ありません。ただし、このガイドの一部の実習では AWS CLIを使用します。このステップをスキップして に移動しステップ 2.2: を使用した演習の開始方法 AWS CLI、後で必要な AWS CLI ときに を設定できます。

のセットアップ AWS CLI

を設定するには AWS Command Line Interface

1. AWS CLIをダウンロードして設定します。手順については、AWS Command Line Interface ユーザーガイドの次のトピックを参照してください。
   - のセットアップ AWS Command Line Interface
   - の設定 AWS Command Line Interface
2. Config ファイルに管理者ユーザー AWS CLI AWS の名前付きプロファイルを追加します。このプロファイルは、 AWS CLI コマンドの実行時に使用できます。名前付きプロファイルの詳細については、AWS Command Line Interface ユーザーガイドの「名前付きプロファイル」を参照してください。

```
[profile adminuser]
    aws_access_key_id = adminuser access key ID
    aws_secret_access_key = adminuser secret access key
    region = aws-region
```

利用可能な AWS リージョンと Amazon Polly でサポートされているリージョンのリストについては、「」の「リージョンとエンドポイント」を参照してくださいAmazon Web Services 全般のリファレンス。

> **Note**
>
> の設定時に指定した Amazon Polly でサポートされているリージョンを使用している場合は AWS CLI、 AWS CLI コード例から次の行を省略します。
>
> ```
> --region aws-region
> ```

3. コマンドプロンプトで以下のヘルプコマンドを入力して、セットアップを確認します。

```
aws help
```

有効な AWS コマンドのリストが AWS CLI ウィンドウに表示されます。

から Amazon Polly をアクティブ化する AWS CLI

以前に をダウンロードして設定したことがある場合は AWS CLI、 を再設定しない限り、Amazon Polly が使用できない場合があります AWS CLI。次の手順では、これが必要かどうかを確認します。

から Amazon Polly をアクティブ化するには AWS CLI

1. AWS CLI コマンドプロンプトで次のヘルプコマンドを入力して、Amazon Polly の可用性を確認します。

```
aws polly help
```

Amazon Polly の説明が表示され、有効なコマンドのリストが AWS CLI ウィンドウに表示される場合は、すぐに から AWS CLI Amazon Polly を使用できます。この手順の残りはスキップできます。これが表示されない場合は、ステップ 2 に進みます。

2. 次の 2 つのオプションのいずれかを使用して Amazon Polly をアクティブ化します。

   a. をアンインストールして再インストールします AWS CLI。

      手順については、AWS Command Line Interface ユーザーガイドの AWS Command Line Interfaceのインストールを参照してください。

      または

   b. ファイル service-2.json をダウンロードします。

      コマンドプロンプトで、次のコマンドを入力します。

```
aws configure add-model --service-model file://service-2.json --service-name
 polly
```

3. 再度 Amazon Polly が使用できるかどうかを確認します。

```
aws polly help
```

   Amazon Polly の説明が表示されます。

から音声エンジンをセットアップする AWS CLI

から AWS CLI、 engineパラメータはオプションで、generative、、long-form、および の 4 つの値を指定できますneuralstandard。例えば、次のコードを使用して米国West-2 (オレゴン) リージョンで start-speech-synthesis-task AWS CLI コマンドを実行するとします。

```
aws polly start-speech-synthesis-task \
  --engine neural
  --region us-west-2 \
  --endpoint-url "https://polly.us-west-1.amazonaws.com/" \
  --output-format mp3 \
  --output-s3-bucket-name your-bucket-name \
  --output-s3-key-prefix optional/prefix/path/file \
  --voice-id Joanna \
  --text file://text_file.txt
```

出力は次のようになります。

```
"SynthesisTask":
{
     "CreationTime": [..],
     "Engine": "neural",
     "OutputFormat": "mp3",
     "OutputUri": "https://s3.us-west-1.amazonaws.com/your-bucket-name/optional/prefix/
path/file.<task_id>.mp3",
     "TextType": "text",
     "RequestCharacters": [..],
     "TaskStatus": "scheduled",
     "TaskId": [task_id],
     "VoiceId": "Joanna"
 }
```

ステップ 2.2: を使用した演習の開始方法 AWS CLI

をすでにセットアップしている場合は AWS CLI、Amazon Polly が提供する音声合成をテストできます。この演習では、入力テキストを渡して SynthesizeSpeechオペレーションを呼び出します。生成された音声をファイルとして保存し、そのコンテンツを確認します。

1. synthesize-speech AWS CLI コマンドを実行して、サンプルテキストをオーディオファイル () に合成しますhello.mp3。

   次の AWS CLI 例は、Unix、Linux、macOS 用にフォーマットされています。Windows の場合は、各行末のバックスラッシュ (\) Unix 連結文字をカレット (^) に置き換え、入力テキストは二重引用符 (") で囲み、内部タグは一重引用符 (') で囲みます。

   ```
   aws polly synthesize-speech \
       --output-format mp3 \
       --voice-id Joanna \
       --text 'Hello, my name is Joanna. I learned about the W3C on 10/3 of last
    year.' \
       hello.mp3
   ```

   への呼び出しではsynthesize-speech、任意の音声で合成されるサンプルテキストを指定します。音声 ID (次のステップで説明) と出力形式を指定する必要があります。このコマンドは、生成された音声を hello.mp3 ファイルに保存します。MP3 ファイルに加えて、次の出力がコンソールに送信されます。

   ```
   {
   ```

```
        "ContentType": "audio/mpeg",
        "RequestCharacters": "71"
}
```

2. 生成された hello.mp3 ファイルを再生して、合成された音声を確認します。
3. DescribeVoices オペレーションを使用することにより、使用可能な音声のリストを取得します。次の describe-voices AWS CLI コマンドを実行します。

```
aws polly describe-voices
```

これに応じて、Amazon Polly は使用可能なすべての音声のリストを返します。音声ごとに、各種メタデータ (音声 ID、言語コード、言語名、音声の性別) がレスポンスに示されます。レスポンスの例を次に示します。

```
{
    "Voices": [
        {
            "Gender": "Female",
            "Name": "Salli",
            "LanguageName": "US English",
            "Id": "Salli",
            "LanguageCode": "en-US",
            "SupportedEngines": [
                "neural",
                "standard",
                "generative"
            ]
        },
        {
            "Gender": "Female",
            "Name": "Danielle",
            "LanguageName": "US English",
            "Id": "Danielle",
            "LanguageCode": "en-US",
            "SupportedEngines": [
                "long-form"
            ]
        }
    ]
}
```

オプションで、言語コードを指定して、特定の言語の利用可能な音声を見つけることができます。Amazon Polly では多数の音声がサポートされています。次の例では、ブラジルポルトガル語のすべての音声がリストされます。

```
aws polly describe-voices \
    --language-code pt-BR
```

言語コードのリストについては、「Amazon Polly の言語」を参照してください。これらの言語コードは W3C 言語識別タグ (*#### ISO 639 ### - ISO 3166 ####)* です。たとえば、en-US (アメリカ英語)、en-GB (イギリス英語)、es-ES (スペイン語) などです。さらに、AWS CLI で `help` オプションを使用して、言語コードのリストを取得することもできます。

```
aws polly describe-voices help
```

Python の例

このガイドでは、 を使用して Amazon Polly AWS SDK for Python (Boto) に API コールを実行するための Python コード例をいくつか紹介します。 Amazon Polly Python をセットアップし、次のセクションに用意されているサンプルコードをテストすることをお勧めします。その他の例については、「サンプルアプリケーション」を参照してください。

Python をセットアップしてサンプル (SDK) をテストする

Python サンプルコードをテストするには、 AWS SDK for Python (Boto)が必要です。手順については、AWS SDK for Python (Boto3) を参照してください。

Python コード例をテストするには

次の Python コード例は、次のアクションを実行します。

- を呼び出し AWS SDK for Python (Boto) てAmazon Pollyに`SynthesizeSpeech`リクエストを送信します (入力としてテキストを指定) 。
- レスポンスで生成された音声ストリームにアクセスし、音声をローカルディスク上のファイル (`speech.mp3`) に保存します。
- ローカルシステムのデフォルト音声プレイヤーを使用して音声ファイルを再生します。

コードをファイル (example.py) に保存して、実行します。

```
"""Getting Started Example for Python 2.7+/3.3+"""
from boto3 import Session
from botocore.exceptions import BotoCoreError, ClientError
from contextlib import closing
import os
import sys
import subprocess
from tempfile import gettempdir

# Create a client using the credentials and region defined in the [adminuser]
# section of the AWS credentials file (~/.aws/credentials).
session = Session(profile_name="adminuser")
polly = session.client("polly")

try:
    # Request speech synthesis
    response = polly.synthesize_speech(Text="Hello world!", OutputFormat="mp3",
                                       VoiceId="Joanna")
except (BotoCoreError, ClientError) as error:
    # The service returned an error, exit gracefully
    print(error)
    sys.exit(-1)

# Access the audio stream from the response
if "AudioStream" in response:
    # Note: Closing the stream is important because the service throttles on the
    # number of parallel connections. Here we are using contextlib.closing to
    # ensure the close method of the stream object will be called automatically
    # at the end of the with statement's scope.
        with closing(response["AudioStream"]) as stream:
           output = os.path.join(gettempdir(), "speech.mp3")

           try:
            # Open a file for writing the output as a binary stream
                with open(output, "wb") as file:
                   file.write(stream.read())
           except IOError as error:
              # Could not write to file, exit gracefully
              print(error)
              sys.exit(-1)

else:
```

```
    # The response didn't contain audio data, exit gracefully
    print("Could not stream audio")
    sys.exit(-1)

# Play the audio using the platform's default player
if sys.platform == "win32":
    os.startfile(output)
else:
    # The following works on macOS and Linux. (Darwin = mac, xdg-open = linux).
    opener = "open" if sys.platform == "darwin" else "xdg-open"
    subprocess.call([opener, output])
```

サンプルアプリケーションなどの追加の例については、「サンプルアプリケーション」を参照してください。

Amazon Polly の音声

Amazon Polly には、さまざまな言語で多数のリアルな音声とサポートが用意されています。各音声は各対象のネイティブスピーカーによって作成されるため、同じ言語であっても、音声によって異なります。を使用して AWS Management Console 、選択したテキストで各音声をテストすることもできます。ほとんどの言語で、少なくとも男性と女性の声が 1 種類ずつあります。多くの場合、種類はそれ以上あります。わずかですが、音声が 1 種類のみの言語もあります。

> **Note**
>
> ブラウザで Amazon Polly 音声の例を聞くには、Amazon Polly 製品の概要」を参照してください。

トピック

音声を再生する

Amazon Polly を設定したら、コンソールでカスタムテキストを使用して音声をテストできます。

コンソールで Amazon Polly の音声を聞くには

1. にサインイン AWS Management Console し、https://console.aws.amazon.com/polly/ で Amazon Polly コンソールを開きます。
2. [Text-to-Speech (テキスト読み上げ機能)] タブを選択します。
3. エンジン で、生成 、ロングフォーム 、ニューラル 、または標準 を選択します。
4. 言語とリージョンを選択します。次に、音声を選択します。
5. 再生する音声のテキストを入力するか、デフォルトのフレーズを使用して、[Listen] (聴く) を選択します。

> **Note**
>
> 音声の一覧表と、含まれる言語の数は、随時追加されています。新しい言語や音声を提案するには、このページからフィードバックを送信してください。申し訳ありませんが、特定の新しい言語のリリース予定については、そのリリース前にお問い合わせにお答えすることはできません。

使用可能な音声

Amazon Polly は、テキストから音声を合成するために、さまざまなリアルな音声を複数の言語で提供します。次の表は、Amazon Polly が提供するすべての音声を示しています。

	言語と言語バリアント	言語コード	名前/ID	性別	生成音声	ロングフォーム音声	ニューラル音声	標準音声
1	アラビア語	arb	Zeina	女性	いいえ	いいえ	いいえ	はい
2	アラビア語 (湾岸)	ar-AE	Hala* Zayd*	女性 男	いいえ いいえ	いいえ いいえ	はい はい	いいえ いいえ
3	オランダ語 (ベルギー)	nl-BE	Lisa	女性	いいえ	いいえ	はい	いいえ
4	カタロニア語	ca-ES	Arlet	女性	いいえ	いいえ	はい	いいえ
5	中国語 (広東語)	yue-CN	Hiujin	女性	いいえ	いいえ	はい	いいえ
6	標準中国語	cmn-CN	Zhiyu	女性	いいえ	いいえ	はい	はい

	言語と言語バリアント	言語コード	名前/ID	性別	生成音声	ロングフォーム音声	ニューラル音声	標準音声
7	デンマーク語	da-DK	Naja	女性	いいえ	いいえ	いいえ	はい
			Mads	男	いいえ	いいえ	いいえ	はい
			Sofie	女性	いいえ	いいえ	はい	いいえ
8	オランダ語	nl-NL	Laura	女性	いいえ	いいえ	はい	いいえ
			Lotte	女性	いいえ	いいえ	いいえ	はい
			Ruben	男	いいえ	いいえ	いいえ	はい
9	英語 (オーストラリア)	en-AU	Nicole	女性	いいえ	いいえ	いいえ	はい
			Olivia	女性	いいえ	いいえ	はい	いいえ
			Russell	男	いいえ	いいえ	いいえ	はい
10	英語 (英国)	en-GB	Amy**	女性	はい	いいえ	はい	はい
			Emma	女性	いいえ	いいえ	はい	はい
			Brian	男	いいえ	いいえ	はい	はい
			Arthur	男	いいえ	いいえ	はい	いいえ
11	英語 (インド)	en-IN	Aditi*	女性	いいえ	いいえ	いいえ	はい
			Raveena	女性	いいえ	いいえ	いいえ	はい
			Kajal*	女性	いいえ	いいえ	はい	いいえ
12	英語 (アイルランド)	en-IN	Niamh	女性	いいえ	いいえ	はい	いいえ

	言語と言語バリアント	言語コード	名前/ID	性別	生成音声	ロングフォーム音声	ニューラル音声	標準音声
13	英語 (ニュージーランド)	en-NZ	Aria	女性	いいえ	いいえ	はい	いいえ
14	英語 (南アフリカ)	en-ZA	Ayanda	女性	いいえ	いいえ	はい	いいえ

	言語と言語バリアント	言語コード	名前/ID	性別	生成音声	ロングフォーム音声	ニューラル音声	標準音声
15	英語 (米国)	en-US	Danielle	女性	いいえ	はい	はい	いいえ
			Gregory	男	いいえ	はい	はい	いいえ
			Ivy	女性(子)	いいえ	いいえ	はい	はい
			Joanna**		いいえ	いいえ	はい	はい
			Kendra	女性	いいえ	いいえ	はい	はい
			Kimberly	女性	いいえ	いいえ	はい	はい
			Salli	女性	いいえ	いいえ	はい	はい
			Joey	女性	いいえ	いいえ	はい	はい
			Justin	男	いいえ	いいえ	はい	いいえ
			Kevin	男性(子)	いいえ	いいえ	はい	はい
			Matthew**		はい	いいえ	はい	いいえ
			Ruth	女性	はい	はい	はい	いいえ
			Stephen	男	いいえ	いいえ	はい	いいえ
16	英語 (ウェールズ)	en-GB-WLS	Geraint	男	いいえ	いいえ	いいえ	はい
17	フィンランド語	fi-FI	Suvi	女性	いいえ	いいえ	はい	いいえ

	言語と言語バリアント	言語コード	名前/ID	性別	生成音声	ロングフォーム音声	ニューラル音声	標準音声
18	フランス語	fr-FR	Céline/ Celine Léa Mathieu Rémi	女性 女性 男 男	いいえ いいえ いいえ いいえ	いいえ いいえ いいえ いいえ	いいえ はい いいえ はい	はい はい はい いいえ
19	フランス語 (ベルギー)	fr-BE	Isabelle	女性	いいえ	いいえ	はい	いいえ
20	フランス語 (カナダ)	fr-CA	Chantal Gabrielle Liam	女性 女性 男	いいえ いいえ いいえ	いいえ いいえ いいえ	いいえ はい はい	はい いいえ いいえ
21	ドイツ語	de-DE	Marlene Vicki Hans Daniel	女性 女性 男 男	いいえ いいえ いいえ いいえ	いいえ いいえ いいえ いいえ	いいえ はい いいえ はい	はい はい はい いいえ
22	ドイツ語 (オーストリア)	de-AT	Hannah	女性	いいえ	いいえ	はい	いいえ
23	ヒンディー語	hi-IN	Aditi* Kajal*	女性 女性	いいえ いいえ	いいえ いいえ	いいえ はい	はい いいえ

	言語と言語バリアント	言語コード	名前/ID	性別	生成音声	ロングフォーム音声	ニューラル音声	標準音声
24	アイスランド語	is-IS	Dóra/Dora	女性	いいえ	いいえ	いいえ	はい
			Karl	男	いいえ	いいえ	いいえ	はい
25	イタリア語	it-IT	Carla	女性	いいえ	いいえ	いいえ	はい
			Bianca	女性	いいえ	いいえ	はい	はい
			Giorgio	男	いいえ	いいえ	いいえ	はい
			Adriano	男	いいえ	いいえ	はい	いいえ
26	日本語	ja-JP	Mizuki	女性	いいえ	いいえ	いいえ	はい
			Takumi	男	いいえ	いいえ	はい	はい
			Kazuha	女性	いいえ	いいえ	はい	いいえ
			Tomoko	女性	いいえ	いいえ	はい	いいえ
27	韓国語	ko-KR	Seoyeon	女性	いいえ	いいえ	はい	はい
28	ノルウェー語	nb-NO	Liv	女性	いいえ	いいえ	いいえ	はい
			Ida	女性	いいえ	いいえ	はい	いいえ
29	ポーランド語	pl-PL	Ewa	女性	いいえ	いいえ	いいえ	はい
			Maja	女性	いいえ	いいえ	いいえ	はい
			Jacek	男	いいえ	いいえ	いいえ	はい
			Jan	男	いいえ	いいえ	いいえ	はい
			Ola	女性	いいえ	いいえ	はい	いいえ

	言語と言語バリアント	言語コード	名前/ID	性別	生成音声	ロングフォーム音声	ニューラル音声	標準音声
30	ポルトガル語 (ブラジル)	pt-BR	Camila	女性	いいえ	いいえ	はい	はい
			Vitória/ Vitoria	女性	いいえ	いいえ	はい	はい
			Ricardo	男	いいえ	いいえ	いいえ	はい
			Thiago	男	いいえ	いいえ	はい	いいえ
31	ポルトガル語 (欧州)	pt-PT	Inês/ Ines	女性	いいえ	いいえ	はい	はい
			Cristiano	男	いいえ	いいえ	いいえ	はい
32	ルーマニア語	ro-RO	Carmen	女性	いいえ	いいえ	いいえ	はい
33	ロシア語	ru-RU	Tatyana	女性	いいえ	いいえ	いいえ	はい
			Maxim	男	いいえ	いいえ	いいえ	はい
34	スペイン語 (欧州)	es-ES	Conchita	女性	いいえ	いいえ	いいえ	はい
			Lucia	女性	いいえ	いいえ	はい	はい
			Enrique	男	いいえ	いいえ	いいえ	はい
			Sergio	男	いいえ	いいえ	はい	いいえ
35	スペイン語 (メキシコ)	es-MX	Mia	女性	いいえ	いいえ	はい	はい
			Andrés	男	いいえ	いいえ	はい	いいえ

	言語と言語バリアント	言語コード	名前/ID	性別	生成音声	ロングフォーム音声	ニューラル音声	標準音声
36	スペイン語 (米国)	es-US	Lupe**	女性	いいえ	いいえ	はい	はい
			Penélope/Penelope	女性	いいえ	いいえ	いいえ	はい
			Miguel	男	いいえ	いいえ	いいえ	はい
			Pedro	男	いいえ	いいえ	はい	いいえ
37	スウェーデン語	sv-SE	Astrid	女性	いいえ	いいえ	いいえ	はい
			Elin	女性	いいえ	いいえ	はい	いいえ
38	トルコ語	tr-TR	Filiz	女性	いいえ	いいえ	いいえ	はい
			ブルク	女性	いいえ	いいえ	はい	いいえ
39	ウェールズ語	cy-GB	Gwyneth	女性	いいえ	いいえ	いいえ	はい

* この音声はバイリンガルです。詳細については、「バイリンガル音声」を参照してください。

** これらの音声では、ニューラル形式で使用する場合、ニュースキャスターの話し方を使用できます。詳細については、「ニュースキャスター音声」を参照してください。

各 Amazon Polly 音声エンジンには固有の機能があります。Amazon Polly が提供する音声エンジンの機能およびリージョンの可用性について説明します。

- 生成音声
- ロングフォーム音声
- ニューラル音声
- 標準音声

ブランド音声

前の表にリストされている利用可能な音声に加えて、Amazon Polly を使用してブランドペルソナのカスタム音声を構築できます。ブランド音声を使用すると、一意で排他的な音声を顧客に提供できます。Amazon Polly ブランドの音声の詳細については、「Brand Voice」を参照してください。

音声の速度

音声は自然に異なるため、使用可能な各音声の速度は若干異なります。例えば、米国英語の音声では、Ivy と Joanna は Matthew よりも少し速く、Joey よりもかなり高速です。音声間には非常に多くのバリエーションがあるため、Amazon Polly 音声には標準速度 (1 分あたりの単語数) はありません。ただし、音声が音声マーク を使用して選択したテキストを発するまでにかかる時間を確認できます。

音声テキストパッセージの長さを時間指定するには

1. を開きます AWS CLI。
2. 次のコードを実行し、必要に応じて入力します。

```
aws polly synthesize-speech \
    --language-code optional language code if needed
    --output-format json \
    --voice-id [name of desired voice] \
    --text '[desired text]' \
    --speech-mark-types='["viseme"]' \
    LengthOfText.txt
```

3. `LengthOfText.txt` を開きます。

テキストが「Mary had a little lamb,」の場合、Amazon Polly から返される結果の最後の数行は次のようになります。

```
{"time":882,"type":"viseme","value":"t"}
{"time":964,"type":"viseme","value":"a"}
{"time":1082,"type":"viseme","value":"p"}
```

最後のビゼーム (基本的に「lamb」の最後の文字の音) は、読み上げ開始から 1,082 ミリ秒後に始まります。これは音声の長さと厳密には異なりますが、近いために、音声の比較の基礎として使用できます。

音声の速度を変更する

一部のアプリケーションでは、読み上げ速度の調整が必要な場合があります。音声速度の調整が必要な場合に備えて、Amazon Polly では SSML タグを使用して速度を変更する機能を用意しています。例えば、組織が移民の視聴者に本を読み込むアプリケーションを作成していた場合、音声速度を変更したい場合があります。対象者は英語を話せますが、流暢さは限られています。Amazon Polly は、SSML <prosody> タグを使用して音声速度を遅くするのに役立ちます。

パーセンテージを使用できます。

```
<speak>
     In some cases, it might help your audience to <prosody rate="85%">slow
     the speaking rate slightly to aid in comprehension.</prosody>
</speak>
```

または、プリセット速度：

```
<speak>
     In some cases, it might help your audience to <prosody rate="slow">slow
     the speaking rate slightly to aid in comprehension.</prosody>
</speak>
```

Amazon Polly で SSML を使用する場合、2 種類の速度から選択することができます。

- プリセット速度： `x-slow`、`slow`、`medium`、`fast`、および `x-fast`。上記の場合、各オプションの速度は、目的の音声に応じて、概算値を示します。`medium` オプションでは、通常の音声速度に設定されます。
- n% の音声レート： 20% から 200% までの任意の割合を使用できます。このような場合は、正確に速度を選択できます。ただし、実際の音声速度は、選択した音声によって異なります。100% は、通常の音声速度とみなされます。

> **Note**
>
> 選択した音声をさまざまな速度でテストします。各オプションの速度は概算であり、選択した音声によって異なります。

prosody タグの使用の詳細については、「」を参照してください音量、話す速度、ピッチを制御する 。

バイリンガル音声

Amazon Polly には、バイリンガル音声を生成する方法が 2 つあります。

- アクセント付きバイリンガル音声
- 完全なバイリンガル音声

アクセント付きバイリンガル音声

アクセント付きバイリンガル音声は、任意の Amazon Polly 音声を使用して作成できますが、SSML タグを使用する場合にのみ作成できます。

通常、入力テキストのすべての単語は、使用している音声のデフォルト言語で話されます。

例えば、Joanna (米国英語の話者) の音声を使用している場合、Amazon Polly は以下をフランス語のアクセントを使用せずに Joanna の音声で発声します。

```
<speak>
     Why didn't she just say, 'Je ne parle pas français?'
</speak>
```

この場合、Je ne parle pas français という単語は、英語のように発声されます。

ただし、<lang> タグを使用して Joanna の音声を使用すると、Amazon Polly は文章をアメリカのアクセントがあるフランス語として発声します。

```
<speak>
     Why didn't she just say, <lang xml:lang="fr-FR">'Je ne parle pas français?'</
lang>.
</speak>
```

Joanna はネイティブのフランス語音声ではないため、発音は、ネイティブである米国英語に基づきます。たとえば、français という単語の完ぺきなフランス語の発音では口蓋垂ふるえ音の /R/ ですが、Joanna の米国英語の発音では、この音素に対応する /r/ が使用されます。

イタリア語を話す Giorgio の音声を使用する場合、次のテキストでは、Amazon Polly は Giorgio の音声でイタリア語発音を使用して文を発声します。

```
<speak>
     Mi piace Bruce Springsteen.
</speak>
```

完全なバイリンガル音声

Aditi または Kajal (インド英語とヒンディー語) のような完全なバイリンガルの音声は、2 つの言語を流暢に話すことができます。これにより、同じ音声を使用して、1 つのテキスト内で両方の言語の単語や語句を使用することができます。

現在、利用できる完全なバイリンガル音声は、Aditi、Kajal、Hala、Zayd だけです。

バイリンガル音声 (例: Aditi) の使用

Aditi は、インド英語 (en-IN) とヒンディー語 (hi-IN) の両方を流暢に話します。英語とヒンディー語の音声を合成することができ、同じ文の途中でも音声を 2 つの言語間で切り替えられます。

ヒンディー語は、2 つの異なる形式で使用できます。

- デバナガリ: 「उसेन कहाँ, खेल तोह अब शुर होगा」
- ロマナガリ (ラテンアルファベットを使用): 「Usne kahan, khel toh ab shuru hoga」

さらに、英語とヒンディー語のどちらか、または両方の形式を 1 つの文の中で混在させることもできます。

- デバナガリ + 英語: 「This is the song कभी कभी अदिति」
- ロマナガリ + 英語: 「This is the song from the movie Jaane Tu Ya Jaane Na.」
- デバナガリ + ロマナガリ + 英語: 「This is the song कभी कभी अदिति from the movie Jaane Tu Ya Jaane Na.」

Aditi はバイリンガル音声であるため、Amazon Polly は言語と文字を区別することができ、テキストはこれらのすべてのケースで正しく読み上げられます。

Amazon Polly は、英語 (アラビア語数字) とヒンディー語 (デバナガリ数字) の両方で、数字、日付、時刻、通貨の拡張もサポートしています。デフォルトでは、アラビア数字はインド英語で読み上げられます。Amazon Polly がヒンディー語で読み上げるようにするには、`hi-IN` 言語コードパラメータを使用する必要があります。

ニュースキャスター音声

人は、状況に応じて異なる話し方を使用します。たとえば、くだけた会話は、テレビやラジオのニュース放送の話し方とは大きく異なります。標準音声では、その設計上、異なる話し方を作成することはできません。しかし、ニューラル音声では可能です。特定の話し方に合わせてトレーニングし、その話し方に固有の特定部分にバリエーションを設け、重点を置きます。

Amazon Polly では、デフォルトのニューラル音声に加えて、ニュースキャスターの話し方が用意されています。これは、ニューラルシステムを使用して、テレビやラジオのニュースキャスターの話し方を生成します。ニュースキャスタースタイルは、米国英語 (en-US) の Matthew と Joanna の音声、スペイン語 (米国) (es-US) の Lupe の音声、およびイギリス英語 (en-GB) の Amy の音声で利用できます。

ニュースキャスターの話し方を使用するには、まずニューラルエンジンを選択してから、入力テキストで以下の手順に説明している構文を使用します。

> **Note**
>
> - ニューラルの話し方を使用するには、ニューラル音声に対応している AWS リージョンのいずれかを使用する必要があります。このオプションは一部のリージョンでは利用できません。詳細については、「機能とリージョンの互換性」を参照してください。

ニュースキャスタースタイルを適用するには (コンソール)

1. Amazon Polly コンソール (https://console.aws.amazon.com/polly/) を開きます。
2. ニューラル音声がサポートされている AWS リージョンを使用していることを確認してください。
3. [Text-to-Speech (テキスト読み上げ機能)] ページの [エンジン] で、[ニューラル] を選択します。
4. 使用する言語と音声を選択します。ニュースキャスターの音声は、米国英語 (en-US) の Matthew と Joanna、スペイン語 (米国) (es-US) の Lupe、およびイギリス英語 (en-GB) の Amy でのみ利用できます。
5. [SSML] を有効にします。
6. Newscaster スタイルの SSML 構文を使用して、 text-to-speech リクエストに入力テキストを追加します。

```
<amazon:domain name="news">text</amazon:domain>
```

たとえば、newscaster タグを以下のように使用できます。

```
<speak>
<amazon:domain name="news">
From the Tuesday, April 16th, 1912 edition of The Guardian newspaper:

The maiden voyage of the White Star liner Titanic, the largest ship ever launched
ended in disaster.

The Titanic started her trip from Southampton for New York on Wednesday. Late on
Sunday night she struck an iceberg off the Grand Banks of Newfoundland. By
wireless telegraphy she sent out signals of distress, and several liners were
near enough to catch and respond to the call.
</amazon:domain>
</speak>
```

7. [Listen] (聴く) を選択します。

ニュースキャスタースタイルを適用するには (CLI)

1. API リクエストで、エンジンパラメータに `neural` 値を含めます。

```
  --engine neural
```

2. ニュースキャスターの話し方の SSML 構文を使用して、API リクエストに入力テキストを追加します。

```
<amazon:domain name="news">text</amazon:domain>
```

たとえば、newscaster タグを以下のように使用できます。

```
<speak>
<amazon:domain name="news">
From the Tuesday, April 16th, 1912 edition of The Guardian newspaper:

The maiden voyage of the White Star liner Titanic, the largest ship ever launched
ended in disaster.

The Titanic started her trip from Southampton for New York on Wednesday. Late on
Sunday night she struck an iceberg off the Grand Banks of Newfoundland. By
```

```
wireless telegraphy she sent out signals of distress, and several liners were
near enough to catch and respond to the call.
</amazon:domain>
</speak>
```

SSML の詳細については、「サポートされている SSML タグ」を参照してください。

Amazon Polly の言語

次の言語は Amazon Polly でサポートされており、音声の合成に対応しています。各言語には固有の言語コードがあります。これらの言語コードは W3C 言語識別タグ (言語名には *ISO 639-3*、国コードには *ISO 3166*) です。

Amazon Polly が提供する音素とビゼームの詳細については、次の表から言語を選択してください。

	[言語]	言語コード
1	アラビア語	arb
2	アラビア語 (湾岸)	ar-AE
3	カタロニア語	ca-ES
4	中国語 (広東語)	yue-CN
5	標準中国語	cmn-CN
6	デンマーク語	da-DK
7	オランダ語 (ベルギー)	nl-BE
8	オランダ語	nl-NL
9	英語 (オーストラリア)	en-AU
10	英語 (英国)	en-GB
11	英語 (インド)	en-IN
12	英語 (ニュージーランド)	en-NZ
13	英語 (南アフリカ)	en-ZA
14	英語 (米国)	en-US
15	英語 (ウェールズ)	en-GB-WLS

	[言語]	言語コード
16	フィンランド語	fi-FI
17	フランス語	fr-FR
18	フランス語 (ベルギー)	fr-BE
19	フランス語 (カナダ)	fr-CA
20	ヒンディー語	hi-IN
21	ドイツ語	de-DE
22	ドイツ語 (オーストリア)	de-AT
23	アイスランド語	is-IS
24	イタリア語	it-IT
25	日本語	ja-JP
26	韓国語	ko-KR
27	ノルウェー語	nb-NO
28	ポーランド語	pl-PL
29	ポルトガル語 (ブラジル)	pt-BR
30	ポルトガル語 (欧州)	pt-PT
31	ルーマニア語	ro-RO
32	ロシア語	ru-RU
33	スペイン語 (欧州)	es-ES
34	スペイン語 (メキシコ)	es-MX
35	スペイン語 (米国)	es-US

	[言語]	言語コード
36	スウェーデン語	sv-SE
37	トルコ語	tr-TR
38	ウェールズ語	cy-GB

詳細については、「サポートされている言語の音素およびビゼームテーブル」を参照してください。

サポートされている言語の音素およびビゼームテーブル

以下の表に、Amazon Polly でサポートされている言語の音素、およびその例と対応するビゼームを示します。

トピック

- アラビア語 (ar)
- アラビア語 (湾岸) (ar-AE)
- カタロニア語 (ca-ES)
- 中国語 (広東語) (yue-CN)
- 標準中国語 (cmn-CN)
- デンマーク語 (da-DK)
- オランダ語 (ベルギー) (nl-BE)
- オランダ語 (nl-NL)
- 英語 (米国) (en-US)
- 英語 (オーストラリア) (en-AU)
- 英語 (英国) (en-GB)
- 英語 (インド) (en-IN)
- 英語 (アイルランド) (en-IE)
- 英語 (ニュージーランド) (en-NZ)
- 英語 (南アフリカ) (en-ZA)

- 英語 (ウェールズ) (en-GB-WLS)
- フィンランド語 (fi-FI)
- フランス語 (fr-FR)
- フランス語 (ベルギー) (fr-BE)
- フランス語 (カナダ) (fr-CA)
- ドイツ語 (de-DE)
- ドイツ語 (オーストリア) (de-AT)
- ヒンディー語 (hi-IN)
- アイスランド語 (is-IS)
- イタリア語 (it-IT)
- 日本語 (ja-JP)
- 韓国語 (ko-KR)
- ノルウェー語 (nb-NO)
- ポーランド語 (pl-PL)
- ポルトガル語 (pt-PT)
- ポルトガル語 (ブラジル) (pt-BR)
- ルーマニア語 (ro-RO)
- ロシア語 (ru-RU)
- スペイン語 (es-ES)
- スペイン語 (メキシコ) (es-MX)
- スペイン語 (米国) (es-US)
- スウェーデン語 (sv-SE)
- トルコ語 (tr-TR)
- ウェールズ語 (cy-GB)

アラビア語 (ar)

以下の表では、国際音声記号 (IPA) の音素および拡張 SAM 音声記号 (X-SAMPA) の記号の完全なリスト、および Amazon Polly でサポートされるアラビア語の音声 (Zeina) の対応するビゼームについて説明しています。

音素/ビゼームテーブル

IPA	X-SAMPA	説明	例	ビゼーム
子音				
ʔ	?	声門破裂音	اأنا	
ʕ	?\	有声咽頭摩擦音	عُمَر	k
b	b	有声両唇破裂音	دَلَب	p
d	d	有声歯茎破裂音	داري	t
dˤ	d_?\	emphatic 有声歯茎破裂音	ءوَض	t
ʤ	dZ	有声後部歯茎破擦音	ليمَج	S
ð	D	有声歯摩擦音	كَلذ	T
ðˤ	D_?\	emphatic 有声歯摩擦音	مالَظ	T
f	f	無声唇歯摩擦音	لصَف	f
g	Amazon EC2 のセキュリティグループ	有声軟口蓋破裂音	إرتلجنإ	k
ɣ	G	有声軟口蓋摩擦音	برَغ	k
h	h	無声声門摩擦音	اذه	k
j	j	硬口蓋接近音	يَمشي	インスタンス
k	k	無声軟口蓋破裂音	بلَك	k
l	l	歯茎側面接近音	ىقال	t

IPA	X-SAMPA	説明	例	ビゼーム
lˠ	l_G	有声歯茎側面接近音	هلدابع	t
m	m	両唇鼻音	امام	p
n	n	歯茎鼻音	ヌール	t
p	p	無声両唇破裂音	سبَح	p
q	q	無声口蓋垂破裂音	بيرَق	k
r	r	歯茎ふるえ音	لمَر	r
s	s	無声歯茎摩擦音	سُؤال	s
sˤ	s_?\	有声無声歯茎摩擦音	بحاص	s
ʃ	S	無声後部歯茎摩擦音	رشُك	S
t	t	無声歯茎破裂音	رمَت	t
tˤ	t_?\	有声無声歯茎破裂音	بالط	t
θ	T	無声歯摩擦音	ثاَلَث	T
v	v	有声唇歯摩擦音	نيماتيف	f
w	w	有声両唇軟口蓋接近音	وَلَد	u
x	x	無声軟口蓋摩擦音	خَوْف	k
ħ	X\	無声咽頭摩擦音	حَوْلَ	k
z	z	有声歯茎摩擦音	زُور	s

IPA	X-SAMPA	説明	例	ビゼーム
母音				
a	a	非円唇前舌広母音	بَرد	a
a:	a:	非円唇前舌広長母音	دار	a
ɑˤ	A_?\	非円唇後舌広有声母音	طَبل	a
ɑˤː	A_?\:	非円唇後舌広有声母音	ظالِم	a
u	u	円唇後舌狭母音	شُرب	u
u:	u:	円唇後舌狭長母音	سور	u
uˤ	u_?\	円唇後舌狭有声母音	بُدّ	u
uˤː	u_?\:	円唇後舌狭有声母音	طول	u
インスタンス	i	非円唇前舌狭母音	بِنت	インスタンス
iː	インスタンス:	非円唇前舌狭長母音	نَزَح	インスタンス
iˤ	i_?\	非円唇前舌狭有声母音	ضِدّ	i
iˤː	i_?\:	非円唇前舌狭有声長母音	مضايق	i
e	e	非円唇前舌半狭母音	مكارم	e

IPA	X-SAMPA	説明	例	ビゼーム
e:	e:	非円唇前舌半狭長母音	ليدوم	e
ɔ	O	円唇後舌半広母音	يجولونكت	O
ɔ:	O:	円唇後舌半広長母音	نويزفيلت	O

アラビア語 (湾岸) (ar-AE)

以下の表では、国際音声記号 (IPA) の音素および拡張 SAM 音声記号 (X-SAMPA) の記号の完全なリスト、および Amazon Polly でサポートされるアラビア語の音声 (Hala) の対応するビゼームについて説明しています。

音素/ビゼームテーブル

IPA	X-SAMPA	説明	例	発音	ビゼーム
子音					
b	b	有声両唇破裂音	دلب	/ " b a . l a d /	b
d	d	有声歯茎破裂音	درد	/ " r a d d /	d
dˤ	d_?\	有声咽頭歯茎破裂音	ضوء	/ " d_?\ a w ? /	D
f	f	無声唇歯摩擦音	نرف	/ " f l . r l n /	f
Amazon EC2 のセキュリ	Amazon EC2 のセキュリティグループ	有声軟口蓋破裂音	لاق	/ " g a: l /	k

IPA	X-SAMPA	説明	例	発音	ビゼーム
ティグループ					
j	j	有声硬口蓋接近音	يشمي	/ " j I m . S i: /	i
k	k	無声軟口蓋破裂音	لمام	/ " k a: . m i l /	k
l	l	有声歯茎側面接近音	ليل	/ " l e: l /	t
lˤ	l_G	有声咽頭歯茎側面接近音	هللادبع	/ ?\ a b . " d A_?\ l_G . l_G A_?\ /	t
m	m	両唇鼻腔閉鎖音	ةئم	/ " m I j . j a /	p
n	n	歯茎鼻腔閉鎖音	ヌール	/ " n u: r /	t
p	p	無声両唇破裂音	أوبرا	/ " ? O . p e . r a: /	p
q	q	無声口蓋垂破裂音	رصق	/ " q A_?\ s_?\ r /	k
r	r	歯茎ふるえ音	لمر	/ " r a . m I l /	r
s	s	無声歯茎摩擦音	مسمس	/ " s I m . s I m /	s
sˤ	s_?\	無声咽頭歯茎摩擦音	بحاص	/ " s_?\ A_?: . X \ I b /	s

IPA	X-SAMPA	説明	例	発音	ビゼーム
t	t	無声歯茎破裂音	رمت	/ "t a . m a r /	t
tˤ	t_?\	無声咽頭歯茎摩擦音	بالط	/ " t_?\ A_?: . l l b /	t
v	v	有声唇歯摩擦音	نيمات ي ف	/ v i: . t A . " m i: n /	f
w	w	有声両唇軟口蓋接近音	دواي	/ " w a: . j l d /	u
x	x	無声軟口蓋摩擦音	فورخ	/ x a . " r u: f /	k
z	z	無声軟口蓋摩擦音	رورز	/ " z h u: r /	s
ð	D	有声歯間摩擦音	كلذ	/ " D a: . l l k /	D
ðˤ	D_?\	有声咽頭歯間摩擦音	مالظ	/ D_?\ A_?\ . " l a: m /	D
ħ	X\	無声咽頭摩擦音	نيجلا	/ ? a l . " X\ i: n /	k
ŋ	N	軟口蓋腔閉鎖音	غنوك غنوه	/ h O N . " k O N g /	k
ɣ	G	有声軟口蓋摩擦音	ةبيرغ	/ G l . " r i: . b a /	k
ʃ	S	無声後部歯茎摩擦音	سمش	/ " S a m s /	S
ʒ	Z	有声後部歯茎摩擦音	تيكاج	/ Z a . " k e: t /	S

IPA	X-SAMPA	説明	例	発音	ビゼーム
ʔ	?	声門破裂音	مؤسسة	/ m u . " ? a s . s a . s a /	
ʕ	?\	有声咽頭摩擦音	عام	/ " ?\ a: m m /	k
dʒ	dZ	有声後部歯茎破擦音	جامعة	/ " dZ a: m . ?\ a /	S
θ	T	有声歯間摩擦音	ثالثة	/ T a . " l a: . T a /	T
ɦ	h	有声声門摩擦音	هلال	/ " h l a: l /	k
母音					
æ	a	非円唇前舌半広短母音	فرس	/ " s a . f a r /	a
ɑʕ	A_?\	非円唇咽頭後舌広短母音	صلب	/ " s_?\ A_?\ l b /	a
æː	a:	非円唇前舌半広長母音	باب	/ " b a: b /	a
ɑʕː	A_?\:	非円唇咽頭後舌広長母音	نضاج	/ " n A_?: . D_? \ i_?\ dZ /	a
a	A	非円唇中舌広短母音	wifi	/ " w A j . f A j /	a
i	i	非円唇前舌狭緊張短母音 (MSA)	إسحاق	/ ? i s . " X\ A_? \: q /	i
ɪ	I	非円唇前舌狭弛緩母音	بنت	/ " b I n t /	インスタンス

IPA	X-SAMPA	説明	例	発音	ビゼーム
iˤ	i_?\	非円唇咽頭前舌狭短母音	لفط	/ " t_?\ i_?\ f l l /	i
iː	i:	非円唇前舌狭長母音	ليبس	/ s a . " b i: l /	i
iˤː	i_?:	非円唇咽頭前舌狭長母音	بيطر	/ r A_?\ . " t_?\ i_?: b /	i
u	u	円唇後舌狭緊張短母音 (MSA)	عرتخم	/ " m u x . t a . r i ?\ /	u
ʊ	U	円唇後舌狭弛緩短母音	موسر	/ r U . " s u: m /	u
uˤ	u_?\	円唇咽頭後舌狭短母音	روفصع	/ ?\ u_?\ s_?\ . " f u: r /	u
uː	u:	円唇後舌狭長母音	توت	/ " t u: t /	u
uˤː	u_?\:	円唇咽頭後舌狭長母音	روص	/ " s_?\ u_?\: r /	u
e	e	非円唇前舌中央短母音	إنْتَرْنِت	/ " s e n t /	e
eː	e:	非円唇前舌中央長母音	إيش	/ " ? e: S /	e
ɔ	O	円唇後舌半広短母音	دولار	/ d O . " l A r /	O
ɔː	O:	円唇後舌半広長母音	لون	/ " l O: n /	O

カタロニア語 (ca-ES)

以下の表では、国際音声記号 (IPA) の音素および拡張 SAM 音声記号 (X-SAMPA) の記号の完全なリスト、および Amazon Polly でサポートされるカタロニア語の音声 (Arlet) の対応するビゼームについて説明しています。

音素/ビゼームテーブル

IPA	X-SAMPA	説明	例	ビゼーム
子音				
p	p	無声両唇破裂音	ploure	p
t	t	無声歯茎破裂音	Tarragona	t
k	k	無声軟口蓋破裂音	com	k
b	b	有声両唇破裂音	bata	p
d	d	有声歯茎破裂音	endoll	t
g	Amazon EC2 のセキュリティグループ	有声軟口蓋破裂音	gros	k
m	m	有声両唇鼻音	manera	p
n	n	有声歯茎鼻音	donar	t
ɲ	J	有声硬口蓋鼻音	any	J
ŋ	N	有声軟口蓋鼻音	pingüí	k
ɬ	5	有声軟口蓋歯茎側面接近音 (dark l)	albercoc	l
ʎ	L	有声硬口蓋側面接近音	llop	J
r	r	有声歯茎ふるえ音	parra	r

IPA	X-SAMPA	説明	例	ビゼーム
ɾ	4	有声歯茎弾音	para	t
f	f	無声唇歯摩擦音	èmfasi	f
s	s	無声歯茎摩擦音	sac	s
z	z	有声歯茎摩擦音	calzes	s
ʃ	S	無声後部歯茎摩擦音	guix	S
ʒ	Z	有声後部歯茎摩擦音	col·legi	S
͡tʃ	tS	無声後部歯茎破擦音	cotxe	S
͡dʒ	dZ	有声後部歯茎破擦音	platja	S
β	B	有声両唇接近音	obert	B
ð	D	有声歯接近音	bedoll	T
j	j	有声硬口蓋接近音	noia	i
ɣ	G	有声軟口蓋接近音	pega	k
v	v	有声唇歯摩擦音	afgà	f
w	w	有声両唇軟口蓋接近音	aigua	u
x	x	無声軟口蓋摩擦音	Jiménez	k
ʝ	j\	有声硬口蓋摩擦音	yeso	J
l	l	有声歯茎側面接近音	alondra	t

IPA	X-SAMPA	説明	例	ビゼーム
θ	T	無声歯摩擦音	González	T
母音				
a	a	後舌広母音	casa	a
e	e	非円唇前舌半狭母音	llenya	e
ɛ	E	非円唇前舌半広母音	xec	E
i	i	非円唇前舌閉母音	visca	i
o	o	円唇後舌半狭母音	gos	o
ɔ	O	円唇後舌半広母音	joc	O
u	u	円唇後舌閉母音	un	u
ə	@	中段中舌	casa	@
その他の記号				
ˈ	"	第一アクセント	Alabama	
ˌ	%	第二アクセント	Alabama	
.	.	音節境界	A.la.ba.ma	

中国語 (広東語) (yue-CN)

以下の表は、Amazon Polly でサポートされている広東語の粤拼および国際音声記号 (IPA) の音素を示しています。粤拼は広東語のローマ字表記法で、学界や広東語話者の間ではよく使われています。IPA と X-SAMPA はあまり使用されませんが、英語のサポートのために利用できます。表の IPA と X-SAMPA 記号は参照用のみであり、中国語の書き起こしには使用しないでください。粤拼の例と対応するビゼームも示されています。

粤拼の音声発音を Amazon Polly で使用するには、phoneme alphabet="x-amazon-*jyutping*" タグを使用します。

以下の例は、標準別になっています。

粤拼:

```
<speak>
     ## <phoneme alphabet="x-amazon-jyutping" ph="sing2">#</phoneme>#
     ## <phoneme alphabet="x-amazon-jyutping" ph="seng2">#</phoneme>#
</speak>
```

IPA

```
<speak>
     ## <phoneme alphabet="ipa" ph="p##k##n">pecan</phoneme>#
     ## <phoneme alphabet="ipa" ph="#pi.kæn">pecan</phoneme>#
</speak>
```

X-SAMPA

```
<speak>
     ## <phoneme alphabet='x-sampa' ph='pI"kA:n'>pecan</phoneme>#
     ## <phoneme alphabet='x-sampa' ph='"pi.k{n'>pecan</phoneme>#
</speak>
```

> **Note**
>
> Amazon Polly では、UTF-8 コードで入力された広東語のみ、使用することができます。

音素/ビゼームテーブル

粤拼	IPA	X-SAMPA	説明	粤拼の例	ビゼーム
子音					
b	p	p	無声両唇破裂音	巴, baa1	p
c	ts^h	ts_h	有気無声歯茎破擦音	叉, caa1	s

粵拼	IPA	X-SAMPA	説明	粵拼の例	ビゼーム
d	t	t	無声歯茎破裂音	打, daa2	t
f	f	f	無声唇歯摩擦音	花, faa1	f
g	k	k	無声軟口蓋破裂音	家, gaa1	k
gw	k^w	k_w	唇音化無声軟口蓋破裂音	瓜, gwaa1	u
h	h	h	無声声門摩擦音	哈, haa1	k
k	k^h	k_h	有気無声軟口蓋破裂音	卡, kaa1	k
kw	k^{wh}	k_wh	唇音化有気無声軟口蓋破裂音	誇, kwaa1	u
l	l	l	歯茎側面接近音	啦, laa1	t
m	m	m	両唇鼻音	媽, maa1	p
m	m	m=	音節両唇鼻音	唔, m4	p
ng	ŋ	N	軟口蓋鼻音	牙, ngaa4	k
ng	ŋ	N=	音節軟口蓋鼻音	吳, ng4	k
n	n	n	歯茎鼻音	拿, naa4	t
p	p^h	p_h	有気無声両唇破裂音	趴, paa1	p
s	s	s	無声歯茎摩擦音	沙, saa1	s
t	t^h	t_h	有気無声歯茎破裂音	他, taa1	t
w	w	w	有声両唇軟口蓋接近音	娃, waa1	u
y	j	j	硬口蓋接近音	也, jaa5	i
z	ts	ts	無声歯茎破擦音	渣, zaa1	s

粵拼	IPA	X-SAMPA	説明	粵拼の例	ビゼーム
母音					
a	ɐ	6	中舌狭めの広母音	吉, gat1	a
aa	ɑ	A	非円唇後舌広母音	家, gaa1	a
aai	ɑi	Ai	二重母音	街, gaai1	a
aau	ɑu	Au	二重母音	交, gaau1	a
ai	ɐi	6i	二重母音	雞, gai1	a
au	ɐu	6u	二重母音	溝, kau1	a
e	ɛ	E	非円唇前舌半広母音	爹, de1	E
ei	ei	ei	二重母音	基, gei1	e
eo	ɵ	8	円唇中舌半狭母音	春, ceon1	o
eoi	ɵy	8y	二重母音	居, geoi1	o
eu	ɛu	Eu	二重母音	掉 in 掉垃圾, deu6	E
i	i	i	非円唇前舌狭母音	斯, si1	i
i	I	I	非円唇前舌め広めの狭母音	激, gik1	i
iu	iu	iu	二重母音	驕, giu1	i
o	ɔ	O	円唇後舌半広母音	哥, go1	O
oe	œ	9	円唇前舌半広母音	鋸, goe3	O
oi	ɔi	Oi	二重母音	該, goi1	O
ou	ou	ou	二重母音	高, gou1	o

粤拼	IPA	X-SAMPA	説明	粤拼の例	ビゼーム
u	u	u	円唇後舌狭母音	姑, gu1	u
u	ʊ	U	円唇後舌め広めの狭母音	谷, guk5	u
ui	ui	ui	二重母音	劫, gui6	u
yu	y	y	円唇前舌狭母音	於, jyu1	u
トーン記号および追加記号					
1			高平調	詩, si1	
2			中上昇調	史, si2	
3			中平坦調	試, si3	
4			超低平坦超	時, si4	
5			低上昇調	市, si5	
6			低平坦超	是, si6	
-	.	.	音節境界	語音 jyu5-jam1	

標準中国語 (cmn-CN)

以下の表は、Amazon Polly でサポートされている標準中国語の拼音および国際音声記号 (IPA) の音素を示しています。拼音は、標準中国語の口ーマ字化の国際標準です。IPA と X-SAMPA はあまり使用されませんが、英語のサポートのために利用できます。表の IPA と X-SAMPA 記号は参照用のみであり、中国語の書き起こしには使用しないでください。拼音の例と対応するビゼームも示されています。

拼音の音声発音を Amazon Polly で使用するには、`phoneme alphabet="x-amazon-`*`phonetic standard used`*`"` タグを使用します。

以下の例は、標準別になっています。

拼音:

```
<speak>
     ## <phoneme alphabet="x-amazon-pinyin" ph="bo2">#</phoneme>#
     ## <phoneme alphabet="x-amazon-pinyin" ph="bao2">#</phoneme>#
</speak>
```

IPA

```
<speak>
     ## <phoneme alphabet="ipa" ph="p##k##n">pecan</phoneme>#
     ## <phoneme alphabet="ipa" ph="#pi.kæn">pecan</phoneme>#
</speak>
```

X-SAMPA

```
<speak>
     ## <phoneme alphabet='x-sampa' ph='pI"kA:n'>pecan</phoneme>#
     ## <phoneme alphabet='x-sampa' ph='"pi.k{n'>pecan</phoneme>#
</speak>
```

> **Note**
>
> Amazon Polly では、UTF-8 コードで入力された標準中国語のみ、使用することができます。現在、Amazon Polly では GB 18030 規格はサポートされていません。

音素/ビゼームテーブル

拼音	IPA	X-SAMPA	説明	拼音の例	ビゼーム
子音					
f	f	f	無声唇歯摩擦音	发、fa1	f
h	h	h	無声声門摩擦音	和、he2	k
Amazon EC2 の	k	k	無声軟口蓋破裂音	古、gu3	k

拼音	IPA	X-SAMPA	説明	拼音の例	ビゼーム
セキュリティグループ					
k	kʰ	k_h	有気無声軟口蓋破裂音	苦、ku3	k
l	l	l	歯茎側面接近音	拉、la1	t
m	m	m	両唇鼻音	骂、ma4	p
n	n	n	歯茎鼻音	那、na4	t
ng	ŋ	N	軟口蓋鼻音	正、zheng4	k
b	p	p	無声両唇破裂音	爸、ba4	p
p	pʰ	p_h	有気無声両唇破裂音	怕、pa4	p
s	s	s	無声歯茎摩擦音	四、si4	s
x	ɕ	s\	無声歯茎硬口蓋摩擦音	西、xi1	J
sh	ʂ	s`	無声そり舌摩擦音	是、shi4	S
d	t	t	無声歯茎破裂音	打、da3	t
t	tʰ	t_h	有気無声歯茎破裂音	他、ta1	t
zh	͡tʂ	t`s`	無声そり舌破擦音	之、zhi1	S
ch	͡tʂʰ	t`s`_h	無声有気反舌破裂音	吃、chi1	S
s	͡ts	ts	無声歯茎破擦音	字、zi4	s
j	͡tɕ	ts\	無声歯茎硬口蓋破擦音	鸡、ji1	J
q	͡tɕʰ	ts_h	有気無声歯茎硬口蓋破擦音	七、qi1	J

拼音	IPA	X-SAMPA	説明	拼音の例	ビゼーム
c	͡tsʰ	ts_h	有気無声歯茎破擦音	次、ci4	s
w	w	w	有声両唇軟口蓋接近音	我、wo3	u
r	ʐ	z`	有声そり舌摩擦音	日、ri4	S
着色された「er」および「r」音節					
er	ɚ	@`	中段 R 音声母音	二、er4	@
-r			r で着色された音節	馅儿、xianr4	@
母音					
e	ɤ	7	非円唇後舌半狭母音	恶、e4	e
e	ə	@	中段中舌	恩、en1	@
a	a	a	非円唇前舌広母音	安、an1	a
ai	aɪ	aI	二重母音	爱、ai4	a
ao	aʊ	aU	二重母音	奥、ao4	a
ei	eɪ	e	二重母音	诶、ei4	e
e	ɛ	E	非円唇前舌半広母音	姐、jie3	E
i	i	i	非円唇前舌狭母音	鸡、ji1	i
ou	oʊ	oU	二重母音	欧、ou1	o
o	ɔ	O	円唇後舌半広母音	哦、o4	o
u	u	u	円唇後舌狭母音	主、zhu3	u
yu	y	y	円唇前舌狭母音	于、yu2	u
トーン記号および追加記号					

拼音	IPA	X-SAMPA	説明	拼音の例	ビゼーム
1			高平調	淤、yu1	
2			上昇調	鱼、yu2	
3			低 (下降上昇) 調	语、yu3	
4			下降調	育、yu4	
0			中立調	的、de0	
-	.	.	音節境界	语音 yu3-yin1	

デンマーク語 (da-DK)

以下の表では、国際音声記号 (IPA) の音素および拡張 SAM 音声記号 (X-SAMPA) の記号の完全なリスト、および Amazon Polly でサポートされるデンマーク語音声の対応するビゼームについて説明しています。

音素/ビゼームテーブル

IPA	X-SAMPA	説明	例	ビゼーム
子音				
b	b	有声両唇破裂音	bat	p
d	d	有声歯茎破裂音	da	t
ð	D	有声歯摩擦音	mad、thriller	T
f	f	無声唇歯摩擦音	fat	f
Amazon EC2 のセキュリティ	Amazon EC2 のセキュリティグループ	有声軟口蓋破裂音	gat	k

IPA	X-SAMPA	説明	例	ビゼーム
グループ				
h	h	無声声門摩擦音	hat	k
j	j	硬口蓋接近音	jo	i
k	k	無声軟口蓋破裂音	kat	k
l	l	歯茎側面接近音	ladt	t
m	m	両唇鼻音	mat	p
n	n	歯茎鼻音	nay	t
ŋ	N	軟口蓋鼻音	lang	k
p	p	無声両唇破裂音	pande	p
r	r	歯茎ふるえ音	thriller、story	r
ʁ	R	有声口蓋垂摩擦音	rat	k
s	s	無声歯茎摩擦音	sat	s
t	t	無声歯茎破裂音	tal	t
v	v	有声唇歯摩擦音	vat	f
w	w	有声両唇軟口蓋接近音	hav、weekend	u
母音				
ø	2	円唇前舌半狭母音	øst	o
ø:	2:	円唇前舌半狭長母音	øse	o
ɐ	6	中舌狭めの広母音	mor	a

IPA	X-SAMPA	説明	例	ビゼーム
œ	9	円唇前舌半広母音	skøn、grønt	O
œ:	9:	円唇前舌半広長母音	høne、gøre	O
ə	@	中段中舌	ane	@
æ:	{:	非円唇前舌狭めの広長母音	male	a
a	a	非円唇前舌広母音	man	a
æ	{	非円唇前舌狭めの広母音	adresse	a
ɑ	A	非円唇後舌広母音	lak、tak	a
ɑ:	A:	非円唇後舌広長母音	rase	a
e	e	非円唇前舌半狭母音	midt	e
e:	e:	非円唇前舌半狭長母音	mele	e
ɛ	E	非円唇前舌半広母音	mæt	E
ɛ:	E:	非円唇前舌半広長母音	mæle	E
i	i	非円唇前舌狭母音	mit	i
インスタンス:	インスタンス:	非円唇前舌狭長母音	mile	i
o	o	円唇後舌半狭母音	foto	o

IPA	X-SAMPA	説明	例	ビゼーム
オ:	オ:	円唇後舌半狭長母音	mole	o
ɔ	O	円唇後舌半広母音	mund	O
ɔ:	O:	円唇後舌半広長母音	måle	O
ɒ:	Q:	円唇後舌広長母音	morse	O
u	u	円唇後舌狭母音	lusk	u
u:	u:	円唇後舌狭長母音	mule	u
ʌ	V	非円唇後舌半広母音	kører	E
y	y	円唇前舌狭母音	yt	u
y:	y:	円唇前舌狭長母音	hyle	u
その他の記号				
ˈ	"	第一アクセント	Alabama	
ˌ	%	第二アクセント	Alabama	
.	.	音節境界	A.la.ba.ma	

オランダ語 (ベルギー) (nl-BE)

以下の表では、国際音声記号 (IPA) の音素および拡張 SAM 音声記号 (X-SAMPA) の記号の完全なリスト、および Amazon Polly でサポートされるベルギーオランダ語 (フランドル後) 音声の対応するビゼームについて説明しています。

音素/ビゼームテーブル

IPA	X-SAMPA	説明	例	ビゼーム
子音				
b	b	有声両唇破裂音	bak	p
d	d	有声歯茎破裂音	dak	t
ʤ	dZ	有声後部歯茎破擦音	manager	S
f	f	無声唇歯摩擦音	fel	f
Amazon EC2 のセキュリティグループ	Amazon EC2 のセキュリティグループ	有声軟口蓋破裂音	goal	k
ɣ	G	有声軟口蓋摩擦音	hoed	k
ɦ	h\	有声声門摩擦音	hand	k
j	j	硬口蓋接近音	ja	i
k	k	無声軟口蓋破裂音	kap	k
l	l	歯茎側面接近音	land	t
m	m	両唇鼻音	met	p
n	n	歯茎鼻音	net	t
ŋ	N	軟口蓋鼻音	bang	k
p	p	無声両唇破裂音	pak	p
r	r	歯茎ふるえ音	rand	r

IPA	X-SAMPA	説明	例	ビゼーム
s	s	無声歯茎摩擦音	sein	s
ʃ	S	無声後部歯茎摩擦音	show	S
t	t	無声歯茎破裂音	tak	t
v	v	有声唇歯摩擦音	vel	f
ʋ	v\	唇歯接近音	wit	f
x	x	無声軟口蓋摩擦音	toch	k
z	z	有声歯茎摩擦音	ziin	s
ʒ	Z	有声後部歯茎摩擦音	bagage	S
母音				
øː	2:	円唇前舌半狭長母音	neus	o
œy	9y	二重母音	buit	O
ə	@	中段中舌	de	@
aː	a:	非円唇前舌広長母音	baad	a
ɑː	A	非円唇後舌広母音	bad	a
eː	e:	非円唇前舌半狭長母音	beet	e
ɜː	3:	非円唇中舌半広長母音	barrière	E

IPA	X-SAMPA	説明	例	ビゼーム
ɛ	E	非円唇前舌半広母音	bed	E
ɛi	Ei	二重母音	beet	E
i	i	非円唇前舌狭母音	vier	i
ɪ	I	非円唇前舌め広めの狭母音	pit	i
オ:	オ:	円唇後舌半狭長母音	boot	o
ɔ	O	円唇後舌半広母音	pot	O
u	u	円唇後舌狭母音	hoed	u
ʌu	Vu	二重母音	fout	E
y:	y:	円唇前舌狭長母音	fuut	u
ʏ	Y	円唇前舌め広めの狭母音	hut	u
その他の記号				
ˈ	"	第一アクセント	Alabama	
ˌ	%	第二アクセント	Alabama	
.	.	音節境界	A.la.ba.ma	

オランダ語 (nl-NL)

以下の表では、国際音声記号 (IPA) の音素および拡張 SAM 音声記号 (X-SAMPA) の記号の完全なリスト、および Amazon Polly でサポートされるオランダ語音声の対応するビゼームについて説明しています。

音素/ビゼームテーブル

IPA	X-SAMPA	説明	例	ビゼーム
子音				
b	b	有声両唇破裂音	bak	p
d	d	有声歯茎破裂音	dak	t
ʤ	dZ	有声後部歯茎破擦音	manager	S
f	f	無声唇歯摩擦音	fel	f
Amazon EC2 のセキュリティグループ	Amazon EC2 のセキュリティグループ	有声軟口蓋破裂音	goal	k
ɣ	G	有声軟口蓋摩擦音	hoed	k
ɦ	h\	有声声門摩擦音	hand	k
j	j	硬口蓋接近音	ja	i
k	k	無声軟口蓋破裂音	kap	k
l	l	歯茎側面接近音	land	t
m	m	両唇鼻音	met	p
n	n	歯茎鼻音	net	t
ŋ	N	軟口蓋鼻音	bang	k
p	p	無声両唇破裂音	pak	p
r	r	歯茎ふるえ音	rand	r

IPA	X-SAMPA	説明	例	ビゼーム
s	s	無声歯茎摩擦音	sein	s
ʃ	S	無声後部歯茎摩擦音	show	S
t	t	無声歯茎破裂音	tak	t
v	v	有声唇歯摩擦音	vel	f
ʋ	v\	唇歯接近音	wit	f
x	x	無声軟口蓋摩擦音	toch	k
z	z	有声歯茎摩擦音	ziin	s
ʒ	Z	有声後部歯茎摩擦音	bagage	S
母音				
øː	2:	円唇前舌半狭長母音	neus	o
œy	9y	二重母音	buit	O
ə	@	中段中舌	de	@
aː	a:	非円唇前舌広長母音	baad	a
ɑː	A	非円唇後舌広母音	bad	a
eː	e:	非円唇前舌半狭長母音	beet	e
ɜː	3:	非円唇中舌半広長母音	barrière	E

IPA	X-SAMPA	説明	例	ビゼーム
ɛ	E	非円唇前舌半広母音	bed	E
ɛi	Ei	二重母音	beet	E
i	i	非円唇前舌狭母音	vier	i
ɪ	I	非円唇前舌め広めの狭母音	pit	i
オ:	オ:	円唇後舌半狭長母音	boot	o
ɔ	O	円唇後舌半広母音	pot	O
u	u	円唇後舌狭母音	hoed	u
ʌu	Vu	二重母音	fout	E
y:	y:	円唇前舌狭長母音	fuut	u
ʏ	Y	円唇前舌め広めの狭母音	hut	u
その他の記号				
ˈ	"	第一アクセント	Alabama	
ˌ	%	第二アクセント	Alabama	
.	.	音節境界	A.la.ba.ma	

英語 (米国) (en-US)

以下の表では、国際音声記号 (IPA) の音素および拡張 SAM 音声記号 (X-SAMPA) の記号の完全なリスト、および Amazon Polly でサポートされるアメリカ英語音声の対応するビゼームについて説明しています。

音素/ビゼームテーブル

IPA	X-SAMPA	説明	例	ビゼーム
子音				
b	b	有声両唇破裂音	bed	p
d	d	有声歯茎破裂音	dig	t
ʧ	dZ	有声後部歯茎破擦音	jump	S
ð	D	有声歯摩擦音	次に	T
f	f	無声唇歯摩擦音	five	f
g	Amazon EC2 のセキュリティグループ	有声軟口蓋破裂音	game	k
h	h	無声声門摩擦音	house	k
j	j	硬口蓋接近音	はい	i
k	k	無声軟口蓋破裂音	cat	k
l	l	歯茎側面接近音	lay	l
m	m	両唇鼻音	mouse	p
n	n	歯茎鼻音	nap	t
ŋ	N	軟口蓋鼻音	thing	k
p	p	無声両唇破裂音	speak	p
ɹ	r\	歯茎接近音	red	r
s	s	無声歯茎摩擦音	seem	s

IPA	X-SAMPA	説明	例	ビゼーム
ʃ	S	無声後部歯茎摩擦音	ship	S
t	t	無声歯茎破裂音	trap	t
t͡ʃ	tS	無声後部歯茎破擦音	chart	S
θ	T	無声歯摩擦音	thin	T
v	v	有声唇歯摩擦音	vest	f
w	w	有声両唇軟口蓋接近音	west	u
z	z	有声歯茎摩擦音	zero	s
ʒ	Z	有声後部歯茎摩擦音	vision	S
母音				
ə	@	中段中舌	arena	@
ɚ	@`	中段 R 音声母音	reader	@
æ	{	非円唇前舌狭めの広母音	trap	a
aɪ	aI	二重母音	price	a
aʊ	aU	二重母音	mouth	a
ɑ	A	非円唇後舌広長母音	father	a
eɪ	eI	二重母音	face	e

IPA	X-SAMPA	説明	例	ビゼーム
ɝ	3`	非円唇中舌半広 R 音声母音	nurse	E
ɛ	E	非円唇前舌半広母音	dress	E
i	i	非円唇前舌狭長母音	fleece	i
ɪ	I	非円唇前舌め広めの狭母音	kit	i
oʊ	oU	二重母音	goat	o
ɔ	O	円唇後舌半広長母音	thought	O
ɔɪ	OI	二重母音	choice	O
u	u	円唇後舌狭長母音	goose	u
ʊ	U	円唇後舌め広めの狭母音	foot	u
ʌ	V	open-mid-back 円唇なし母音	strut	E
その他の記号				
ˈ	"	第一アクセント	Alabama	
ˌ	%	第二アクセント	Alabama	
.	.	音節境界	A.la.ba.ma	

英語 (オーストラリア) (en-AU)

以下の表では、国際音声記号 (IPA) の音素および拡張 SAM 音声記号 (X-SAMPA) の記号の完全なリスト、および Amazon Polly でサポートされるオーストラリア英語音声の対応するビゼームについて説明しています。

音素/ビゼームテーブル

IPA	X-SAMPA	説明	例	ビゼーム
子音				
b	b	有声両唇破裂音	bed	p
d	d	有声歯茎破裂音	dig	t
ʤ	dZ	有声後部歯茎破擦音	jump	S
ð	D	有声歯摩擦音	次に	T
f	f	無声唇歯摩擦音	five	f
Amazon EC2 のセキュリティグループ	Amazon EC2 のセキュリティグループ	有声軟口蓋破裂音	game	k
h	h	無声声門摩擦音	house	k
j	j	硬口蓋接近音	はい	i
k	k	無声軟口蓋破裂音	cat	k
l	l	歯茎側面接近音	lay	t
l̩	l=	音節歯茎側面接近音	battle	t

IPA	X-SAMPA	説明	例	ビゼーム
m	m	両唇鼻音	mouse	p
m̩	m=	音節両唇鼻音	anthem	p
n	n	歯茎鼻音	nap	t
n̩	n=	音節歯茎鼻音	ボタン	t
ŋ	N	軟口蓋鼻音	thing	k
p	p	無声両唇破裂音	pin	p
ɹ	r\	歯茎接近音	red	r
s	s	無声歯茎摩擦音	seem	s
ʃ	S	無声後部歯茎摩擦音	ship	S
t	t	無声歯茎破裂音	task	t
͡tʃ	tS	無声後部歯茎破擦音	chart	S
Θ	T	無声歯摩擦音	thin	T
v	v	有声唇歯摩擦音	vest	f
w	w	有声両唇軟口蓋接近音	west	u
z	z	有声歯茎摩擦音	zero	s
ʒ	Z	有声後部歯茎摩擦音	vision	S
母音				
ə	@	中段中舌	arena	@

IPA	X-SAMPA	説明	例	ビゼーム
əʊ	@U	二重母音	goat	@
æ	{	非円唇前舌狭めの広母音	trap	a
aɪ	aI	二重母音	price	a
aʊ	aU	二重母音	mouth	a
ɑː	A:	非円唇後舌広長母音	father	a
eɪ	eI	二重母音	face	e
ɜː	3:	非円唇中舌半広長母音	nurse	E
ɛ	E	非円唇前舌半広母音	dress	E
ɛə	E@	二重母音	square	E
インスタンス:	i	非円唇前舌狭長母音	fleece	i
ɪ	I	非円唇前舌め広めの狭母音	kit	i
ɪə	I@	二重母音	near	i
ɔː	OI	円唇後舌半広長母音	thought	O
ɔɪ	OI	二重母音	choice	O
ɒ	Q	円唇後舌広母音	lot	O
uː	u:	円唇後舌狭長母音	goose	u

IPA	X-SAMPA	説明	例	ビゼーム
ʊ	U	円唇後舌め広めの狭母音	foot	u
ʊə	U@	二重母音	cure	u
ʌ	V	O pen-mid-back 非円唇母音	strut	E
その他の記号				
ˈ	"	第一アクセント	Alabama	
ˌ	%	第二アクセント	Alabama	
.	.	音節境界	A.la.ba.ma	

英語 (英国) (en-GB)

以下の表では、国際音声記号 (IPA) の音素および拡張 SAM 音声記号 (X-SAMPA) の記号の完全なリスト、および Amazon Polly でサポートされるイギリス英語音声の対応するビゼームについて説明しています。

音素/ビゼームテーブル

IPA	X-SAMPA	説明	例	ビゼーム
子音				
b	b	有声両唇破裂音	bed	p
d	d	有声歯茎破裂音	dig	t
ʤ	dZ	有声後部歯茎破擦音	jump	S
ð	D	有声歯摩擦音	次に	T
f	f	無声唇歯摩擦音	five	f

IPA	X-SAMPA	説明	例	ビゼーム
Amazon EC2 のセキュリティグループ	Amazon EC2 のセキュリティグループ	有声軟口蓋破裂音	game	k
h	h	無声声門摩擦音	house	k
j	j	硬口蓋接近音	はい	i
k	k	無声軟口蓋破裂音	cat	k
l	l	歯茎側面接近音	lay	t
l̩	l=	音節歯茎側面接近音	battle	t
m	m	両唇鼻音	mouse	p
m̩	m=	音節両唇鼻音	anthem	p
n	n	歯茎鼻音	nap	t
n̩	n=	音節歯茎鼻音	ボタン	t
ŋ	N	軟口蓋鼻音	thing	k
p	p	無声両唇破裂音	pin	p
ɹ	r\	歯茎接近音	red	r
s	s	無声歯茎摩擦音	seem	s
ʃ	S	無声後部歯茎摩擦音	ship	S
t	t	無声歯茎破裂音	task	t

IPA	X-SAMPA	説明	例	ビゼーム
͡tʃ	tS	無声後部歯茎破擦音	chart	S
Θ	T	無声歯摩擦音	thin	T
v	v	有声唇歯摩擦音	vest	f
w	w	有声両唇軟口蓋接近音	west	u
z	z	有声歯茎摩擦音	zero	s
ʒ	Z	有声後部歯茎摩擦音	vision	S
母音				
ə	@	中段中舌	arena	@
əʊ	@U	二重母音	goat	@
æ	{	非円唇前舌狭めの広母音	trap	a
aɪ	aI	二重母音	price	a
aʊ	aU	二重母音	mouth	a
ɑː	A:	非円唇後舌広長母音	father	a
eɪ	eI	二重母音	face	e
ɜː	3:	非円唇中舌半広長母音	nurse	E
ɛ	E	非円唇前舌半広母音	dress	E

IPA	X-SAMPA	説明	例	ビゼーム
ɛə	E@	二重母音	square	E
インスタンス:	i	非円唇前舌狭長母音	fleece	i
ɪ	I	非円唇前舌め広めの狭母音	kit	i
ɪə	I@	二重母音	near	i
ɔ:	O:	円唇後舌半広長母音	thought	O
ɔɪ	OI	二重母音	choice	O
ɒ	Q	円唇後舌広母音	lot	O
u:	u:	円唇後舌狭長母音	goose	u
ʊ	U	円唇後舌め広めの狭母音	foot	u
ʊə	U@	二重母音	cure	u
ʌ	V	O pen-mid-back 非円唇母音	strut	E
その他の記号				
ˈ	"	第一アクセント	Alabama	
ˌ	%	第二アクセント	Alabama	
.	.	音節境界	A.la.ba.ma	

英語 (インド) (en-IN)

以下の表では、国際音声記号 (IPA) の音素および拡張 SAM 音声記号 (X-SAMPA) の記号の完全なリスト、および Amazon Polly でサポートされるインド英語音声の対応するビゼームについて説明しています。

インド英語と組み合わせて使用される音素の詳細については、「ヒンディー語 (hi-IN)」を参照してください。

音素/ビゼームテーブル

IPA	X-SAMPA	説明	例	ビゼーム
子音				
b	b	有声両唇破裂音	bed	p
d	d	有声歯茎破裂音	dig	t
ʤ	dZ	有声後部歯茎破擦音	jump	S
ð	D	有声歯摩擦音	次に	T
f	f	無声唇歯摩擦音	five	f
Amazon EC2 のセキュリティグループ	Amazon EC2 のセキュリティグループ	有声軟口蓋破裂音	game	k
h	h	無声声門摩擦音	house	k
j	j	硬口蓋接近音	はい	i
k	k	無声軟口蓋破裂音	cat	k
l	l	歯茎側面接近音	lay	t

IPA	X-SAMPA	説明	例	ビゼーム
l̩	l=	音節歯茎側面接近音	battle	t
m	m	両唇鼻音	mouse	p
m̩	m=	音節両唇鼻音	anthem	p
n	n	歯茎鼻音	nap	t
n̩	n=	音節歯茎鼻音	nap	t
ŋ	N	軟口蓋鼻音	thing	k
p	p	無声両唇破裂音	pin	p
ɹ	r\	歯茎接近音	red	r
s	s	無声歯茎摩擦音	seem	s
ʃ	S	無声後部歯茎摩擦音	ship	S
t	t	無声歯茎破裂音	task	t
t͡ʃ	tS	無声後部歯茎破擦音	chart	S
Θ	T	無声歯摩擦音	thin	T
v	v	有声唇歯摩擦音	vest	f
w	w	有声両唇軟口蓋接近音	west	u
z	z	有声歯茎摩擦音	zero	s
ʒ	Z	有声後部歯茎摩擦音	vision	S

IPA	X-SAMPA	説明	例	ビゼーム
母音				
ə	@	中段中舌	arena	@
əʊ	@U	二重母音	goat	@
æ	{	非円唇前舌狭めの広母音	trap	a
aɪ	aI	二重母音	price	a
aʊ	aU	二重母音	mouth	a
ɑː	A:	非円唇後舌広長母音	father	a
eɪ	eI	二重母音	face	e
ɜː	3:	非円唇中舌半広長母音	nurse	E
ɛ	E	非円唇前舌半広母音	dress	E
ɛə	E@	二重母音	square	E
インスタンス:	i	非円唇前舌狭長母音	fleece	i
ɪ	I	非円唇前舌め広めの狭母音	kit	i
ɪə	I@	二重母音	near	i
ɔː	OI	円唇後舌半広長母音	thought	O
ɔɪ	OI	二重母音	choice	O

IPA	X-SAMPA	説明	例	ビゼーム
ɒ	Q	円唇後舌広母音	lot	O
uː	u:	円唇後舌狭長母音	goose	u
ʊ	U	円唇後舌め広めの狭母音	foot	u
ʊə	U@	二重母音	cure	u
ʌ	V	O pen-mid-back 非円唇母音	strut	E
その他の記号				
ˈ	"	第一アクセント	Alabama	
ˌ	%	第二アクセント	Alabama	
.	.	音節境界	A.la.ba.ma	

英語 (アイルランド) (en-IE)

以下の表では、国際音声記号 (IPA) の音素および拡張 SAM 音声記号 (X-SAMPA) の記号の完全なリスト、および Amazon Polly でサポートされるアイルランド英語音声の対応するビゼームについて説明しています。

音素/ビゼームテーブル

IPA	X-SAMPA	説明	例	ビゼーム
子音				
b	b	有声両唇破裂音	bed	p
d	d	有声歯茎破裂音	dig	t
ʤ	dZ	有声後部歯茎破擦音	jump	S

IPA	X-SAMPA	説明	例	ビゼーム
ð	D	有声歯摩擦音	次に	T
f	f	無声唇歯摩擦音	five	f
g	Amazon EC2 のセキュリティグループ	有声軟口蓋破裂音	game	k
h	h	無声声門摩擦音	house	k
j	j	硬口蓋接近音	はい	i
k	k	無声軟口蓋破裂音	cat	k
l	l	歯茎側面接近音	lay	t
m	m	両唇鼻音	mouse	p
n	n	歯茎鼻音	nap	t
ŋ	N	軟口蓋鼻音	thing	k
p	p	無声両唇破裂音	speak	p
ɹ	r\	歯茎接近音	red	r
s	s	無声歯茎摩擦音	seem	s
ʃ	S	無声後部歯茎摩擦音	ship	S
t	t	無声歯茎破裂音	trap	t
‿tʃ	tS	無声後部歯茎破擦音	chart	S
θ	T	無声歯摩擦音	thin	T
v	v	有声唇歯摩擦音	vest	f

IPA	X-SAMPA	説明	例	ビゼーム
w	w	有声両唇軟口蓋接近音	west	u
z	z	有声歯茎摩擦音	zero	s
ʒ	Z	有声後部歯茎摩擦音	vision	S
母音				
ə	@	中段中舌	arena	@
ɚ	@`	中段 R 音声母音	reader	@
æ	{	非円唇前舌狭めの広母音	trap	a
aɪ	aI	二重母音	price	a
aʊ	aU	二重母音	mouth	a
ɑ	A	非円唇後舌広長母音	father	a
eɪ	eI	二重母音	face	e
ɝ	3`	非円唇中舌半広 R 音声母音	nurse	E
ɛ	E	非円唇前舌半広母音	dress	E
i	i	非円唇前舌狭長母音	fleece	i
ɪ	I	非円唇前舌め広めの狭母音	kit	i
oʊ	oU	二重母音	goat	o

IPA	X-SAMPA	説明	例	ビゼーム
ɔ	O	円唇後舌半広長母音	thought	O
ɔɪ	OI	二重母音	choice	O
u	u	円唇後舌狭長母音	goose	u
ʊ	U	円唇後舌め広めの狭母音	foot	u
ʌ	V	open-mid-back 円唇なし母音	strut	E
その他の記号				
ˈ	"	第一アクセント	Alabama	
ˌ	%	第二アクセント	Alabama	
.	.	音節境界	A.la.ba.ma	

英語 (ニュージーランド) (en-NZ)

以下の表では、国際音声記号 (IPA) の音素および拡張 SAM 音声記号 (X-SAMPA) の記号の完全なリスト、および Amazon Polly でサポートされるニュージーランド英語音声の対応するビゼームについて説明しています。

音素/ビゼームテーブル

IPA	X-SAMPA	説明	例	ビゼーム
子音				
b	b	有声両唇破裂音	bed	p
d	d	有声歯茎破裂音	dig	t

IPA	X-SAMPA	説明	例	ビゼーム
ʤ	dZ	有声後部歯茎破擦音	jump	S
ð	D	有声歯摩擦音	次に	T
f	f	無声唇歯摩擦音	five	f
Amazon EC2 のセキュリティグループ	Amazon EC2 のセキュリティグループ	有声軟口蓋破裂音	game	k
h	h	無声声門摩擦音	house	k
j	j	硬口蓋接近音	はい	i
k	k	無声軟口蓋破裂音	cat	k
l	l	歯茎側面接近音	lay	t
l̩	l=	音節歯茎側面接近音	battle	t
m	m	両唇鼻音	mouse	p
m̩	m=	音節両唇鼻音	anthem	p
n	n	歯茎鼻音	nap	t
n̩	n=	音節歯茎鼻音	ボタン	t
ŋ	N	軟口蓋鼻音	thing	k
p	p	無声両唇破裂音	pin	p
ɹ	r\	歯茎接近音	red	r

IPA	X-SAMPA	説明	例	ビゼーム
s	s	無声歯茎摩擦音	seem	s
ʃ	S	無声後部歯茎摩擦音	ship	S
t	t	無声歯茎破裂音	task	t
͡tʃ	tS	無声後部歯茎破擦音	chart	S
Θ	T	無声歯摩擦音	thin	T
v	v	有声唇歯摩擦音	vest	f
w	w	有声両唇軟口蓋接近音	west	u
z	z	有声歯茎摩擦音	zero	s
ʒ	Z	有声後部歯茎摩擦音	vision	S
母音				
ə	@	中段中舌	arena	@
əʊ	@U	二重母音	goat	@
æ	{	非円唇前舌狭めの広母音	trap	a
aɪ	aI	二重母音	price	a
aʊ	aU	二重母音	mouth	a
ɑː	A:	非円唇後舌広長母音	father	a
eɪ	eI	二重母音	face	e

IPA	X-SAMPA	説明	例	ビゼーム
ɜː	3:	非円唇中舌半広長母音	nurse	E
ɛ	E	非円唇前舌半広母音	dress	E
ɛə	E@	二重母音	square	E
インスタンス:	i	非円唇前舌狭長母音	fleece	i
ɪ	I	非円唇前舌め広めの狭母音	kit	i
ɪə	I@	二重母音	near	i
ɔː	O:	円唇後舌半広長母音	thought	O
ɔɪ	OI	二重母音	choice	O
ɒ	Q	円唇後舌広母音	lot	O
uː	u:	円唇後舌狭長母音	goose	u
ʊ	U	円唇後舌め広めの狭母音	foot	u
ʊə	U@	二重母音	cure	u
ʌ	V	O pen-mid-back 非円唇母音	strut	E
その他の記号				
ˈ	"	第一アクセント	Alabama	
ˌ	%	第二アクセント	Alabama	

IPA	X-SAMPA	説明	例	ビゼーム
.	.	音節境界	A.la.ba.ma	

Aria の音声はニュージーランド英語です。マオリ語には限定的に対応しています。以下のマオリ語の単語や句を発音することができます。マオリ語の句では、大文字と小文字が区別されます。

英語	マオリ語
Hello/Cheers	Kia ora
Welcome (to)	Nau mai (ki)
Hello (one person)/thank you	Tēnā koe
Hello (three or more people)/thank you	Tēnā koutou
Good morning	Ata mārie
Good morning	Mōrena
Thank you	Ngā mihi
Take care	Ngā manaakitanga
See you	Ka kite
See you later	Mā te wā
Have a good day	Kia pai tō rā
Merry Christmas	Meri Kirihimete
Maori	Māori
Maori language	te reo Māori
Maori language week	Te wiki o te reo Māori
ニュージーランド	Aotearoa

英語	マオリ語
Maori New Year	Mātariki
Town in New Zealand / Waitangi Day is the national day of New Zealand	Waitangi
1	tahi
2	rua
Three	toru
Four	whā
Five	rima
Six	ono
Seven	whitu
Eight	waru
Nine	iwa
Ten	tekau
Twenty	rua tekau
Thirty	Toru tekau

英語 (南アフリカ) (en-ZA)

以下の表では、国際音声記号 (IPA) の音素および拡張 SAM 音声記号 (X-SAMPA) の記号の完全なリスト、および Amazon Polly でサポートされる南アフリカ英語音声の対応するビゼームについて説明しています。

音素/ビゼームテーブル

IPA	X-SAMPA	説明	例	ビゼーム
子音				
b	b	有声両唇破裂音	bed	p
d	d	有声歯茎破裂音	dig	t
ʤ	dZ	有声後部歯茎破擦音	jump	S
ð	D	有声歯摩擦音	次に	T
f	f	無声唇歯摩擦音	five	f
Amazon EC2 のセキュリティグループ	Amazon EC2 のセキュリティグループ	有声軟口蓋破裂音	game	k
h	h	無声声門摩擦音	house	k
j	j	硬口蓋接近音	はい	i
k	k	無声軟口蓋破裂音	cat	k
l	l	歯茎側面接近音	lay	t
l̩	l=	音節歯茎側面接近音	battle	t
ɬ	K	無声側面摩擦音	umhlanga	t
m	m	両唇鼻音	mouse	p
m̩	m=	音節両唇鼻音	anthem	p

IPA	X-SAMPA	説明	例	ビゼーム
n	n	歯茎鼻音	nap	t
ɳ	n=	音節歯茎鼻音	ボタン	t
ŋ	N	軟口蓋鼻音	thing	k
p	p	無声両唇破裂音	pin	p
ɹ	r\	歯茎接近音	red	r
r	r	歯茎ふるえ音	pareis	r
s	s	無声歯茎摩擦音	seem	s
ʃ	S	無声後部歯茎摩擦音	ship	S
t	t	無声歯茎破裂音	task	t
͡tʃ	tS	無声後部歯茎破擦音	chart	S
Θ	T	無声歯摩擦音	thin	T
v	v	有声唇歯摩擦音	vest	f
w	w	有声両唇軟口蓋接近音	west	u
x	x	無声軟口蓋摩擦音	gauteng	k
z	z	有声歯茎摩擦音	zero	s
!	!\	post-alveolar click	gqeberha	k
|	|\	dental click	ncube	t
||	||\	lateral click	xhosa	t
母音				

IPA	X-SAMPA	説明	例	ビゼーム
ə	@	中段中舌	arena	@
əi	@i	二重母音	ネルスプルイ T	i
əʊ	@U	二重母音	goat	@
æ	{	非円唇前舌狭めの広母音	trap	a
aɪ	aI	二重母音	price	a
aʊ	aU	二重母音	mouth	a
ɑː	A:	非円唇後舌広長母音	father	a
eɪ	eI	二重母音	face	e
ɜː	3:	非円唇中舌半広長母音	nurse	E
ɛ	E	非円唇前舌半広母音	dress	E
ɛə	E@	二重母音	square	E
インスタンス:	i	非円唇前舌狭長母音	fleece	i
iə	I@	二重母音	du preez	i
ɪ	I	非円唇前舌め広めの狭母音	kit	i
ɪə	I@	二重母音	near	i
ɔː	O:	円唇後舌半広長母音	thought	O

IPA	X-SAMPA	説明	例	ビゼーム
ɔɪ	OI	二重母音	choice	O
ɒ	Q	円唇後舌広母音	lot	O
uː	u:	円唇後舌狭長母音	goose	u
ʊ	U	円唇後舌め広めの狭母音	foot	u
ʊə	U@	二重母音	cure	u
ʌ	V	O pen-mid-back 非円唇母音	strut	E
y	y	円唇前舌狭母音	van vuuren	u
その他の記号				
ˈ	"	第一アクセント	Alabama	
ˌ	%	第二アクセント	Alabama	
.	.	音節境界	A.la.ba.ma	

英語 (ウェールズ) (en-GB-WLS)

以下の表では、国際音声記号 (IPA) の音素および拡張 SAM 音声記号 (X-SAMPA) の記号の完全なリスト、および Amazon Polly でサポートされるウェールズ英語音声の対応するビゼームについて説明しています。

音素/ビゼームテーブル

IPA	X-SAMPA	説明	例	ビゼーム
子音				
b	b	有声両唇破裂音	bed	p

IPA	X-SAMPA	説明	例	ビゼーム
d	d	有声歯茎破裂音	dig	t
ʤ	dZ	有声後部歯茎破擦音	jump	S
ð	D	有声歯摩擦音	次に	T
f	f	無声唇歯摩擦音	five	f
Amazon EC2 のセキュリティグループ	Amazon EC2 のセキュリティグループ	有声軟口蓋破裂音	game	k
h	h	無声声門摩擦音	house	k
j	j	硬口蓋接近音	はい	i
k	k	無声軟口蓋破裂音	cat	k
l	l	歯茎側面接近音	lay	t
l̩	l=	音節歯茎側面接近音	battle	t
m	m	両唇鼻音	mouse	p
m̩	m=	音節両唇鼻音	anthem	p
n	n	歯茎鼻音	nap	t
n̩	n=	音節歯茎鼻音	nap	t
ŋ	N	軟口蓋鼻音	thing	k
p	p	無声両唇破裂音	pin	p

IPA	X-SAMPA	説明	例	ビゼーム
ɹ	r\	歯茎接近音	red	r
s	s	無声歯茎摩擦音	seem	s
ʃ	S	無声後部歯茎摩擦音	ship	S
t	t	無声歯茎破裂音	task	t
͡tʃ	tS	無声後部歯茎破擦音	chart	S
Θ	T	無声歯摩擦音	thin	T
v	v	有声唇歯摩擦音	vest	f
w	w	有声両唇軟口蓋接近音	west	u
z	z	有声歯茎摩擦音	zero	s
ʒ	Z	有声後部歯茎摩擦音	vision	S
母音				
ə	@	中段中舌	arena	@
əʊ	@U	二重母音	goat	@
æ	{	非円唇前舌狭めの広母音	trap	a
aɪ	aI	二重母音	price	a
aʊ	aU	二重母音	mouth	a
ɑː	A:	非円唇後舌広長母音	father	a

IPA	X-SAMPA	説明	例	ビゼーム
eɪ	eI	二重母音	face	e
ɜː	3:	非円唇中舌半広長母音	nurse	E
ɛ	E	非円唇前舌半広母音	dress	E
ɛə	E@	二重母音	square	E
インスタンス:	i	非円唇前舌狭長母音	fleece	i
ɪ	I	非円唇前舌め広めの狭母音	kit	i
ɪə	I@	二重母音	near	i
ɔː	OI	円唇後舌半広長母音	thought	O
ɔɪ	OI	二重母音	choice	O
ɒ	Q	円唇後舌広母音	lot	O
uː	u:	円唇後舌狭長母音	goose	u
ʊ	U	円唇後舌め広めの狭母音	foot	u
ʊə	U@	二重母音	cure	u
ʌ	V	O pen-mid-back 非円唇母音	strut	E
その他の記号				
ˈ	"	第一アクセント	Alabama	

IPA	X-SAMPA	説明	例	ビゼーム
ˌ	%	第二アクセント	Alabama	
.	.	音節境界	A.la.ba.ma	

フィンランド語 (fi-FI)

以下の表では、国際音声記号 (IPA) の音素および拡張 SAM 音声記号 (X-SAMPA) の記号の完全なリスト、および Amazon Polly でサポートされるフィンランド語音声の対応するビゼームについて説明しています。

音素/ビゼームテーブル

IPA	X-SAMPA	説明	例	ビゼーム
フィンランド語の子音				
p	p	無声両唇破裂音	[p]ankki	p
t	t	無声歯茎破裂音	[t]alo	t
k	k	無声軟口蓋破裂音	[k]aali	k
d	d	有声歯茎破裂音	[d]ata	t
s	s	無声歯茎摩擦音	[s]ali	s
h	h	無声声門摩擦音	[h]attu	k
ʋ	v\	有声唇歯接近音	[v]aiv̄a	v
j	j	硬口蓋接近音	[j]oki	i
l	l	歯茎側面接近音	[l]oma	t
r	r	有声歯茎ふるえ音	[r]iita	r
m	m	両唇鼻音	[m]ato	p

IPA	X-SAMPA	説明	例	ビゼーム
n	n	歯茎鼻音	[n]enäa	t
ŋ	N	軟口蓋鼻音	he[n]ki	k
外来語に含まれる子音				
b	b	有声両唇破裂音	[b]ussi	p
f	f	無声唇歯摩擦音	[f]irma	v
w	w	有声両唇軟口蓋接近音	[w]iki	u
z	z	有声歯茎摩擦音	[z]ulu	s
g	Amazon EC2 のセキュリティグループ	有声軟口蓋破裂音	[g]aala	k
ʃ	S	無声後部歯茎摩擦音	[sh]akki	S
ʒ	Z	有声後部歯茎摩擦音	[g]enre	S
θ	T	無声歯摩擦音	ear[th]	T
ð	D	有声歯摩擦音	ei[th]er	T
短母音				
i	i	非円唇前舌狭母音	k[i]lo	i
ɛ	E	非円唇前舌半広母音	k[e]sä	E
æ	{	非円唇前舌狭めの広母音	k[ä]ly	A

IPA	X-SAMPA	説明	例	ビゼーム
y	y	円唇前舌狭母音	k[y]lä	u
ø	2	円唇前舌半狭母音	p[ö]ly	O
u	u	円唇後舌狭母音	k[u]lo	u
ɔ	O	円唇後舌半広母音	k[o]lo	O
ɑ	A	非円唇後舌広母音	k[a]la	A
長母音				
iː	i:	非円唇前舌狭長母音	s[ii]li	i
ɛː	E:	非円唇前舌半広長母音	[ee]tu	E
æː	{:	非円唇前舌狭めの広長母音	t[ää]llä	A
yː	y:	非円唇前舌狭長母音	t[yy]li	u
øː	2:	円唇前舌半狭長母音	t[öö]lö	O
uː	u:	円唇後舌狭長母音	t[uu]li	u
ɔː	O:	円唇後舌半広長母音	r[oo]li	O
ɑː	A:	非円唇後舌広長母音	k[aa]su	A
二重母音				
ɛi	Ei	二重母音	l[ei]pä	E

IPA	X-SAMPA	説明	例	ビゼーム
æi	{i	二重母音	[äi]ti	A
ui	ui	二重母音	k[ui]n	u
ɑi	Ai	二重母音	k[ai]kki	A
ɔi	Oi	二重母音	p[oi]ka	O
øi	2i	二重母音	s[öi]n	O
yi	yi	二重母音	l[yi]jy	u
ɑu	Au	二重母音	s[au]na	A
ɔu	Ou	二重母音	k[ou]lu	O
ɛu	Eu	二重母音	r[eu]na	E
iu	iu	二重母音	v[iu]lu	i
æy	{y	二重母音	t[äy]nnä	A
øy	2y	二重母音	k[öy]hä	O
眼鏡	Ey	二重母音	pes[ey]tyä	E
iy	iy	二重母音	käär[iy]tyä	i
iɛ	iE	二重母音	t[ie]	i
yø	y2	二重母音	[yö]	u
uɔ	uO	二重母音	t[uo]	u
英語の外来語に含まれる母音				
ɪ	I	非円唇前舌め広めの狭母音	b[i]t	i

IPA	X-SAMPA	説明	例	ビゼーム
ʊ	U	円唇後舌め広めの狭母音	b[oo]k	u
ə	@	中段中舌	[a]bout	@
ʌ	V	open-mid-back 円唇なし母音	c[u]t	E

フランス語 (fr-FR)

以下の表では、国際音声記号 (IPA) の音素および拡張 SAM 音声記号 (X-SAMPA) の記号の完全なリスト、および Amazon Polly でサポートされるフランス語音声の対応するビゼームについて説明しています。

音素/ビゼームテーブル

IPA	X-SAMPA	説明	例	ビゼーム
子音				
b	b	有声両唇破裂音	boire	p
d	d	有声歯茎破裂音	madame	t
f	f	無声唇歯摩擦音	femme	f
Amazon EC2 のセキュリティグループ	Amazon EC2 のセキュリティグループ	有声軟口蓋破裂音	grand	k
ɥ	H	有声両唇硬口蓋接近音	bruit	u
j	j	硬口蓋接近音	meilleur	i

IPA	X-SAMPA	説明	例	ビゼーム
k	k	無声軟口蓋破裂音	quatre	k
l	l	歯茎側面接近音	malade	t
m	m	両唇鼻音	maison	p
n	n	歯茎鼻音	astronome	t
ɲ	J	硬口蓋鼻音	baigner	J
ŋ	N	軟口蓋鼻音	parking	k
p	p	無声両唇破裂音	pomme	p
ʁ	R	有声口蓋垂摩擦音	amoureux	k
s	s	無声歯茎摩擦音	santé	s
ʃ	S	無声後部歯茎摩擦音	chat	S
t	t	無声歯茎破裂音	téléphone	t
v	v	有声唇歯摩擦音	vrai	f
w	w	有声両唇軟口蓋接近音	soir	u
z	z	有声歯茎摩擦音	raison	s
ʒ	Z	有声後部歯茎摩擦音	aubergine	S
母音				
ø	2	円唇前舌半狭母音	deux	o
œ	9	円唇前舌半広母音	neuf	O

IPA	X-SAMPA	説明	例	ビゼーム
œ̃	9~	円唇前舌半広鼻母音	brun	O
ə	@	中段中舌	je	@
a	a	非円唇前舌広母音	table	a
ɑ̃	A~	非円唇後舌広鼻母音	camembert	a
e	e	非円唇前舌半狭母音	marché	e
ɛ	E	非円唇前舌半広母音	neige	E
ɛ̃	E~	非円唇前舌半広鼻母音	sapin	E
i	i	非円唇前舌狭母音	mille	i
o	o	円唇後舌半狭母音	hôpital	o
ɔ	O	円唇後舌半広母音	homme	O
ɔ̃	O~	円唇後舌半広鼻母音	bon	O
u	u	円唇後舌狭母音	sous	u
y	y	円唇前舌狭母音	dur	u
その他の記号				
ˈ	"	第一アクセント	Alabama	
ˌ	%	第二アクセント	Alabama	
.	.	音節境界	A.la.ba.ma	

フランス語 (ベルギー) (fr-BE)

以下の表では、国際音声記号 (IPA) の音素および拡張 SAM 音声記号 (X-SAMPA) の記号の完全なリスト、および Amazon Polly でサポートされるベルギーフランス語音声の対応するビゼームについて説明しています。

音素/ビゼームテーブル

IPA	X-SAMPA	説明	例	ビゼーム
子音				
b	b	有声両唇破裂音	boire	p
d	d	有声歯茎破裂音	madame	t
f	f	無声唇歯摩擦音	femme	f
Amazon EC2 のセキュリティグループ	Amazon EC2 のセキュリティグループ	有声軟口蓋破裂音	grand	k
ɥ	H	有声両唇硬口蓋接近音	bruit	u
j	j	硬口蓋接近音	meilleur	i
k	k	無声軟口蓋破裂音	quatre	k
l	l	歯茎側面接近音	malade	t
m	m	両唇鼻音	maison	p
n	n	歯茎鼻音	astronome	t
ɲ	J	硬口蓋鼻音	baigner	J
ŋ	N	軟口蓋鼻音	parking	k

IPA	X-SAMPA	説明	例	ビゼーム
p	p	無声両唇破裂音	pomme	p
ʁ	R	有声口蓋垂摩擦音	amoureux	k
s	s	無声歯茎摩擦音	santé	s
ʃ	S	無声後部歯茎摩擦音	chat	S
t	t	無声歯茎破裂音	téléphone	t
v	v	有声唇歯摩擦音	vrai	f
w	w	有声両唇軟口蓋接近音	soir	u
z	z	有声歯茎摩擦音	raison	s
ʒ	Z	有声後部歯茎摩擦音	aubergine	S
母音				
ø	2	円唇前舌半狭母音	deux	o
œ	9	円唇前舌半広母音	neuf	O
œ̃	9~	円唇前舌半広鼻母音	brun	O
ə	@	中段中舌	je	@
a	a	非円唇前舌広母音	table	a
ɑ̃	A~	非円唇後舌広鼻母音	camembert	a
e	e	非円唇前舌半狭母音	marché	e

IPA	X-SAMPA	説明	例	ビゼーム
ɛ	E	非円唇前舌半広母音	neige	E
ɛ̃	E~	非円唇前舌半広鼻母音	sapin	E
i	i	非円唇前舌狭母音	mille	i
o	o	円唇後舌半狭母音	hôpital	o
ɔ	O	円唇後舌半広母音	homme	O
ɔ̃	O~	円唇後舌半広鼻母音	bon	O
u	u	円唇後舌狭母音	sous	u
y	y	円唇前舌狭母音	dur	u
その他の記号				
ˈ	"	第一アクセント	Alabama	
ˌ	%	第二アクセント	Alabama	
.	.	音節境界	A.la.ba.ma	

フランス語 (カナダ) (fr-CA)

以下の表では、国際音声記号 (IPA) の音素および拡張 SAM 音声記号 (X-SAMPA) の記号の完全なリスト、および Amazon Polly でサポートされるフランス語 (カナダ) 音声の対応するビゼームについて説明しています。

音素/ビゼームテーブル

IPA	X-SAMPA	説明	例	ビゼーム
子音				

IPA	X-SAMPA	説明	例	ビゼーム
b	b	有声両唇破裂音	boire	p
d	d	有声歯茎破裂音	madame	t
f	f	無声唇歯摩擦音	femme	f
Amazon EC2 のセキュリティグループ	Amazon EC2 のセキュリティグループ	有声軟口蓋破裂音	grand	k
ɥ	H	有声両唇硬口蓋接近音	bruit	u
j	j	硬口蓋接近音	meilleur	i
k	k	無声軟口蓋破裂音	quatre	k
l	l	歯茎側面接近音	malade	t
m	m	両唇鼻音	maison	p
n	n	歯茎鼻音	astronome	t
ɲ	J	硬口蓋鼻音	baigner	J
ŋ	N	軟口蓋鼻音	parking	k
p	p	無声両唇破裂音	pomme	p
ʁ	R	有声口蓋垂摩擦音	amoureux	k
s	s	無声歯茎摩擦音	santé	s
ʃ	S	無声後部歯茎摩擦音	chat	S

IPA	X-SAMPA	説明	例	ビゼーム
t	t	無声歯茎破裂音	téléphone	t
v	v	有声唇歯摩擦音	vrai	f
w	w	有声両唇軟口蓋接近音	soir	u
z	z	有声歯茎摩擦音	raison	s
ʒ	Z	有声後部歯茎摩擦音	aubergine	S
母音				
ø	2	円唇前舌半狭母音	deux	o
œ	9	円唇前舌半広母音	neuf	O
œ̃	9~	円唇前舌半広鼻母音	brun	O
ə	@	中段中舌	je	@
a	a	非円唇前舌広母音	table	a
ɑ̃	A~	非円唇後舌広鼻母音	camembert	a
e	e	非円唇前舌半狭母音	marché	e
ɛ	E	非円唇前舌半広母音	neige	E
ɛ̃	E~	非円唇前舌半広鼻母音	sapin	E
i	i	非円唇前舌狭母音	mille	i

IPA	X-SAMPA	説明	例	ビゼーム
o	o	円唇後舌半狭母音	hôpital	o
ɔ	O	円唇後舌半広母音	homme	O
ɔ̃	O~	円唇後舌半広鼻母音	bon	O
u	u	円唇後舌狭母音	sous	u
y	y	円唇前舌狭母音	dur	u
その他の記号				
ˈ	"	第一アクセント	Alabama	
ˌ	%	第二アクセント	Alabama	
.	.	音節境界	A.la.ba.ma	

ドイツ語 (de-DE)

以下の表では、国際音声記号 (IPA) の音素および拡張 SAM 音声記号 (X-SAMPA) の記号の完全なリスト、および Amazon Polly でサポートされるドイツ語音声の対応するビゼームについて説明しています。

音素/ビゼームテーブル

IPA	X-SAMPA	説明	例	ビゼーム
子音				
ʔ	?	声門破裂音		
b	b	有声両唇破裂音	Bier	p
d	d	有声歯茎破裂音	Dach	t
ç	C	無声硬口蓋摩擦音	ich	k

IPA	X-SAMPA	説明	例	ビゼーム
͡dʒ	dZ	有声後部歯茎破擦音	Dschungel	S
f	f	無声唇歯摩擦音	Vogel	f
Amazon EC2 のセキュリティグループ	Amazon EC2 のセキュリティグループ	有声軟口蓋破裂音	Gabel	k
h	h	無声声門摩擦音	Haus	k
j	j	無声声門摩擦音	jemand	i
k	k	無声軟口蓋破裂音	Kleid	k
l	l	歯茎側面接近音	Loch	t
m	m	両唇鼻音	Milch	p
n	n	歯茎鼻音	Natur	t
ŋ	N	軟口蓋鼻音	klingen	k
p	p	無声両唇破裂音	Park	p
͡pf	pf	無声唇歯破擦音	Apfel	
ʀ	R	口蓋垂ふるえ音	Regen	
s	s	無声歯茎摩擦音	Messer	s
ʃ	S	無声後部歯茎摩擦音	Fischer	S
t	t	無声歯茎破裂音	Topf	T

IPA	X-SAMPA	説明	例	ビゼーム
͡ts	Ts	無声歯茎破擦音	Zahl	
͡tʃ	tS	無声後部歯茎破裂音	deutsch	S
v	v	有声唇歯摩擦音	Wasser	f
x	x	無声軟口蓋摩擦音	kochen	k
z	z	有声歯茎摩擦音	See	s
ʒ	Z	有声後部歯茎摩擦音	Orange	S
母音				
øː	2:	円唇前舌半狭長母音	böse	o
ɐ	6	中舌狭めの広母音	besser	a
ɐ̯	6_^	非音節中舌狭めの広母音	Klar	a
œ	9	円唇前舌半広母音	können	O
ə	@	中段中舌	Rede	@
a	a	非円唇前舌広母音	Salz	a
aː	a:	非円唇前舌広長母音	Sahne	a
aɪ	aI	二重母音	nein	a
aʊ	aU	二重母音	Augen	a
ɑ̃	A~	非円唇後舌広鼻母音	Restaurant	a

IPA	X-SAMPA	説明	例	ビゼーム
e:	e:	非円唇前舌半狭長母音	Rede	e
ɛ	E	非円唇前舌半広母音	Keller	E
ɛ̃	E~	非円唇前舌半広鼻母音	Terrain	E
インスタンス:	インスタンス:	非円唇前舌狭長母音	Lied	i
ɪ	I	非円唇前舌め広めの狭母音	bitte	i
オ:	オ:	円唇後舌半狭長母音	Kohl	o
ɔ	O	円唇後舌半広母音	Koffer	O
ɔ̃	O~	円唇後舌半広鼻母音	Annonce	O
ɔʏ	OY	二重母音	neu	O
u:	u:	円唇後舌狭長母音	Bruder	u
ʊ	U	円唇後舌め広めの狭母音	Wunder	u
y:	y:	円唇前舌狭長母音	kühl	u
ʏ	Y	円唇前舌め広めの狭母音	Küche	u
その他の記号				
ˈ	"	第一アクセント	Alabama	

IPA	X-SAMPA	説明	例	ビゼーム
ˌ	%	第二アクセント	Alabama	
.	.	音節境界	A.la.ba.ma	

ドイツ語 (オーストリア) (de-AT)

以下の表では、国際音声記号 (IPA) の音素および拡張 SAM 音声記号 (X-SAMPA) の記号の完全なリスト、および Amazon Polly でサポートされるオーストリアドイツ語音声の対応するビゼームについて説明しています。

音素/ビゼームテーブル

IPA	X-SAMPA	説明	例	ビゼーム
子音				
ʔ	?	声門破裂音		
b	b	有声両唇破裂音	Bier	p
d	d	有声歯茎破裂音	Dach	t
ç	C	無声硬口蓋摩擦音	ich	k
͡dʒ	dZ	有声後部歯茎破擦音	Dschungel	S
f	f	無声唇歯摩擦音	Vogel	f
Amazon EC2 のセキュリティグループ	Amazon EC2 のセキュリティグループ	有声軟口蓋破裂音	Gabel	k
h	h	無声声門摩擦音	Haus	k

IPA	X-SAMPA	説明	例	ビゼーム
j	j	無声声門摩擦音	jemand	i
k	k	無声軟口蓋破裂音	Kleid	k
l	l	歯茎側面接近音	Loch	t
m	m	両唇鼻音	Milch	p
n	n	歯茎鼻音	Natur	t
ŋ	N	軟口蓋鼻音	klingen	k
p	p	無声両唇破裂音	Park	p
͡pf	pf	無声唇歯破擦音	Apfel	
ʀ	R	口蓋垂ふるえ音	Regen	
s	s	無声歯茎摩擦音	Messer	s
ʃ	S	無声後部歯茎摩擦音	Fischer	S
t	t	無声歯茎破裂音	Topf	T
͡ts	Ts	無声歯茎破擦音	Zahl	
͡tʃ	tS	無声後部歯茎破裂音	deutsch	S
v	v	有声唇歯摩擦音	Wasser	f
x	x	無声軟口蓋摩擦音	kochen	k
z	z	有声歯茎摩擦音	See	s
ʒ	Z	有声後部歯茎摩擦音	Orange	S
母音				

IPA	X-SAMPA	説明	例	ビゼーム
øː	2:	円唇前舌半狭長母音	böse	o
ɐ	6	中舌狭めの広母音	besser	a
ɐ̯	6_^	非音節中舌狭めの広母音	Klar	a
œ	9	円唇前舌半広母音	können	O
ə	@	中段中舌	Rede	@
a	a	非円唇前舌広母音	Salz	a
aː	a:	非円唇前舌広長母音	Sahne	a
aɪ	aI	二重母音	nein	a
aʊ	aU	二重母音	Augen	a
ɑ̃	A~	非円唇後舌広鼻母音	Restaurant	a
eː	e:	非円唇前舌半狭長母音	Rede	e
ɛ	E	非円唇前舌半広母音	Keller	E
ɛ̃	E~	非円唇前舌半広鼻母音	Terrain	E
インスタンス:	インスタンス:	非円唇前舌狭長母音	Lied	i
ɪ	I	非円唇前舌め広めの狭母音	bitte	i

IPA	X-SAMPA	説明	例	ビゼーム
オ:	オ:	円唇後舌半狭長母音	Kohl	o
ɔ	O	円唇後舌半広母音	Koffer	O
ɔ̃	O~	円唇後舌半広鼻母音	Annonce	O
ɔʏ	OY	二重母音	neu	O
u:	u:	円唇後舌狭長母音	Bruder	u
ʊ	U	円唇後舌め広めの狭母音	Wunder	u
y:	y:	円唇前舌狭長母音	kühl	u
ʏ	Y	円唇前舌め広めの狭母音	Küche	u
その他の記号				
ˈ	"	第一アクセント	Alabama	
ˌ	%	第二アクセント	Alabama	
.	.	音節境界	A.la.ba.ma	

ヒンディー語 (hi-IN)

以下の表では、国際音声記号 (IPA) の音素および拡張 SAM 音声記号 (X-SAMPA) の記号の完全なリスト、および Amazon Polly でサポートされるヒンディー語音声の音素のサウンドタイプについて説明しています。

ヒンディー語と組み合わせて使用される音素の詳細については、「英語 (インド) (en-IN)」を参照してください。

音素/ビゼームテーブル

IPA	X-SAMPA	説明	例
子音			
pʰ	p_h	無声有気両唇破裂音	फूल (phool)
bʱ	b_h	有声有気両唇破裂音	भारी (bhaari)
t̪	t_d	無声口腔内破裂音	तापमान (taapmaan)
t̪ʰ	t_d_h	無声有気口腔内破裂音	थोड़ा (thoda)
d̪	d_d	有声口腔内破裂音	दल्ली (dilli)
d̪ʱ	d_d_h	有声有気口腔内破裂音	धोबी (dhobi)
ʈ	t`	無声そり舌破裂音	कटोरा (katora)
ʈʰ	t`_h	無声有気そり舌破裂音	ठंड (thand)
ɖ	d`	有声そり舌破裂音	डर (darr)
ɖʱ	d`_h	有声有気そり舌破裂音	ढाल (dhal)
tʃʰ	tS_h	無声有気硬口蓋破擦音	छाल (chaal)
dʒʱ	dZ_h	有声有気硬口蓋破擦音	झाल (jhaal)
kʰ	k_h	無声有気軟口蓋破裂音	खान (khan)
gʱ	g_h	有声有気軟口蓋破裂音	घान (ghaan)
ɳ	n`	そり舌鼻音	क्षण (kshan)
ɾ	4	歯茎はじき音	राम (ram)
ɽ	r`	側音そり舌はじき音	बड़ा (bada)
ɽʱ	r'_h	有声有気そり舌はじき音	बढ़ी (barhi)
ʋ	v\	両唇接近音	वसूल (wasool)

IPA	X-SAMPA	説明	例
母音			
ə	@_o	中段中舌	अच्छा (achhaa)
ə̃	@~	鼻音化中段中舌母音	हँसना (hansnaa)
a	A_o	非円唇前舌広母音	आग (aag)
ã	A~	鼻音化非円唇前舌広母音	घ़डयिाँ (ghariyaan)
ɪ	I_o	非円唇前舌め広めの狭母音	इक्कीस (ikkees)
ɪ̃	I~	鼻音化非円唇前舌め広めの狭母音	संचिाई (sinchai)
i	i_o	非円唇前舌狭母音	बल्लिी (billee)
インスタンス	インスタンス~	鼻音化非円唇前舌狭母音	नही (nahin)
ʊ	U_o	円唇後舌め広めの狭母音	उल्लू (ullu)
ʊ̃	U~	鼻音化円唇後舌め広めの狭母音	मुँह (munh)
u	u_o	円唇後舌狭母音	फूल (phool)
ũ	u~	鼻音化円唇後舌狭母音	ऊँट (oont)
ɔ	O_o	円唇後舌半広母音	कौन (kaun)
ɔ̃	O~	鼻音化円唇後舌半広母音	भौ (bhaun)
o	o	円唇後舌半狭母音	सोना (sona)
オ	オ~	鼻音化円唇後舌半狭母音	क्यो (kyon)
ɛ	E_o	非円唇前舌半広母音	पैसा (paisa)

IPA	X-SAMPA	説明	例
ɛ̃	E~	鼻音化非円唇前舌半広母音	मैं (main)
e	e	非円唇前舌半狭母音	एक (ek)
ẽ	e~	鼻音化非円唇前舌半狭母音	किताबें (kitabein)

アイスランド語 (is-IS)

以下の表では、国際音声記号 (IPA) の音素および拡張 SAM 音声記号 (X-SAMPA) の記号の完全なリスト、および Amazon Polly でサポートされるアイスランド語音声の対応するビゼームについて説明しています。

音素/ビゼームテーブル

IPA	X-SAMPA	説明	例	ビゼーム
子音				
b	b	有声両唇破裂音	grasbakkanum	0
c	c	無声硬口蓋破裂音	pakkin	k
cʰ	c_h	有気無声硬口蓋破裂音	anarkistai	k
ç	C	無声硬口蓋摩擦音	héðan	k
d	d	有声歯茎破裂音	bóndi	t
ð	D	有声歯摩擦音	borð	T
f	f	無声唇歯摩擦音	duft	f
Amazon EC2 のセキュ	Amazon EC2 のセキュリティグループ	有声軟口蓋破裂音	holgóma	k

IPA	X-SAMPA	説明	例	ビゼーム
リティグループ				
ɣ	G	有声軟口蓋摩擦音	hugur	k
h	h	無声声門摩擦音	heili	k
j	j	硬口蓋接近音	jökull	i
kʰ	k_h	有気無声軟口蓋破裂音	ósköpunum	k
l	l	歯茎側面接近音	gólf	t
l̥	l_0	無声歯茎側面接近音	fólk	t
m	m	両唇鼻音	september	p
m̥	m_0	無声両唇鼻音	kompa	p
n	n	歯茎鼻音	númer	t
n̥	n_0	無声歯茎鼻音	pöntun	t
ɲ	J	硬口蓋鼻音	pælingar	J
ŋ	N	軟口蓋鼻音	söngvarann	k
ŋ̊	N_0	無声軟口蓋鼻音	frænka	k
pʰ	p_h	有気無声両唇破裂音	afplánun	p
r	r	歯茎ふるえ音	afskrifta	r
r̥	r_0	無声歯茎ふるえ音	andvörpum	r
s	s	無声歯茎摩擦音	baðhús	s

IPA	X-SAMPA	説明	例	ビゼーム
t^h	t_h	有気無声歯茎破裂音	tanki	t
θ	T	無声歯摩擦音	þeldökki	T
v	v	有声唇歯摩擦音	silfur	f
w	w	有声両唇軟口蓋接近音		u
x	x	無声軟口蓋摩擦音	samfélags	k
母音				
œ	9	円唇前舌半広母音	þröskuldinum	O
œ:	9:	円唇前舌半広長母音	tvö	O
a	a	非円唇前舌広母音	nefna	a
a:	a:	非円唇前舌広長母音	fara	a
au	au	二重母音	átta	a
au:	au:	二重母音	átján	a
ɛ	E	非円唇前舌半広母音	kennari	E
ɛ:	E:	非円唇前舌半広長母音	dreka	E
i	i	非円唇前舌狭母音	Gúlíver	i
インスタンス:	インスタンス:	非円唇前舌狭長母音	þrír	i

IPA	X-SAMPA	説明	例	ビゼーム
ɪ	I	非円唇前舌め広めの狭母音	samspil	i
ɪː	I:	非円唇前舌め広めの狭長母音	stig	i
ɔ	O	円唇後舌半広母音	regndropar	O
ɔː	O:	円唇後舌半広長母音	ullarbolur	O
ɔu	Ou	二重母音	tólf	O
ɔuː	Ou:	二重母音	fjórir	O
u	u	円唇後舌狭母音	stúlkan	u
uː	u:	円唇後舌狭長母音	frú	u
ʏ	Y	円唇前舌め広めの狭母音	tíu	u
ʏː	Y	円唇前舌め広めの狭長母音	gruninn	u
その他の記号				
ˈ	"	第一アクセント	Alabama	
ˌ	%	第二アクセント	Alabama	
.	.	音節境界	A.la.ba.ma	

イタリア語 (it-IT)

以下の表では、国際音声記号 (IPA) の音素および拡張 SAM 音声記号 (X-SAMPA) の記号の完全なリスト、および Amazon Polly でサポートされるイタリア語音声の対応するビゼームについて説明しています。

音素/ビゼームテーブル

IPA	X-SAMPA	説明	例	ビゼーム
子音				
b	b	有声両唇破裂音	bacca	p
d	d	有声歯茎破裂音	dama	t
͡dz	dz	有声歯茎破擦音	zero	s
͡dʒ	dZ	有声後部歯茎破擦音	giro	S
f	f	無声唇歯摩擦音	famiglia	f
Amazon EC2 のセキュリティグループ	Amazon EC2 のセキュリティグループ	有声軟口蓋破裂音	gatto	k
h	h	無声声門摩擦音	horror	k
j	j	硬口蓋接近音	dieci	i
k	k	無声軟口蓋破裂音	campo	k
l	l	歯茎側面接近音	lido	t
ʎ	L	硬口蓋側面接近音	aglio	J
m	m	両唇鼻音	mille	p
n	n	歯茎鼻音	nove	t
ɲ	J	硬口蓋鼻音	lasagne	J
p	p	無声両唇破裂音	pizza	p

IPA	X-SAMPA	説明	例	ビゼーム
r	r	歯茎ふるえ音	risata	r
s	s	無声歯茎摩擦音	sei	s
ʃ	S	無声後部歯茎摩擦音	scienza	S
t	t	無声歯茎破裂音	tavola	t
͡ts	ts	無声歯茎破擦音	forza	s
͡tʃ	tS	無声後部歯茎破擦音	cielo	S
v	v	有声唇歯摩擦音	venti	f
w	w	有声両唇軟口蓋接近音	quattro	u
z	z	有声歯茎摩擦音	bisogno	s
ʒ	Z	有声後部歯茎摩擦音	bijou	S
母音				
a	a	非円唇前舌広母音	arco	a
e	e	非円唇前舌半狭母音	tre	e
ɛ	E	非円唇前舌半広母音	ettaro	E
i	i	非円唇前舌狭母音	impero	i
o	o	円唇後舌半狭母音	cento	o
ɔ	O	円唇後舌半広母音	otto	O

IPA	X-SAMPA	説明	例	ビゼーム
u	u	円唇後舌狭母音	uno	u
その他の記号				
ˈ	"	第一アクセント	Alabama	
ˌ	%	第二アクセント	Alabama	
.	.	音節境界	A.la.ba.ma	

日本語 (ja-JP)

Amazon Pollyでは、日本語での発音仮名と読み仮名のアルファベットの使用をサポートしています。Amazon Polly で発音記号とこれらのアルファベットを使用するには、音素 `alphabet="x-amazon-`*`phonetic standard used`*`"` 属性を使用します。

- `x-amazon-pron-kana` — 発音仮名が使用されていることを示します。発音仮名は発音表記に使用される特殊なカタカナ文字で、ピッチアクセントをエンコードできます。
- `x-amazon-yomigana` — 読み仮名が使用されていることを示します。読み仮名には、従来のカタカナ、ひらがな、ヘボン式ローマ字の意味で使われているラテンアルファベットを使用できます。

次の例は、これらをを使用する方法を示しています。

発音仮名

```
<speak>
     ###<phoneme alphabet="x-amazon-pron-kana" ph="###'#">##</phoneme>###
</speak>
```

読み仮名

```
<speak>
     ###<phoneme alphabet="x-amazon-yomigana" ph="####">##</phoneme>###
     ###<phoneme alphabet="x-amazon-yomigana" ph="####">##</phoneme>###
     ###<phoneme alphabet="x-amazon-yomigana" ph="Hirokazu">##</phoneme>###
</speak>
```

以下の表では、国際音声記号 (IPA) の音素および拡張 SAM 音声記号 (X-SAMPA) の記号の完全なリスト、および Amazon Polly でサポートされる日本語音声の対応するビゼームについて説明しています。

IPA	X-SAMPA	説明	例	ビゼーム
子音				
ɾ	4	歯茎はじき音	練習、renshuu	t
ʔ	?	声門破裂音	あつっ、atsu'	
b	b	有声両唇破裂音	舞踊、buyou	p
β	B	有声両唇摩擦音	ヴィンテージ、vinteeji	B
c	c	無声硬口蓋破裂音	ききょう、kikyou	k
ç	C	無声硬口蓋摩擦音	人、hito	k
d	d	有声歯茎破裂音	濁点、dakuten	t
ʧ	dz\	有声歯茎硬口蓋破擦音	純、jun	J
g	Amazon EC2 のセキュリティグループ	有声軟口蓋破裂音	ご飯、gohan	k
h	h	無声声門摩擦音	本、hon	k
j	j	硬口蓋接近音	屋根、yane	i
ɟ	J\	有声硬口蓋破裂音	行儀、gyougi	J
k	k	無声軟口蓋破裂音	漢字、kanji	k
ɺ	l\	歯茎側面はじき音	釣り、tsuri	r

IPA	X-SAMPA	説明	例	ビゼーム
ɺj	l\j	歯茎側面はじき音、硬口蓋接近音	流行、ryuukou	r
m	m	両唇鼻音	飯、meshi	p
n	n	歯茎鼻音	猫、neko	t
ɲ	J	硬口蓋鼻音	日本、nippon	J
ɴ	N\	口蓋垂鼻音	缶、kan	k
p	p	無声両唇破裂音	パン、pan	p
ɸ	p\	無声両唇摩擦音	福、huku	f
s	s	無声歯茎摩擦音	層、sou	s
ɕ	s\	無声歯茎硬口蓋摩擦音	書簡、shokan	J
t	t	無声歯茎破裂音	手紙、tegami	t
t͡s	ts	無声歯茎破擦音	釣り、tsuri	s
t͡ɕ	ts\	無声歯茎硬口蓋破擦音	吉、kichi	J
w	w	有声両唇軟口蓋接近音	電話、denwa	u
z	z	有声歯茎摩擦音	座敷、zashiki	s
母音				
äː	a:_"	非円唇中舌広長母音	羽蟻、haari	a
ä	a_"	非円唇中舌広母音	仮名、kana	a

IPA	X-SAMPA	説明	例	ビゼーム
eː	e:_o	非円唇前舌中央長母音	学生、gakusei	@
e	e_o	非円唇前舌中央母音	歴、reki	@
i	i	非円唇前舌狭母音	気、ki	i
iː	インスタンス:	非円唇前舌狭長母音	詩歌、shiika	i
ɯ	M	非円唇後舌狭母音	運、un	i
ɯː	M:	非円唇後舌狭長母音	宗教、shuukyou	i
oː	o:_o	円唇後舌中央長母音	購読、koodoku	o
o	o_o	円唇後舌中央母音	読者、dokusha	o

韓国語 (ko-KR)

以下の表では、国際音声記号 (IPA) の音素および拡張 SAM 音声記号 (X-SAMPA) の記号の完全なリスト、および Amazon Polly でサポートされる韓国語音声の対応するビゼームについて説明しています。

IPA	X-SAMPA	説明	例	ビゼーム
子音				
k	k	無声軟口蓋破裂音	강、[g]ang	k
k#	k_t	強力無声軟口蓋破裂音	깨、[kk]e	k
n	n	歯茎鼻音	남、[n]am	t

IPA	X-SAMPA	説明	例	ビゼーム
t	t	無声歯茎破裂音	도、[d]o	t
t#	t_t	強力無声歯茎破裂音	때、[tt]e	t
ɾ	4	歯茎はじき音	사랑、sa[r]ang	t
l	l	歯茎側面接近音	돌、do[l]	t
m	m	両唇鼻音	무、[m]u	p
p	p	無声両唇破裂音	봄、[b]om	p
p#	p_t	強力無声両唇破裂音	뻘、[pp]eol	p
s	s	無声歯茎摩擦音	새、[s]e	s
s#	s_t	強力無声歯茎摩擦音	씨、[ss]i	s
ŋ	N	軟口蓋鼻音	방、ba[ng]	k
t͡ɕ	ts\	無声歯茎硬口蓋破擦音	조、[j]o	J
t͡ɕ#	ts_t	強力無声歯茎硬口蓋破擦音	찌、[jj]i	J
t͡ɕʰ	ts_h	有気無声歯茎硬口蓋破擦音	차、[ch]a	J
kʰ	k_h	有気無声軟口蓋破裂音	코、[k]o	k
tʰ	t_h	有気無声歯茎破裂音	통、[t]ong	t

IPA	X-SAMPA	説明	例	ビゼーム
p^h	p_h	有気無声両唇破裂音	패、[p]e	p
h	h	無声声門摩擦音	힘、[h]im	k
j	j	硬口蓋接近音	양、[y]ang	i
w	w	有声両唇軟口蓋接近音	왕、[w]ang	u
ɯ̟	M\	軟口接近音 >	의、[wj]i	i
母音				
a	a	非円唇前舌広母音	밥、b[a]b	a
ʌ	V	非円唇後舌半広母音	정、j[eo]ng	E
ɛ	E	非円唇前舌半広母音	배、b[e]	E
o	o	円唇後舌半狭母音	노、n[o]	o
u	u	円唇後舌狭母音	둘、d[u]l	u
ɯ	M	非円唇後舌狭母音	은、[eu]n	i
i	i	非円唇前舌狭母音	김、k[i]m	i

ノルウェー語 (nb-NO)

以下の図では、国際音声記号 (IPA) の音素および拡張 SAM 音声記号 (X-SAMPA) の記号の完全なリスト、および Amazon Polly でサポートされるノルウェー語音声の対応するビゼームについて説明しています。

IPA	X-SAMPA	説明	例	ビゼーム
子音				
ɾ	4	歯茎はじき音	prøv	t
b	b	有声両唇破裂音	labb	p
ç	C	無声硬口蓋摩擦音	kino	k
d	d	有声歯茎破裂音	ladd	t
ɖ	d`	有声そり舌破裂音	verdi	t
f	f	無声唇歯摩擦音	fot	f
g	g	有声軟口蓋破裂音	tagg	k
h	h	無声声門摩擦音	ha	k
j	j	硬口蓋接近音	gi	i
k	k	無声軟口蓋破裂音	takk	k
l	l	歯茎側面接近音	fall、ball	t
ɭ	l`	そり舌側面接近音	ærlig	t
m	m	両唇鼻音	lam	p
n	n	歯茎鼻音	vann	t
ɳ	n`	そり舌鼻音	garn	t
ŋ	N	軟口蓋鼻音	sang	k
p	p	無声両唇破裂音	hopp	p
s	s	無声歯茎摩擦音	lass	s
ʂ	s`	無声そり舌摩擦音	års	S

IPA	X-SAMPA	説明	例	ビゼーム
ʃ	S	無声後部歯茎摩擦音	skyt	S
t	t	無声歯茎破裂音	lat	t
ʈ	t`	無声そり舌破裂音	hardt	t
ʊ	v\	唇歯接近音	vin	f
w	w	有声両唇軟口蓋接近音	will	x
母音				
øː	2:	円唇前舌半狭長母音	søt	o
œ	9	円唇前舌半広母音	søtt	O
ə	@	中段中舌	ape	@
æː	{:	非円唇前舌狭めの広長母音	vær	a
ʉ	}	円唇中舌狭母音	lund	u
ʉː	}:	円唇中舌狭長母音	lun	u
æ	{	非円唇前舌狭めの広母音	vært	a
ɑ	A	非円唇後舌広母音	hatt	a
ɑː	A:	非円唇後舌広長母音	hat	a
eː	e:	非円唇前舌半狭長母音	sen	e

IPA	X-SAMPA	説明	例	ビゼーム
ɛ	E	非円唇前舌半広母音	send	E
インスタンス:	インスタンス:	非円唇前舌狭長母音	vin	i
ɪ	I	非円唇前舌め広めの狭母音	vind	i
oː	oː	円唇後舌半狭長母音	våt	o
ɔ	O	円唇後舌半広母音	vått	O
uː	uː	円唇後舌狭長母音	bok	u
ʊ	U	円唇後舌め広めの狭母音	bukk	u
yː	yː	円唇前舌狭長母音	lyn	u
ʏ	Y	円唇前舌め広めの狭母音	lynne	u
その他の記号				
ˈ	"	第一アクセント	Alabama	
ˌ	%	第二アクセント	Alabama	
.	.	音節境界	A.la.ba.ma	

ポーランド語 (pl-PL)

以下の表では、国際音声記号 (IPA) の音素および拡張 SAM 音声記号 (X-SAMPA) の記号の完全なリスト、および Amazon Polly でサポートされるポーランド語音声の対応するビゼームについて説明しています。

音素/ビゼームテーブル

IPA	X-SAMPA	説明	例	ビゼーム
子音				
b	b	有声両唇破裂音	bobas、belka	p
d	d	有声歯茎破裂音	dar、do	t
ʣ	dz	有声歯茎破擦音	dzwon、widzowie	s
ʥ	dz\	有声歯茎硬口蓋破擦音	dźwięk	J
ʤ	dz`	有声そり舌破擦音	dżem、dżungla	S
f	f	無声唇歯摩擦音	furtka、film	f
Amazon EC2 のセキュリティグループ	Amazon EC2 のセキュリティグループ	有声軟口蓋破裂音	gazeta、waga	k
h	h	無声声門摩擦音	chleb、handel	k
j	j	硬口蓋接近音	jak、maja	i
k	k	無声軟口蓋破裂音	kura、marek	k
l	l	歯茎側面接近音	lipa、alicja	t
m	m	両唇鼻音	matka、molo	p
n	n	歯茎鼻音	norka	t
ɲ	J	硬口蓋鼻音	koń、toruń	J
p	p	無声両唇破裂音	pora、stop	p

IPA	X-SAMPA	説明	例	ビゼーム
r	r	歯茎ふるえ音	rok、park	r
s	s	無声歯茎摩擦音	sum、pas	s
ɕ	s\	無声歯茎硬口蓋摩擦音	śruba、śnieg	J
ʂ	s`	無声そり舌摩擦音	szum、masz	S
t	t	無声歯茎破裂音	tok、stół	t
͡ts	ts	無声歯茎破擦音	car、co	s
͡tɕ	ts\	無声歯茎硬口蓋破擦音	ćma、mieć	J
͡tʂ	ts`	無声そり舌破擦音	czas、raczej	S
v	v	有声唇歯摩擦音	worek、mewa	f
w	w	有声両唇軟口蓋接近音	łaska、mało	u
z	z	有声歯茎摩擦音	zero	s
ʑ	z\	有声歯茎硬口蓋摩擦音	źrebię、bieliźnie	J
ʐ	z`	有声そり舌摩擦音	żar、żona	S
母音				
a	a	非円唇前舌広母音	ja	a
ɛ	E	非円唇前舌半広母音	echo	E
ɛ̃	E~	非円唇前舌半広鼻母音	węże	E

IPA	X-SAMPA	説明	例	ビゼーム
i	i	非円唇前舌狭母音	ile	i
ɔ	O	円唇後舌半広母音	oczy	O
ɔ̃	O~	円唇後舌半広鼻母音	wąż	O
u	u	円唇後舌狭母音	uczta	u
ɨ	1	非円唇中舌狭母音	byk	i
その他の記号				
ˈ	"	第一アクセント	Alabama	
ˌ	%	第二アクセント	Alabama	
.	.	音節境界	A.la.ba.ma	

ポルトガル語 (pt-PT)

以下の表では、国際音声記号 (IPA) の音素および拡張 SAM 音声記号 (X-SAMPA) の記号の完全なリスト、および Amazon Polly でサポートされるポルトガル語音声の対応するビゼームについて説明しています。

音素/ビゼームテーブル

IPA	X-SAMPA	説明	例	ビゼーム
子音				
ɾ	4	歯茎はじき音	pira	t
b	b	有声両唇破裂音	dato	p
d	d	有声歯茎破裂音	dato	t
f	f	無声唇歯摩擦音	facto	f

IPA	X-SAMPA	説明	例	ビゼーム
Amazon EC2 のセキュリティグループ	Amazon EC2 のセキュリティグループ	有声軟口蓋破裂音	gato	k
j	j	硬口蓋接近音	paraguay	i
k	k	無声軟口蓋破裂音	cacto	k
l	l	歯茎側面接近音	galo	t
ʎ	L	硬口蓋側面接近音	galho	J
m	m	両唇鼻音	mato	p
n	n	歯茎鼻音	nato	t
ɲ	J	硬口蓋鼻音	pinha	J
p	p	無声両唇破裂音	pato	p
ʁ	R\	口蓋垂ふるえ音	barroso	k
s	s	無声歯茎摩擦音	saca	s
ʃ	S	無声後部歯茎摩擦音	chato	S
t	t	無声歯茎破裂音	tacto	t
v	v	有声唇歯摩擦音	vaca	f
w	w	有声両唇軟口蓋接近音	mau	u
z	z	有声歯茎摩擦音	zaca	s

IPA	X-SAMPA	説明	例	ビゼーム
ʒ	Z	有声後部歯茎摩擦音	jacto	S
母音				
a	a	非円唇前舌広母音	parto	a
ã	a~	非円唇前舌広鼻母音	pega	a
e	e	非円唇前舌半狭母音	pega	e
ẽ	e~	非円唇前舌半狭鼻母音	movem	e
ɛ	E	非円唇前舌半広母音	café	E
i	i	非円唇前舌狭母音	lingueta	i
インスタンズ	インスタンス~	非円唇前舌狭鼻母音	cinto	i
o	o	円唇後舌半狭母音	poder	o
オ	オ~	円唇後舌半狭鼻母音	compra	o
ɔ	O	円唇後舌半広母音	cotó	O
u	u	円唇後舌狭母音	fui	u
ũ	u~	円唇後舌狭鼻母音	sunto	u
その他の記号				
ˈ	"	第一アクセント	Alabama	

IPA	X-SAMPA	説明	例	ビゼーム
ˌ	%	第二アクセント	Alabama	
.	.	音節境界	A.la.ba.ma	

ポルトガル語 (ブラジル) (pt-BR)

以下の表では、国際音声記号 (IPA) の音素および拡張 SAM 音声記号 (X-SAMPA) の記号の完全なリスト、および Amazon Polly でサポートされるポルトガル語 (ブラジル) 音声の対応するビゼームについて説明しています。

音素/ビゼームテーブル

IPA	X-SAMPA	説明	例	ビゼーム
子音				
ɾ	4	歯茎はじき音	pira	t
b	b	有声両唇破裂音	bato	p
d	d	有声歯茎破裂音	dato	t
ʧ	dZ	有声後部歯茎破擦音	idade	S
f	f	無声唇歯摩擦音	facto	f
Amazon EC2 のセキュリティグループ	Amazon EC2 のセキュリティグループ	有声軟口蓋破裂音	gato	k
j	j	硬口蓋接近音	paraguay	i
k	k	無声軟口蓋破裂音	cacto	k

IPA	X-SAMPA	説明	例	ビゼーム
l	l	歯茎側面接近音	galo	t
ʎ	L	硬口蓋側面接近音	galho	J
m	m	両唇鼻音	mato	p
n	n	歯茎鼻音	nato	t
ɲ	J	硬口蓋鼻音	pinha	J
p	p	無声両唇破裂音	pato	p
s	s	無声歯茎摩擦音	saca	s
ʃ	S	無声後部歯茎摩擦音	chato	S
t	t	無声歯茎破裂音	tacto	t
⁀tʃ	tS	無声後部歯茎破擦音	noite	S
v	v	有声唇歯摩擦音	vaca	f
w	w	有声両唇軟口蓋接近音	mau	u
χ	X	無声口蓋垂摩擦音	carro	k
z	z	有声歯茎摩擦音	zaca	s
ʒ	Z	有声後部歯茎摩擦音	jacto	S
母音				
a	a	非円唇前舌広母音	parto	a

IPA	X-SAMPA	説明	例	ビゼーム
ã	a~	非円唇前舌広鼻母音	pensamos	a
e	e	非円唇前舌半狭母音	pega	e
ẽ	e~	非円唇前舌半狭鼻母音	movem	e
ɛ	E	非円唇前舌半広母音	café	E
i	i	非円唇前舌狭母音	lingueta	i
インスタンズ	インスタンス~	非円唇前舌狭鼻母音	cinto	i
o	o	円唇後舌半狭母音	poder	o
オ	オ~	円唇後舌半狭鼻母音	compra	o
ɔ	O	円唇後舌半広母音	cotó	O
u	u	円唇後舌狭母音	fui	u
ũ	u~	円唇後舌狭鼻母音	sunto	u
その他の記号				
ˈ	"	第一アクセント	Alabama	
ˌ	%	第二アクセント	Alabama	
.	.	音節境界	A.la.ba.ma	

ルーマニア語 (ro-RO)

以下の表では、国際音声記号 (IPA) の音素および拡張 SAM 音声記号 (X-SAMPA) の記号の完全なリスト、および Amazon Polly でサポートされるルーマニア語音声の対応するビゼームについて説明しています。

音素/ビゼームテーブル

IPA	X-SAMPA	説明	例	ビゼーム
子音				
b	b	有声両唇破裂音	bubă	p
d	d	有声歯茎破裂音	după	t
ʧ	dZ	有声後部歯茎破擦音	george	S
f	f	無声唇歯摩擦音	afacere	f
Amazon EC2 のセキュリティグループ	Amazon EC2 のセキュリティグループ	有声軟口蓋破裂音	agri#	k
h	h	無声声門摩擦音	harpă	k
j	j	硬口蓋接近音	baie	i
k	k	無声軟口蓋破裂音	co#	k
l	l	歯茎側面接近音	lampa	t
m	m	両唇鼻音	mama	p
n	n	歯茎鼻音	nor	t
p	p	無声両唇破裂音	pilă	p

IPA	X-SAMPA	説明	例	ビゼーム
r	r	歯茎ふるえ音	rampă	r
s	s	無声歯茎摩擦音	soare	s
ʃ	S	無声後部歯茎摩擦音	ma#ină	S
t	t	無声歯茎破裂音	tata	t
t͡s	ts	無声歯茎破擦音	#ară	s
t͡ʃ	tS	無声後部歯茎破擦音	ceai	S
v	v	有声唇歯摩擦音	via#ă	f
w	w	有声両唇軟口蓋接近音	beau	u
z	z	有声歯茎摩擦音	mozol	s
ʒ	Z	有声後部歯茎摩擦音	joacă	S
母音				
ə	@	中段中舌	babă	@
a	a	非円唇前舌広母音	casa	a
e	e	非円唇前舌半狭母音	elan	e
e̯	e_^	非音節非円唇前舌半狭母音	beau	e
i	i	非円唇前舌狭母音	mie	i
o	o	円唇後舌半狭母音	oră	o

IPA	X-SAMPA	説明	例	ビゼーム
oa	o_^a	二重母音	oare	o
u	u	円唇後舌狭母音	unde	u
ɨ	1	非円唇中舌狭母音	România	i
その他の記号				
ˈ	"	第一アクセント	Alabama	
ˌ	%	第二アクセント	Alabama	
.	.	音節境界	A.la.ba.ma	

ロシア語 (ru-RU)

以下の表では、国際音声記号 (IPA) の音素および拡張 SAM 音声記号 (X-SAMPA) の記号の完全なリスト、および Amazon Polly でサポートされるロシア語音声の対応するビゼームについて説明しています。

音素/ビゼームテーブル

IPA	X-SAMPA	説明	例	ビゼーム
子音				
b	b	有声両唇破裂音	борт	p
bʲ	b'	硬口蓋化有声両唇破裂音	бюро	p
d	d	有声歯茎破裂音	дом	t
dʲ	d'	硬口蓋化有声歯茎破裂音	дядя	t
f	f	無声唇歯摩擦音	флаг	f

IPA	X-SAMPA	説明	例	ビゼーム
f^j	f'	硬口蓋化無声唇歯摩擦音	февраль	f
Amazon EC2 のセキュリティグループ	Amazon EC2 のセキュリティグループ	有声軟口蓋破裂音	нога	k
g^j	Amazon EC2 のセキュリティグループ'	硬口蓋化有声軟口蓋破裂音	герой	k
j	j	硬口蓋接近音	дизайн、ящик	i
k	k	無声軟口蓋破裂音	кот	k
k^j	k'	硬口蓋化無声軟口蓋破裂音	кино	k
l	l	歯茎側面接近音	лампа	t
l^j	l'	硬口蓋化歯茎側面接近音	лес	t
m	m	両唇鼻音	мама	p
m^j	m'	硬口蓋化両唇鼻音	мяч	p
n	n	歯茎鼻音	нос	t
n^j	n'	硬口蓋化歯茎鼻音	няня	t
p	p	無声両唇破裂音	папа	p
p^j	p'	硬口蓋化無声両唇破裂音	перо	p

IPA	X-SAMPA	説明	例	ビゼーム
r	r	歯茎ふるえ音	роза	r
rʲ	r'	硬口蓋化歯茎ふるえ音	рюмка	r
s	s	無声歯茎摩擦音	сыр	s
sʲ	s'	硬口蓋化無声歯茎摩擦音	сердце、русь	s
ɕː	s\:	長い無声歯茎硬口蓋摩擦音	щека	J
ʂ	s`	無声そり舌摩擦音	шум	S
t	t	無声歯茎破裂音	точка	t
tʲ	t'	硬口蓋化無声歯茎破裂音	тётя	t
t͡s	ts	無声歯茎破擦音	царь	s
t͡ɕ	ts\	無声歯茎硬口蓋破擦音	час	J
v	v	有声唇歯摩擦音	вор	f
vʲ	v'	硬口蓋化有声唇歯摩擦音	верфь	f
x	x	無声軟口蓋摩擦音	хор	k
xʲ	x'	硬口蓋化無声軟口蓋摩擦音	химия	k
z	z	有声歯茎摩擦音	зуб	s
zʲ	z'	硬口蓋化有声歯茎摩擦音	зима	s

IPA	X-SAMPA	説明	例	ビゼーム
ʑː	z\:	長い有声歯茎硬口蓋摩擦音	уезжать	J
ʐ	z`	有声そり舌摩擦音	жена	S
母音				
ə	@	中段中舌	канарейка	@
a	a	非円唇前舌広母音	два、яблоко	a
e	e	非円唇前舌半狭母音	печь	e
ɛ	E	非円唇前舌半広母音	это	E
i	i	非円唇前舌狭母音	один、четыре	i
o	o	円唇後舌半狭母音	кот	o
u	u	円唇後舌狭母音	муж、вьюга	u
ɨ	1	非円唇中舌狭母音	мышь	i

スペイン語 (es-ES)

以下の表では、国際音声記号 (IPA) の音素および拡張 SAM 音声記号 (X-SAMPA) の記号の完全なリスト、および Amazon Polly でサポートされるスペイン語音声の対応するビゼームについて説明しています。

音素/ビゼームテーブル

IPA	X-SAMPA	説明	例	ビゼーム
子音				
ɾ	4	歯茎はじき音	pero、bravo、amor、	t

IPA	X-SAMPA	説明	例	ビゼーム
b	b	有声両唇破裂音	bestia	p
β	B	有声両唇摩擦音	bebé	B
d	d	有声歯茎破裂音	cuando	t
ð	D	有声歯摩擦音	arder	T
f	f	無声唇歯摩擦音	fase、café	f
Amazon EC2 のセキュリティグループ	Amazon EC2 のセキュリティグループ	有声軟口蓋破裂音	gato、lengua、guerr	k
ɣ	G	有声軟口蓋摩擦音	trigo、Argos	k
j	j	硬口蓋接近音	hacia、tierra、radio、	i
ʝ	j\	有声硬口蓋摩擦音	enhielar、sayo、inye syerba	J
k	k	無声軟口蓋破裂音	caña、laca、quisimo	k
l	l	歯茎側面接近音	lino、calor、princi pal	t
ʎ	L	硬口蓋側面接近音	llave、pollo	J
m	m	両唇鼻音	madre、comer、anfil	p
n	n	歯茎鼻音	nido、anillo、sin	t
ɲ	J	硬口蓋鼻音	cabaña、ñoquis	J
ŋ	N	軟口蓋鼻音	cinco、venga	k

IPA	X-SAMPA	説明	例	ビゼーム
p	p	無声両唇破裂音	pozo、topo	p
r	r	歯茎ふるえ音	perro、enrachado	r
s	s	無声歯茎摩擦音	saco、casa、puertas	s
t	t	無声歯茎破裂音	tamiz、átomo	t
ʧ	tS	無声後部歯茎破擦音	chubasco	S
θ	T	無声歯摩擦音	cereza、zorro、lacer	T
w	w	有声両唇軟口蓋接近音	fuego、fuimos、cuot	u
x	x	無声軟口蓋摩擦音	jamón、general、suje、reloj	k
z	z	有声歯茎摩擦音	rasgo、mismo	s
母音				
a	a	非円唇前舌広母音	tanque	a
e	e	非円唇前舌半狭母音	peso	e
i	i	非円唇前舌狭母音	cinco	i
o	o	円唇後舌半狭母音	bosque	o
u	u	非円唇前舌半狭母音	publicar	u
その他の記号				
ˈ	"	第一アクセント	Alabama	

IPA	X-SAMPA	説明	例	ビゼーム
ˌ	%	第二アクセント	Alabama	
.	.	音節境界	A.la.ba.ma	

スペイン語 (メキシコ) (es-MX)

以下の表では、国際音声記号 (IPA) の音素および拡張 SAM 音声記号 (X-SAMPA) の記号の完全なリスト、および Amazon Polly でサポートされるスペイン語 (メキシコ) 音声の対応するビゼームについて説明しています。

音素/ビゼームテーブル

IPA	X-SAMPA	説明	例	ビゼーム
子音				
ɾ	4	歯茎はじき音	pero、bravo、amor、	t
b	b	有声両唇破裂音	bestia	p
β	B	有声両唇摩擦音	bebé	B
d	d	有声歯茎破裂音	cuando	t
ð	D	有声歯摩擦音	arder	T
f	f	無声唇歯摩擦音	fase、café	f
Amazon EC2 のセキュリティグループ	Amazon EC2 のセキュリティグループ	有声軟口蓋破裂音	gato、lengua、guerr	k
ɣ	G	有声軟口蓋摩擦音	trigo、Argos	k
j	j	硬口蓋接近音	hacia、tierra、radio、	i

IPA	X-SAMPA	説明	例	ビゼーム
ʝ	j\	有声硬口蓋摩擦音	enhielar、sayo、inye syerba	J
k	k	無声軟口蓋破裂音	caña、laca、quisimo	k
l	l	歯茎側面接近音	lino、calor、princi pal	t
m	m	両唇鼻音	madre、comer、anfil	p
n	n	歯茎鼻音	nido、anillo、sin	t
ɲ	J	硬口蓋鼻音	cabaña、ñoquis	J
ŋ	N	軟口蓋鼻音	angosto、increíble	k
p	p	無声両唇破裂音	pozo、topo	p
r	r	歯茎ふるえ音	perro、enrachado	r
s	s	無声歯茎摩擦音	saco、casa、puertas	s
ʃ	S	無声後部歯茎摩擦音	show、flash	S
t	t	無声歯茎破裂音	tamiz、átomo	t
⁀tʃ	tS	無声後部歯茎破擦音	chubasco	S
w	w	有声両唇軟口蓋接近音	fuego、fuimos、cuot	u
x	x	無声軟口蓋摩擦音	jamón、general、pe aje、reloj	k
z	z	有声歯茎摩擦音	rasgo、mismo	s
母音				

IPA	X-SAMPA	説明	例	ビゼーム
a	a	非円唇中舌広母音	tanque	a
e	e	非円唇前舌半狭母音	peso	e
i	i	非円唇前舌狭母音	cinco	i
o	o	円唇後舌半狭母音	bosque	o
u	u	円唇後舌狭母音	publicar	u
その他の記号				
ˈ	"	第一アクセント	Alabama	
ˌ	%	第二アクセント	Alabama	
.	.	音節境界	A.la.ba.ma	

スペイン語 (米国) (es-US)

以下の表では、国際音声記号 (IPA) の音素および拡張 SAM 音声記号 (X-SAMPA) の記号の完全なリスト、および Amazon Polly でサポートされるスペイン語 (米国) 音声の対応するビゼームについて説明しています。

音素/ビゼームテーブル

IPA	X-SAMPA	説明	例	ビゼーム
子音				
ɾ	4	歯茎はじき音	pero、bravo、amor、	t
b	b	有声両唇破裂音	bestia	p
β	B	有声両唇摩擦音	bebé	B
d	d	有声歯茎破裂音	cuando	t

IPA	X-SAMPA	説明	例	ビゼーム
ð	D	有声歯摩擦音	arder	T
f	f	無声唇歯摩擦音	fase、café	f
Amazon EC2 のセキュリティグループ	Amazon EC2 のセキュリティグループ	有声軟口蓋破裂音	gato、lengua、guerr	k
ɣ	G	有声軟口蓋摩擦音	trigo、Argos	k
j	j	硬口蓋接近音	hacia、tierra、radio、	i
ʝ	j\	有声硬口蓋摩擦音	enhielar、sayo、inye syerba	J
k	k	無声軟口蓋破裂音	caña、laca、quisimo	k
l	l	歯茎側面接近音	lino、calor、princi pal	t
m	m	両唇鼻音	madre、comer、anfil	p
n	n	歯茎鼻音	nido、anillo、sin	t
ɲ	J	硬口蓋鼻音	cabaña、ñoquis	J
ŋ	N	軟口蓋鼻音	angosto、increíble	k
p	p	無声両唇破裂音	pozo、topo	p
r	r	歯茎ふるえ音	perro、enrachado	r
s	s	無声歯茎摩擦音	saco、casa、puertas	s
ʃ	S	無声後部歯茎摩擦音	show、flash	S

IPA	X-SAMPA	説明	例	ビゼーム
t	t	無声歯茎破裂音	tamiz、átomo	t
ʧ	tS	無声後部歯茎破擦音	chubasco	S
w	w	有声両唇軟口蓋接近音	fuego、fuimos、cuot	u
x	x	無声軟口蓋摩擦音	jamón、general、peaje、reloj	k
z	z	有声歯茎摩擦音	rasgo、mismo	s
母音				
a	a	非円唇中舌広母音	tanque	a
e	e	非円唇前舌半狭母音	peso	e
i	i	非円唇前舌狭母音	cinco	i
o	o	円唇後舌半狭母音	bosque	o
u	u	円唇後舌狭母音	publicar	u
その他の記号				
ˈ	"	第一アクセント	Alabama	
ˌ	%	第二アクセント	Alabama	
.	.	音節境界	A.la.ba.ma	

スウェーデン語 (sv-SE)

以下の表では、国際音声記号 (IPA) の音素および拡張 SAM 音声記号 (X-SAMPA) の記号の完全なリスト、および Amazon Polly でサポートされるスウェーデン語音声の対応するビゼームについて説明しています。

音素/ビゼームテーブル

IPA	X-SAMPA	説明	例	ビゼーム
子音				
b	b	有声両唇破裂音	bil	p
d	d	有声歯茎破裂音	dal	t
ɖ	d`	有声そり舌破裂音	bord	t
f	f	無声唇歯摩擦音	fil	f
Amazon EC2 のセキュリティグループ	Amazon EC2 のセキュリティグループ	有声軟口蓋破裂音	gås	k
h	h	無声声門摩擦音	hal	k
j	j	硬口蓋接近音	jag	i
k	k	無声軟口蓋破裂音	kal	k
l	l	歯茎側面接近音	lös	t
ɭ	l`	そり舌側面接近音	härlig	t
m	m	両唇鼻音	mil	p
n	n	歯茎鼻音	nålar	t

IPA	X-SAMPA	説明	例	ビゼーム
ɳ	n`	そり舌鼻音	barn	t
ŋ	N	軟口蓋鼻音	ring	k
p	p	無声両唇破裂音	pil	p
r	r	歯茎ふるえ音	ris	r
s	s	無声歯茎摩擦音	sil	s
ɕ	s\	無声歯茎硬口蓋摩擦音	tjock	J
ʂ	s`	無声そり舌摩擦音	fors、schlager	S
t	t	無声歯茎破裂音	tal	t
ʈ	t`	無声そり舌破裂音	hjort	t
v	v	有声唇歯摩擦音	vår	f
w	w	有声両唇軟口蓋接近音	aula、airways	u
ɧ	x\	無声後部歯茎軟口蓋摩擦音	sjuk	k
母音				
ø	2	円唇前舌半狭母音	föll、förr	o
ø	2:	円唇前舌半狭長母音	föl、nöt、för	o
ɵ	8	円唇中舌半狭母音	buss、full	o
ə	@	中段中舌	pojken	@
ʉː	}:	円唇中舌狭長母音	hus、ful	u

IPA	X-SAMPA	説明	例	ビゼーム
a	a	非円唇前舌広母音	hall、matt	a
æ	{	非円唇前舌狭めの広母音	herr	a
ɑ:	A:	非円唇後舌広長母音	hal、mat	a
e:	e:	非円唇前舌半狭長母音	vet、hel	e
ɛ	E	非円唇前舌半広母音	vett、rätt、hetta、hä	E
ɛ:	E:	非円唇前舌半広長母音	säl、häl、här	E:
インスタンス:	インスタンス:	非円唇前舌狭長母音	vit、sil	インスタンス:
ɪ	I	非円唇前舌め広めの狭母音	vitt、sill	i
オ:	オ:	円唇後舌半狭長母音	hål、mål	o
ɔ	O	円唇後舌半広母音	håll、moll	O
u:	u:	円唇後舌狭長母音	sol、bot	u
ʊ	U	円唇後舌め広めの狭母音	bott	u
y	y	円唇前舌狭母音	bytt	u
y:	y:	円唇前舌狭長母音	syl、syl	u
その他の記号				

IPA	X-SAMPA	説明	例	ビゼーム
ˈ	"	第一アクセント	Alabama	
ˌ	%	第二アクセント	Alabama	
.	.	音節境界	A.la.ba.ma	

トルコ語 (tr-TR)

以下の表では、国際音声記号 (IPA) の音素および拡張 SAM 音声記号 (X-SAMPA) の記号の完全なリスト、および Amazon Polly でサポートされるトルコ語音声の対応するビゼームについて説明しています。

音素/ビゼームテーブル

IPA	X-SAMPA	説明	例	ビゼーム
子音				
ɾ	4	歯茎はじき音	durum	t
ɾ̥	4_0_r	無声摩擦歯茎はじき音	bir	t
ɾ̞	4_r	摩擦歯茎はじき音	raf	t
b	b	有声両唇破裂音	raf	p
c	c	無声硬口蓋破裂音	kedi	k
d	d	有声歯茎破裂音	dede	t
d͡ʒ	dZ	有声後部歯茎破擦音	cam	S
f	f	無声唇歯摩擦音	fare	f

IPA	X-SAMPA	説明	例	ビゼーム
Amazon EC2 のセキュリティグループ	Amazon EC2 のセキュリティグループ	有声軟口蓋破裂音	galibi	k
h	h	無声声門摩擦音	hasta	k
j	j	硬口蓋接近音	yat	i
ɟ	J\	有声硬口蓋破裂音	genç	J
k	k	無声軟口蓋破裂音	akıl	k
l	l	歯茎側面接近音	lale	t
ɬ	5	軟口蓋歯茎側面接近音	labirent	t
m	m	両唇鼻音	maaş	p
n	n	歯茎鼻音	anı	t
p	p	無声両唇破裂音	ip	p
s	s	無声歯茎摩擦音	ses	s
ʃ	S	無声後部歯茎摩擦音	aşı	S
t	t	無声歯茎破裂音	ütü	t
‿tʃ	tS	無声後部歯茎破擦音	çaba	S
v	v	有声唇歯摩擦音	ekvator、kahveci、al	f
z	z	有声歯茎摩擦音	ver	s

IPA	X-SAMPA	説明	例	ビゼーム
ʒ	Z	有声後部歯茎摩擦音	azık	S
母音				
ø	2	円唇前舌半狭母音	göl	0
œ	9	円唇前舌半広母音	banliyö	O
a	a	非円唇前舌広母音	kal	a
a:	a:	非円唇前舌広長母音	davacı	a
æ	{	非円唇前舌狭めの広母音	özlem、güvenlik、gü	a
e	e	非円唇前舌半狭母音	keçi	e
ɛ	E	非円唇前舌半広母音	dede	E
i	i	非円唇前舌狭母音	bir	i
インスタンス:	インスタンス:	非円唇前舌狭長母音	izah	i
ɪ	I	非円唇前舌め広めの狭母音	keçi	i
ɯ	M	非円唇後舌狭母音	kıl	i
o	o	円唇後舌半狭母音	kol	o
オ:	オ:	円唇後舌半狭長母音	dolar	o
u	u	円唇後舌狭母音	durum	u

IPA	X-SAMPA	説明	例	ビゼーム
u:	u:	円唇後舌狭長母音	ruhum	u
ʊ	U	円唇後舌め広めの狭母音	dolu	u
y	y	円唇前舌狭母音	güvenlik	u
ʏ	Y	円唇前舌め広めの狭母音	aşı	u
その他の記号				
ˈ	"	第一アクセント	Alabama	
ˌ	%	第二アクセント	Alabama	
.	.	音節境界	A.la.ba.ma	

ウェールズ語 (cy-GB)

以下の表では、国際音声記号 (IPA) の音素および拡張 SAM 音声記号 (X-SAMPA) の記号の完全なリスト、および Amazon Polly でサポートされるウェールズ語音声の対応するビゼームについて説明しています。

音素/ビゼームテーブル

IPA	X-SAMPA	説明	例	ビゼーム
子音				
b	b	有声両唇破裂音	baban	p
d	d	有声歯茎破裂音	deg	t
ʤ	dZ	有声後部歯茎破擦音	garej	S
ð	D	有声歯摩擦音	deuddeg	T

IPA	X-SAMPA	説明	例	ビゼーム
f	f	無声唇歯摩擦音	ffacs	f
Amazon EC2 のセキュリティグループ	Amazon EC2 のセキュリティグループ	有声軟口蓋破裂音	gadael	k
h	h	無声声門摩擦音	haearn	k
j	j	硬口蓋接近音	astudio	i
k	k	無声軟口蓋破裂音	cant	k
l	l	歯茎側面接近音	lan	t
ɬ	K USD	無声歯茎側面摩擦音	llan	t
m	m	両唇鼻音	mae	p
m̥	m_0	無声両唇鼻音	ymhen	p
n	n	歯茎鼻音	naw	t
n̥	n_0	無声歯茎鼻音	anhawster	t
ŋ	N	軟口蓋鼻音	argyfwng	k
ŋ̊	N_0	無声軟口蓋鼻音	anghenion	k
p	p	無声両唇破裂音	pump	p
r	r	歯茎ふるえ音	rhoi	r
r̥	r_0	無声歯茎ふるえ音	garw	r
s	s	無声歯茎摩擦音	saith	s

IPA	X-SAMPA	説明	例	ビゼーム
ʃ	S	無声後部歯茎摩擦音	siawns	S
t	t	無声歯茎破裂音	tegan	t
͡tʃ	tS	無声後部歯茎破擦音	cytsain	S
θ	T	無声歯摩擦音	aberth	T
v	v	有声唇歯摩擦音	prawf	f
w	w	有声両唇軟口蓋接近音	rhagweld	u
χ	X	無声口蓋垂摩擦音	chwech	k
z	z	有声歯茎摩擦音	aids	s
ʒ	Z	有声後部歯茎摩擦音	rouge	S
母音				
ə	@	中段中舌	ychwanega	@
a	a	非円唇前舌広母音	acen	a
ai	ai	二重母音	dau	a
au	au	二重母音	awdur	a
ɑː	A:	非円唇後舌広長母音	mab	a
ɑːɨ	A:1	二重母音	aelod	a
eː	e:	非円唇前舌半狭長母音	peth	e

IPA	X-SAMPA	説明	例	ビゼーム
ɛ	E	非円唇前舌半広母音	pedwar	E
ɛi	Ei	二重母音	beic	E
インスタンス:	インスタンス:	非円唇前舌狭長母音	tri	i
ɪ	I	非円唇前舌め広めの狭母音	miliwn	i
ɨu	1u	二重母音	unigryw	i
オ:	オ:	円唇後舌半狭長母音	oddi	o
ɔ	O	円唇後舌半広母音	oddieithr	O
ɔi	Oi	二重母音	troi	O
ɔu	Ou	二重母音	rownd	O
u:	u:	円唇後舌狭長母音	cwch	u
ʊ	U	円唇後舌め広めの狭母音	acwstig	u
ʊi	Ui	二重母音	wyth	u
その他の記号				
ˈ	"	第一アクセント	Alabama	
ˌ	%	第二アクセント	Alabama	
.	.	音節境界	A.la.ba.ma	

Amazon Polly 音声エンジン

Amazon Polly には、入力テキストをリアルな音声に変換する 4 つの音声エンジンがあります。これには、生成 、ロングフォーム 、ニューラル 、および標準 が含まれます。Amazon Polly 音声を使用するには、エンジンと音声合成 API オペレーションを選択します。次に、エンジンが合成する入力テキストを入力し、オーディオ出力形式を選択します。これらの入力を考慮すると、Amazon Polly は提供されたテキストを高品質の音声音声ストリームに合成します。

以下のセクションでは、Amazon Polly が提供する音声エンジンの詳細について説明します。

トピック

生成音声

Amazon Polly の生成 text-to-speech (TTS) エンジンは、Amazon Polly コンソールを介して使用できる最も人間らしく、感情的にエンゲージし、適応性の高い会話音声を提供します。

生成エンジンは、現在までに最大の Amazon Polly TTS モデルです。生のテキストを音声コードに変換する 10 億個のパラメータトランスフォーマーをデプロイし、その後にこれらの音声コードを増分でストリーミング可能な方法で波形に変換する畳み込みベースのデコーダーをデプロイします。この方法では、さまざまな音声、言語、スタイルで構成される公開データおよび専有データの量の増加についてトレーニングした場合、大規模言語モデル (LLMsの広く報告された緊急能力を示します。

生成エンジンは、人間の音声と非常によく似た方法で、感情的にエンゲージし、アサーションがあり、高い共謀性を持つ合成音声を作成します。これらの音声は、知識豊富なカスタマーアシスタント、仮想トレーナー、または人間に近い合成音声を持つ広告者として使用できます。

> **Note**
>
> これらの音声の基礎となる state-of-the-art テクノロジーは、言語および音声モデリングの生成 AI のパラダイムに含まれます。テクノロジーの副作用は、トレーニングデータとモデルを更新すると、モデルの更新によって全体的な品質が向上する場合でも、音声のサウンドにわずかなばらつきが生じる可能性があることです。これは、ポッドキャストの季節など、長期

間にわたって合成されたさまざまなコンテンツパートを持つユースケースに影響を与える可能性があります。

使用可能な生成音声

Amazon Polly は現在、生成バリアントで 2 人の女性と 1 人の男性の英語音声を提供しています。これらの生成音声は、会話型 NTTS バリアントでも使用できます。

	[言語]	言語コード	名前/ID	性別
1	英語 (英国)	en-GB	Amy	女性
2	英語 (米国)	en-US	Matthew Ruth	男 女性

> **Note**
>
> 生成音声のコストは、Amazon Polly の料金情報ページ で指定します。

機能とリージョンの互換性

Amazon Polly 生成音声は、次のリージョンで使用できます。

- 米国東部 (バージニア北部) リージョン
- 他のリージョンは使用できません

生成音声では、次の機能がサポートされています。

- リアルタイムおよび非同期の音声合成オペレーション。
- ニュースキャスターの話し方は、生成エンジンではサポートされていません。
- 多くの (すべてではない) SSML タグは、Amazon Polly でサポートされています。NTTS でサポートされる SSML タグの詳細については、「サポートされている SSML タグ」を参照してください。

- 標準音声と同様に、さまざまなサンプリングレートから選択して、アプリケーションの帯域幅と音質を最適化できます。標準およびニューラル音声の有効なサンプリングレートは、8 kHz、16 kHz、22 kHz、または 24 kHz です。標準音声のデフォルトは 22 kHz です。生成音声のデフォルトは 24 kHz です。Amazon Polly は MP3、OGG (Vorbis)、raw PCM オーディオストリーム形式をサポートしています。
- 新しい Amazon Polly 生成音声のレイテンシーは 100 ミリ秒です。

音声マークの生成は現在サポートされていません。

> **Note**
>
> 万一モデル幻覚が発生した場合 (および生成エンジンのモデル動作で音声トークンをトークンでレンダリングする場合) 、強制的な緊急停止メカニズムが設定されています。組み込みメカニズムは、モデルによる音声のレンダリングを停止します。この安全機能は、モデルが幻覚を引き起こす可能性があるデータ分析に基づいています。通常は文の最後にあります。モデルが幻覚を起こしそうだと考え、生成ステップ中に単語を切断し、単語の半分をレンダリングしてしまう場合があります。これにより、不適切な結果が生じる可能性があります。

コンソールで生成エンジンを使用する

Amazon Polly の生成音声には、Amazon Polly コンソールまたは からアクセスできます AWS CLI。コンソールから生成エンジンを選択し、リストから対応する生成音声を選択して、その音声の合成音声を聞きます。`SynthesizeSpeech` および `StartSpeechSynthesisTask` API オペレーションを使用して生成音声を調べることもできます。API オペレーションでは、API リクエストでエンジンと音声の名前を指定できます。Python を使用したクイックスタート入門コードの例については、「Python の例」を参照してください。

コンソールで生成エンジンを使用するには

1. Amazon Polly コンソール (https://console.aws.amazon.com/polly/) を開きます。
2. Amazon Polly コンソールから、生成エンジンを選択します。
3. 音声ドロップダウンメニューから目的の音声を選択します。
4. 選択したテキストで TTS オーディオを生成します。

> **Note**
>
> 生成音声は、 **SynthesizeSpeech**および **StartSpeechSynthesisTask** API オペレーションでも使用できます。API オペレーションでは、お客様は API リクエストでエンジンと音声の名前を指定できます。その他のクイックスタートコードサンプルについては、こちらをご覧ください。

ロングフォーム音声

Amazon Polly には、人間のような表現力が高く、感情に慣れた音声を生成するロングフォームエンジンがあります。ロングフォーム音声は、ニュース記事、トレーニング資料、マーケティング動画など、長いコンテンツのリスナーの注意を引くように設計されています。

Amazon Polly ロングフォーム音声は、最先端の深層学習 TTS テクノロジーを使用して開発されています。このモデルは、人間の言語の音素、韻律、イントネーション、その他の音声や音響の要素を再現することを学習し、非常に自然な音声出力を実現します。

ロングフォームエンジンは、テキスト埋め込みを使用してテキストの意味を解釈します。テキスト埋め込みを使用すると、ロングフォームエンジンは自然な音声の正しい強調、一時停止、トーンを生成できます。その結果、人間のコミュニケーションに存在する感情的要素の全範囲を組み合わせた音声が得られます。これには、あいまいさを模倣したり、対話をナレーションと区別したりすることが含まれます。これらが合わさって、生きている人間のように聞こえるプレミアムな音声製品となっています。

> **Note**
>
> これらの音声の基礎となる state-of-the-art テクノロジーは、言語および音声モデリングの生成 AI のパラダイムに含まれます。テクノロジーの副作用は、トレーニングデータとモデルを更新すると、モデルの更新によって全体的な品質が向上する場合でも、音声のサウンドにわずかなばらつきが生じる可能性があることです。これは、ポッドキャストの季節など、長期間にわたって合成されたさまざまなコンテンツパートを持つユースケースに影響を与える可能性があります。

使用可能なロングフォーム音声

Amazon Polly は現在、2 つの女性と 1 つの男性の en-US ロングフォーム音声を提供しています。これらのロングフォーム音声は、会話型 NTTS バージョンでもご利用いただけます。

	[言語]	言語コード	名前/ID	性別
1	英語 (米国)	en-US	Danielle	女性
			Gregory	男
			Ruth	女性

機能とリージョンの互換性

Amazon Polly のロングフォーム音声は、次のリージョンで使用できます。

- 米国東部 (バージニア北部) リージョン
- 他のリージョンは使用できません

Amazon Polly ロングフォームエンジンは、次の機能をサポートしています。

- リアルタイムおよび非同期の音声合成オペレーション。
- すべてのスピーチマーク。
- 多くの (すべてではない) SSML タグは、Amazon Polly でサポートされています。NTTS でサポートされる SSML タグの詳細については、「サポートされている SSML タグ」を参照してください。
- 100 ミリ秒のレイテンシー。
- 標準音声と同様に、さまざまなサンプリングレートから選択して、アプリケーションの帯域幅と音質を最適化できます。標準の音声、ロングフォーム音声、およびニューラル音声の有効なサンプリングレートは、8 kHz、16 kHz、22 kHz、または 24 kHz です。標準音声のデフォルトは 22 kHz です。ロングフォーム音声およびニューラル音声のデフォルトは 24 kHz です。Amazon Polly は MP3、OGG (Vorbis)、raw PCM オーディオストリーム形式をサポートしています。

> **Note**
>
> ロングフォーム音声のコストは、Amazon Polly の料金情報ページ で指定されています。

コンソールでのロングフォームエンジンの使用

Amazon Polly のロングフォーム音声には、Amazon Polly コンソールまたは AWS CLIからアクセスできます。

コンソールでロングフォームエンジンを使用するには

1. Amazon Polly コンソール (https://console.aws.amazon.com/polly/) を開きます。
2. Amazon Polly コンソールから、ロングフォームエンジンを選択します。
3. 音声ドロップダウンメニューから目的の音声を選択します。
4. 選択したテキストで TTS オーディオを生成します。

> **Note**
>
> ロングフォーム音声は、 **SynthesizeSpeech**および **StartSpeechSynthesisTask** API オペレーションでも使用できます。API オペレーションでは、お客様は API リクエストでエンジンと音声の名前を指定できます。その他のクイックスタートコードサンプルについては、こちらをご覧ください。

ニューラル音声

Amazon Polly には、標準の音声よりもさらに高品質の音声を生成できるニューラル text-to-speech (NTTS) エンジンがあります。標準の TTS 音声では、連結合成が使用されます。標準エンジンは、録音された音声の音素を連結し、非常に自然に聞こえる合成音声を生成します。ただし、音声の必然的なバリエーションや波形をセグメント化するために使用される手法によって、音声の品質が制限されます。Amazon Polly NTTS エンジンは、標準の連結合成を使用して音声を生成しません。これには 2 つの部分があります。

- ニューラルネットワーク — 一連の音素 (最も基本的な言語単位) を一連の分光法に変換します。(スペトグラムは、異なる周波数帯域のエネルギーレベルのスナップショットです。)
- ボコーダー — 分光法をほぼ連続した音声信号に変換します。

ニューラル TTS システムの最初のコンポーネントは sequence-to-sequence モデルです。このモデルは、対応する入力からのみ結果を作成するのではなく、入力要素のシーケンスがどのように連携するかを考慮します。このモデルは、出力するスペクトログラムを選択し、その周波数帯が、音声を処理するときに人間の脳が使用する音響能力を強調するようにします。

このモデルの出力は、ニューラルボコーダーに渡されます。これにより、スペクトログラムが音声波形に変換されます。汎用連結合成システムの構築に使用される大規模なデータセットでトレーニングすると、この sequence-to-sequence アプローチにより、より高品質で自然な音声が生成されます。

使用可能なニューラル音声

ニューラル音声は 33 の言語と言語バリアントで利用できます。以下の表にそれらの設定を示します。

	言語と言語バリアント	言語コード	名前/ID	性別
1	アラビア語 (湾岸)	ar-AE	ハラ Zayd	女性 男
2	ベルギーオランダ語 (フランドル語)	nl-BE	Lisa	女性
3	カタロニア語	ca-ES	Arlet	女性
4	中国語 (広東語)	yue-CN	Hiujin	女性
5	標準中国語	cmn-CN	Zhiyu	女性
6	デンマーク語	da-DK	Sofie	女性
7	オランダ語	nl-NL	Laura	女性
8	英語 (オーストラリア)	en-AU	Olivia	女性
9	英語 (英国)	en-GB	Amy* Emma	女性 女性

	言語と言語バリアント	言語コード	名前/ID	性別
			Brian	男
			Arthur	男
10	英語 (インド)	en-IN	カジャル語	女性
11	英語 (アイルランド語)	en-IN	Niamh	女性
12	英語 (ニュージーランド)	en-NZ	Aria	女性
13	英語 (南アフリカ)	en-ZA	Ayanda	女性

	言語と言語バリアント	言語コード	名前/ID	性別
14	英語 (米国)	en-US	Danielle	女性
			Gregory	男
			Ivy	女性 (子)
			Joanna*	女性
			Kendra	女性
			Kimberly	女性
			Salli	女性
			Joey	男
			Justin	男性 (子)
			Kevin	男性 (子)
			Matthew*	男
			Ruth	女性
			Stephen	男
15	フィンランド語	fi-FI	Suvi	女性
16	フランス語 (ベルギー)	fr-BE	Isabelle	女性
17	フランス語 (カナダ)	fr-CA	Gabrielle	女性
			Liam	男
18	フランス語	fr-FR	Léa	女性
			Rémi	男

	言語と言語バリアント	言語コード	名前/ID	性別
19	ドイツ語	de-DE	Vicki	女性
			Daniel	男
20	ドイツ語 (オーストリア)	de-AT	Hannah	女性
21	ヒンディー語	hi-IN	カジャル語	女性
22	イタリア語	it-IT	Bianca	女性
			Adriano	男
23	日本語	ja-JP	Takumi	男
			Kazuha	女性
			Tomoko	女性
24	韓国語	ko-KR	Seoyeon	女性
25	ノルウェー語	nb-NO	Ida	女性
26	ポーランド語	pl-PL	Ola	女性
27	ポルトガル語 (ブラジル)	pt-BR	Camila	女性
			Vitória/Vitoria	女性
			Thiago	男
28	ポルトガル語 (欧州)	pt-PT	Inês/Ines	女性
29	スペイン語 (欧州)	es-ES	Lucia	女性
			Sergio	男

	言語と言語バリアント	言語コード	名前/ID	性別
30	スペイン語 (メキシコ)	es-MX	Mia Andrés	女性 男
31	スペイン語 (米国)	es-US	Lupe* Pedro	女性 男
32	スウェーデン語	sv-SE	Elin	女性
33	トルコ語	tr-TR	ブルク	女性

*Amy、Joanna、Lupe、Matthew の音声は、ニュースキャスターの話し方で使用できます。詳細については、「ニュースキャスター音声」を参照してください。

トピック

- 機能とリージョンの互換性
- コンソールでのニューラルエンジンの使用

機能とリージョンの互換性

ニューラル音声は、すべての AWS リージョンで利用できるわけではなく、すべての Amazon Polly 機能をサポートしているわけでもありません。

ニューラル音声は、次のリージョンでサポートされています。

- 米国東部 (バージニア北部): us-east-1
- 米国西部 (オレゴン): us-west-2
- アフリカ (ケープタウン): af-south-1
- アジアパシフィック (東京): ap-northeast-1
- アジアパシフィック (ソウル): ap-northeast-2
- アジアパシフィック (大阪): ap-northeast-3
- アジアパシフィック (ムンバイ): ap-south-1

- アジアパシフィック (シンガポール): ap-southeast-1
- アジアパシフィック (シドニー): ap-southeast-2
- カナダ (中部): ca-central-1
- 欧州 (フランクフルト): eu-central-1
- 欧州 (アイルランド): eu-west-1
- 欧州 (ロンドン): eu-west-2
- 欧州 (パリ): eu-west-3
- AWS GovCloud （ 米国西部): us-gov-west-1

これらのリージョンのエンドポイントとプロトコルは、標準音声に使用されるものと同じです。詳細については、Amazon Polly エンドポイントとクォータを参照してください。

ニューラル音声では、以下の機能がサポートされています。

- リアルタイムおよび非同期の音声合成オペレーション。
- ニュースキャスターの話し方。話し方の詳細については、ニュースキャスター音声を参照してください。
- すべてのスピーチマーク。
- Amazon Polly でサポートされている多くの SSML タグ (すべてではありません)。NTTS でサポートされている SSML タグの詳細については、「サポートされているタグ」を参照してください。

標準音声と同様に、さまざまなサンプリングレートから選択して、アプリケーションの帯域幅と音質を最適化できます。標準およびニューラル音声の有効なサンプリングレートは、8 kHz、16 kHz、22 kHz、または 24 kHz です。標準音声のデフォルトは 22 kHz です。ニューラル音声のデフォルトは 24 kHz です。Amazon Polly は MP3、OGG (Vorbis)、raw PCM オーディオストリーム形式をサポートしています。

コンソールでのニューラルエンジンの使用

Amazon Polly ニューラル音声には、Amazon Polly コンソールまたは からアクセスできます AWS CLI。

コンソールでニューラルエンジンを使用するには

1. Amazon Polly コンソール (https://console.aws.amazon.com/polly/) を開きます。
2. コンソールからニューラルエンジンを選択します。

3. 音声ドロップダウンメニューから目的の音声を選択します。
4. 選択したテキストで TTS オーディオを生成します。

標準音声

Amazon Polly には、連結合成を使用する標準エンジンがあります。標準エンジンは、録音された音声の音素を連結し、非常に自然に聞こえる合成音声を生成します。

使用可能な標準音声

Amazon Polly は現在、29 の言語および言語バリアントで 40 の女性音声と 20 の男性標準音声を提供しています。

	[言語]	言語コード	名前/ID	性別
1	アラビア語	arb	Zeina	女性
2	標準中国語	cmn-CN	Zhiyu	女性
3	デンマーク語	da-DK	Naja	女性
			Mads	男
4	オランダ語	nl-NL	Lotte	女性
			Ruben	男
5	英語 (オーストラリア)	en-AU	Nicole	女性
			Russell	男
6	英語 (英国)	en-GB	Amy	女性
			Emma	女性
			Brian	男
7	英語 (インド)	en-IN	Aditi	女性
			Raveena	女性

	[言語]	言語コード	名前/ID	性別
8	英語 (米国)	en-US	Ivy	女性
			Joanna	女性
			Kendra	女性
			Kimberly	女性
			Salli	女性
			Joey	男
			Kevin	男
9	英語 (ウェールズ)	en-GB-WLS	Geraint	男
10	フランス語	fr-FR	Céline/Celine	女性
			Léa	女性
			Mathieu	男
11	フランス語 (カナダ)	fr-CA	Chantal	女性
12	ドイツ語	de-DE	Marlene	女性
			Vicki	女性
			Hans	男
13	ヒンディー語	hi-IN	Aditi	女性
14	アイスランド語	is-IS	Dóra/Dora	女性
			Karl	男

	[言語]	言語コード	名前/ID	性別
15	イタリア語	it-IT	Carla	女性
			Bianca	女性
			Giorgio	男
16	日本語	ja-JP	Mizuki	女性
			Takumi	男
17	韓国語	ko-KR	Seoyeon	女性
18	ノルウェー語	nb-NO	Liv	女性
19	ポーランド語	pl-PL	Ewa	女性
			Maja	女性
			Jacek	男
			Jan	男
20	ポルトガル語 (ブラジル)	pt-BR	Camila	女性
			Vitória/Vitoria	女性
			Ricardo	男
21	ポルトガル語 (欧州)	pt-PT	Inês/Ines	女性
			Cristiano	男
22	ルーマニア語	ro-RO	Carmen	女性
23	ロシア語	ru-RU	Tatyana	女性
			Maxim	男

	[言語]	言語コード	名前/ID	性別
24	スペイン語 (欧州)	es-ES	Conchita Lucia Enrique	女性 女性 男
25	スペイン語 (メキシコ)	es-MX	Mia	女性
26	スペイン語 (米国)	es-US	Lupe Penélope/ Penelope Miguel	女性 女性 男
27	スウェーデン語	sv-SE	Astrid	女性
28	トルコ語	tr-TR	Filiz	男
29	ウェールズ語	cy-GB	Gwyneth	女性

機能とリージョンの互換性

Amazon Polly 標準音声は、次の 22 の Amazon Polly リージョンすべてで使用できます。

- 米国東部 (バージニア北部): us-east-1
- 米国西部 (オレゴン): us-west-2
- アフリカ (ケープタウン): af-south-1
- アジアパシフィック (東京): ap-northeast-1
- アジアパシフィック (ソウル): ap-northeast-2
- アジアパシフィック (大阪): ap-northeast-3
- アジアパシフィック (ムンバイ): ap-south-1
- 中国 (寧夏) リージョン
- アジアパシフィック (シンガポール): ap-southeast-1

- アジアパシフィック (シドニー): ap-southeast-2
- カナダ (中部): ca-central-1
- 欧州 (フランクフルト): eu-central-1
- 欧州 (アイルランド): eu-west-1
- 欧州 (ロンドン): eu-west-2
- 欧州 (パリ): eu-west-3
- AWS GovCloud （ 米国西部): us-gov-west-1

これらのリージョンのエンドポイントとプロトコルは、ニューラル音声に使用されるものと同じです。詳細については、Amazon Polly エンドポイントとクォータを参照してください。

Amazon Polly 標準エンジンは、以下の機能 (TBD) をサポートしています。

- リアルタイムおよび非同期の音声合成オペレーション。
- すべてのスピーチマーク。
- 多くの (ただし、すべてではない) SSML タグは Amazon Polly でサポートされています。NTTS でサポートされている SSML タグの詳細については、「サポートされている SSML タグ」を参照してください。
- 100 ミリ秒のレイテンシー。
- さまざまなサンプリングレートから選択して、アプリケーションの帯域幅とオーディオ品質を最適化できます。標準音声のデフォルトのサンプリングレートは 22 kHz です。Amazon Polly は MP3、OGG (Vorbis)、raw PCM オーディオストリーム形式をサポートしています。

> **Note**
>
> 標準音声のコストは、Amazon Polly の料金情報ページ で指定します。

コンソールでの標準音声の使用

Amazon Polly コンソールまたは から Amazon Polly 標準音声にアクセスできます AWS CLI。

コンソールで標準音声を使用するには

1. Amazon Polly コンソール (https://console.aws.amazon.com/polly/) を開きます。

2. Amazon Polly コンソールから、標準エンジンを選択します。
3. 音声ドロップダウンメニューから目的の音声を選択します。
4. 選択したテキストで TTS オーディオを生成します。

> **Note**
>
> 標準音声は、 **SynthesizeSpeech**および **StartSpeechSynthesisTask** API オペレーションでも使用できます。API オペレーションでは、お客様は API リクエストでエンジンと音声の名前を指定できます。クイックスタートコードのサンプルは、より多く見つかります。

スピーチマーク

スピーチマークは、合成する音声を表すメタデータです。たとえば、文章または単語が音声ストリームで開始し、終了する場合です。テキストのスピーチマークをリクエストすると、Amazon Polly は、合成された音声の代わりにこのメタデータを返します。合成されたスピーチ音声ストリームとスピーチマークを組み合わせて使用することで、高度なビジュアル体験を提供するアプリケーションを構築できます。

例えば、テキストのメタデータと音声ストリームを組み合わせると、スピーチと表情アニメーション (リップシンキング) を同期したり、読み上げられたとおりに書かれた単語をハイライト表示したりできます。

音声マークは、ニューラル形式または標準 text-to-speech形式を使用する場合に使用できます。

トピック

スピーチマークタイプ

SynthesizeSpeech または StartSpeechSynthesisTask コマンドの SpeechMarkTypesオプションを使用して、音声マークをリクエストします。入力テキストから返すメタデータ要素を指定します。最大 4 種類のメタデータをリクエストできますが、リクエストごとに少なくとも 1 種類指定する必要があります。このリクエストで音声出力が生成されることはありません。

AWS CLI例えば、 では、次のようにします。

```
--speech-mark-types='["sentence", "word", "viseme", "ssml"]'
```

Amazon Polly は、次の要素を使用してスピーチマークを生成します。

- sentence (文) - 入力テキストで文の要素を示します。
- word (単語) - 入力テキストで単語の要素を示します。
- viseme (ビゼーム) - 話されている各音素に対応する顔と口の動きを表します。詳細については、「ビゼームと Amazon Polly」を参照してください。

- ssml – SSML 入力テキストからの <mark> 要素を表します。詳細については、「SSML ドキュメントからの音声の生成」を参照してください。

ビゼームと Amazon Polly

ビゼームは、単語を読み上げる際の顔と口の位置を表します。これは、単語が作られる基本的な音声の単位である音素と視覚的に同等です。ビゼームは、音声の基本的な視覚の構成要素です。

各言語には、特定の音素に対応する一連のビゼームがあります。言語の各音素には、音を作り出す際の口の形を表すビゼームが含まれます。ただし、音声が異なる場合でも、読み上げる際に膨大な音素が表示されるため、すべてのビゼームが特定の音素にマッピングできるとは限りません。たとえば、英語では、"pet" と "bet" という単語は聴覚的に異なります。ただし、視覚で確認された場合 (音なし)、それらの見た目は全く同じです。

以下の図では、国際音声記号 (IPA) の音素のリストの一部、拡張 SAM 音声記号 (X-SAMPA)、米国英語の音声に関するビゼームについて説明しています。

完全な表および使用可能なすべての言語の表については、「サポートされている言語の音素およびビゼームテーブル」を参照してください。

IPA	X-SAMPA	説明	例	ビゼーム
子音				
b	b	有声両唇破裂音	bed	p
d	d	有声歯茎破裂音	dig	t
ʧ	dZ	有声後部歯茎破裂音	jump	S
ð	D	有声歯摩擦音	次に	T
f	f	無声唇歯摩擦音	five	f
Amazon EC2 のセキュリティ	Amazon EC2 のセキュリティグループ	有声軟口蓋破裂音	game	k

IPA	X-SAMPA	説明	例	ビゼーム
グループ				
h	h	無声声門摩擦音	house	k
...	...	...	...	...

スピーチマークの使用

スピーチマークのリクエスト

入力テキストのスピーチマークをリクエストするには、synthesize-speech コマンドを使用します。このメタデータを返すには、入力テキストに加えて、以下の要素が必要です。

- output-format

  Amazon Polly は、スピーチマークを返す際、JSON 形式のみサポートしています。

  ```
  --output-format json
  ```

  サポートされていない出力形式を使用する場合、Amazon Polly は例外をスローします。

- voice-id

  メタデータが、音声ストリームと一致することを確認するには、合成されたスピーチ音声ストリームの生成に使用されているのと同じ音声を指定します。使用可能な音声に一致する音声速度がありません。音声を生成するために使用する音声とは別の音声を使用する場合、メタデータは音声ストリームと一致しません。

  ```
  --voice-id Joanna
  ```

- speech-mark-types

  スピーチマークの種類を指定します。スピーチマークの種類のいずれか、またはすべてをリクエストできますが、少なくとも 1 つ指定する必要があります。

  ```
  --speech-mark-types='["sentence", "word", "viseme", "ssml"]'
  ```

- `text-type`

  プレーンテキストは、Amazon Polly のデフォルトの入力テキストのため、SSML スピーチマークを返す場合は、`text-type ssml` を使用する必要があります。

- `outfile`

  メタデータが書き込まれた出力ファイルを指定します。

```
MaryLamb.txt
```

次の AWS CLI 例は、Unix、Linux、macOS 用にフォーマットされています。Windows の場合は、各行末のバックスラッシュ (\) Unix 連結文字をカレット (^) に置き換え、入力テキストは二重引用符(") で囲み、内部タグは一重引用符 (') で囲みます。

```
aws polly synthesize-speech \
  --output-format json \
  --voice-id Voice ID \
  --text 'Input text' \
  --speech-mark-types='["sentence", "word", "viseme"]' \
  outfile
```

スピーチマーク出力

Amazon Polly は、スピーチマークのオブジェクトを改行で区切られた JSON ストリームで返します。スピーチマークのオブジェクトには、次のフィールドが含まれます。

- time (時間) – 対応する音声ストリームの開始からのタイムスタンプ (ミリ秒)。
- type (種類) - スピーチマークの種類 (文、単語、ビゼーム、または ssml)
- start (開始) - 入力テキストのオブジェクトの開始からのオフセット (文字ではなくバイト) (ビゼームマークを含まない)
- end (終了) - 入力テキストのオブジェクトの終了のオフセット (文字ではなくバイト) (ビゼームマークを含まない)
- value (値) - スピーチマークの種類によって異なります
  - SSML: <mark> SSML タグ
  - viseme: ビゼーム名

- word または sentence: 入力テキストの部分文字列。開始および終了フィールドで区切られます

例えば、Amazon Polly では、「Mary had a little lamb」というテキストから、次の word スピーチマークのオブジェクトを生成します。

```
{"time":373,"type":"word","start":5,"end":8,"value":"had"}
```

記載された単語 (「had」) は、音声ストリーム開始後、373 ミリ秒後に開始して、入力テキストの 5 バイトめで開始し、8 バイトめで終了します。

> **Note**
>
> このメタデータは、Joanna の voice-id に使用されます。同じ入力テキストの別の音声を使用する場合、メタデータは異なる場合があります。

スピーチマークの例

スピーチマークの次の例では、共通リクエストの作成方法と、そのリクエストによって生成される出力を示します。

例 1: SSML を使用しないスピーチマーク

以下の例では、リクエストされたメタデータが、シンプルな文章「Mary had a little lamb.」で画面に表示される様子を示しています。分かりやすいように、SSML のスピーチマークはこの例に含んでいません。

次の AWS CLI 例は、Unix、Linux、macOS 用にフォーマットされています。Windows の場合は、各行末のバックスラッシュ (\) Unix 連結文字をカレット (^) に置き換え、入力テキストは二重引用符 (") で囲み、内部タグは一重引用符 (') で囲みます。

```
aws polly synthesize-speech \
  --output-format json \
  --voice-id Joanna \
  --text 'Mary had a little lamb.' \
  --speech-mark-types='["viseme", "word", "sentence"]' \
```

```
MaryLamb.txt
```

このリクエストをすると、Amazon Polly は、以下の内容を .txt ファイルで返します。

```
{"time":0,"type":"sentence","start":0,"end":23,"value":"Mary had a little lamb."}
{"time":6,"type":"word","start":0,"end":4,"value":"Mary"}
{"time":6,"type":"viseme","value":"p"}
{"time":73,"type":"viseme","value":"E"}
{"time":180,"type":"viseme","value":"r"}
{"time":292,"type":"viseme","value":"i"}
{"time":373,"type":"word","start":5,"end":8,"value":"had"}
{"time":373,"type":"viseme","value":"k"}
{"time":460,"type":"viseme","value":"a"}
{"time":521,"type":"viseme","value":"t"}
{"time":604,"type":"word","start":9,"end":10,"value":"a"}
{"time":604,"type":"viseme","value":"@"}
{"time":643,"type":"word","start":11,"end":17,"value":"little"}
{"time":643,"type":"viseme","value":"t"}
{"time":739,"type":"viseme","value":"i"}
{"time":769,"type":"viseme","value":"t"}
{"time":799,"type":"viseme","value":"t"}
{"time":882,"type":"word","start":18,"end":22,"value":"lamb"}
{"time":882,"type":"viseme","value":"t"}
{"time":964,"type":"viseme","value":"a"}
{"time":1082,"type":"viseme","value":"p"}
```

この出力では、各テキスト部分がスピーチマーク単位で表示されます。

- 文「Mary had a little lamb.」
- テキスト内の各単語: "Mary"、"had"、"a"、"little"、"lamb"
- 対応する音声データストリームの各音のビゼームには、"p"、"E"、"r"、"i" などがあります。ビゼームの詳細については、「ビゼームと Amazon Polly」を参照してください。

例 2: SSML を使用したスピーチマーク

SSML 拡張テキストからスピーチマークを生成する手順は、SSML が存在しない場合のプロセスと同様です。`synthesize-speech` コマンドを使用して、次の例に示すように SSML 拡張テキスト、スピーチマークの種類を指定します。例を読みやすくするために、ビゼームスピーチマークは含めませんが、これらも含めることもできます。

次の AWS CLI 例は、Unix、Linux、macOS 用にフォーマットされています。Windows の場合は、各行末のバックスラッシュ (\) Unix 連結文字をカレット (^) に置き換え、入力テキストは二重引用符 (") で囲み、内部タグは一重引用符 (') で囲みます。

```
aws polly synthesize-speech \
  --output-format json \
  --voice-id Joanna \
  --text-type ssml \
  --text '<speak><prosody volume="+20dB">Mary had <break time="300ms"/>a little <mark
 name="animal"/>lamb</prosody></speak>' \
  --speech-mark-types='["sentence", "word", "ssml"]' \
  output.txt
```

このリクエストをすると、Amazon Polly は、以下の内容を .txt ファイルで返します。

```
{"time":0,"type":"sentence","start":31,"end":95,"value":"Mary had <break time=\"300ms
\"\/>a little <mark name=\"animal\"\/>lamb"}
{"time":6,"type":"word","start":31,"end":35,"value":"Mary"}
{"time":325,"type":"word","start":36,"end":39,"value":"had"}
{"time":897,"type":"word","start":40,"end":61,"value":"<break time=\"300ms\"\/>"}
{"time":1291,"type":"word","start":61,"end":62,"value":"a"}
{"time":1373,"type":"word","start":63,"end":69,"value":"little"}
{"time":1635,"type":"ssml","start":70,"end":91,"value":"animal"}
{"time":1635,"type":"word","start":91,"end":95,"value":"lamb"}
```

コンソールでの音声マークのリクエスト

コンソールを使用して、Amazon Polly からスピーチマークをリクエストできます。その後、メタデータを表示するか、ファイルに保存できます。

スピーチマークを生成するには (コンソール)

1. にサインイン AWS Management Console し、https://console.aws.amazon.com/polly/ で Amazon Polly コンソールを開きます。
2. [Text-to-Speech (テキスト読み上げ機能)] タブを選択します。
3. SSML を使用するには、[SSML] を有効にします。
4. テキストを入力ボックスに入力するか貼り付けます。
5. [Language] (言語) でテキストの言語を選択します。
6. [Voice] で、このテキストで使用する音声を選択します。

7. テキストの発音を変更するには、[Additional settings] (詳細設定) を展開し、[Customize pronunciation] (発音のカスタマイズ) を有効にして、[Apply lexicon] (レキシコンの適用) で目的のレキシコンを選択します。
8. スピーチが最終形式であることを確認するには、[Listen] (聴く) を選択します。
9. [Speech file format settings] (音声ファイル形式の設定) を有効にします。

> **Note**
>
> MP3、OGG、または PCM 形式をダウンロードしてもスピーチマークは生成されません。

10. [File Format] (ファイル形式) で、[Speech Marks] (スピーチマーク) を選択します。
11. [Speech Mark Types] (スピーチマークの種類) で、生成するスピーチマークの種類を選択します。SSML メタデータを選択するオプションは、[SSML] が有効になっている場合にのみ利用できます。Amazon Polly での SSML の使用の詳細については、SSML ドキュメントからの音声の生成 を参照してください。
12. [ダウンロード] を選択します。

SSML ドキュメントからの音声の生成

Amazon Polly を使用して、プレーンテキストまたは音声合成マークアップ言語 (SSML) を使用したドキュメントから音声を生成できます。SSML 拡張のテキストを使用すれば、Amazon Polly を使用して入力したテキストから音声を生成する方法を詳細に制御することができます。

たとえば、テキスト内に長い一時停止時間を追加したり、話す速度やピッチを変更できます。その他のオプションには以下が含まれます。

- 特定の単語やフレーズを強調する
- 発音記号を使用する
- 呼吸音を含む
- ウィスパー
- ニュースキャスターの話し方を使用します。

Amazon Polly でサポートされる SSML タグとその使用方法の詳細については、サポートされている SSML タグを参照してください。

SSML を使用する場合、特別な処理が必要な予約文字がいくつかあります。これは、SSML がコードの一部としてこれらの文字を使用するためです。それらを使用するには、特定のエンティティを使用してエスケープします。詳細については、SSML の予約文字を参照してください。

Amazon Polly では、音声合成マークアップ言語 (SSML) バージョン 1.1、W3C 勧告で定義されている SSML マークアップタグのサブセットを使用して、これらのタイプのコントロールを提供します。

SSML を使用するには、Amazon Polly コンソール、または AWS CLIを使用します。次のトピックでは、SSML を使用して音声を生成する方法および目的に正確に適合するように出力を制御する方法を説明します。

トピック

- SSML の予約文字
- コンソールでの SSML の使用
- での SSML の使用 AWS CLI
- サポートされている SSML タグ

SSML の予約文字

SSML ステートメントには、通常使用できない事前定義された文字が 5 つあります。これらのエンティティは、言語仕様によって予約されています。これらの文字は次のとおりです。

名前	文字	エンティティコード
引用符 (二重引用符)	"	&quot;
アンパサンド	&	&amp;
アポストロフィまたは	'	&apos;

名前	文字	エスケープコード
一重引用符	'	&apos;
より小さい記号	<	&lt;
より大きい記号	>	&gt;

SSML はコードの一部としてこれらの文字を使用するため、SSML でこれらの記号を使用するには、文字を使用するときにエスケープする必要があります。実際の文字の代わりにエスケープコードを使用すると、有効な SSML ドキュメントを作成しながら正しく表示できます。たとえば、次の文は、

```
We're using the lawyer at Peabody & Chambers, attorneys-at-law.
```

SSML で次のようにレンダリングされます

```
<speak>
We&apos;re using the lawyer at Peabody &amp; Chambers, attorneys-at-law.
</speak>
```

この場合、アポストロフィとアンパサンドの特殊文字はエスケープされ、SSML ドキュメントは有効なままです。

&、<、および > 記号については、SSML を使用するときに常にエスケープコードが必要です。さらに、アポストロフィ/一重引用符 (') をアポストロフィとして使用する場合も、エスケープコードを使用する必要があります。

ただし、二重引用符 (") またはアポストロフィ/一重引用符 (') を引用符として使用する場合、エスケープコードを使用するかどうかはコンテキストによって異なります。

二重引用符

- 属性値が二重引用符で区切られている場合はエスケープする必要があります。例えば、次の AWS CLI コードでは

```
--text "Pete &quot;Maverick&quot; Mitchell"
```

- テキストコンテキストではエスケープする必要はありません。次のコードについて考えてみます。

```
He said, "Turn right at the corner."
```

- 属性値が一重引用符で区切られている場合は、エスケープする必要はありません。例えば、以下の AWS CLI コードは、

```
--text 'Pete "Maverick" Mitchell'
```

一重引用符

- アポストロフィとして使用する場合は、エスケープする必要があります。次のコードについて考えてみます。

```
We&apos;ve got to leave quickly.
```

- テキストコンテキストではエスケープする必要はありません。次のコードについて考えてみます。

```
"And then I said, 'Don't quote me.'"
```

- コード属性を二重引用符で区切ってエスケープする必要はありません。例えば、以下の AWS CLI コードは、

```
--text "Pete 'Maverick' Mitchell"
```

コンソールでの SSML の使用

SSML タグを使用すると、発音、音量、ピッチ、話す速度など、音声のさまざまな要素をカスタマイズして制御できます。では AWS Management Console、音声に変換する SSML 拡張テキストがテキスト読み上げページの SSML タブに入力されます。プレーンテキストで入力されるテキストは、選択した言語と音声のデフォルト設定によって異なりますが、SSML 拡張テキストでは、単なる音声だけではなく、話し方まで Amazon Polly に伝えられます。追加された SSML タグを除き、Amazon Polly は、プレーンテキストの場合と同様に SSML 拡張テキストを合成します。詳細については、「ステップ 1.2: コンソールで音声をプレーンテキスト入力と合成する」を参照してください。

SSML を使用する場合は、SSML を使用していることが Amazon Polly に分かるように、テキスト全体を <speak> タグで囲みます。例:

```
<speak>Hi! My name is Joanna. I will read any text you type here.</speak>
```

次に、<speak> 内のテキストに特定の SSML タグを使用して、テキストの発音方法をカスタマイズします。一時停止の追加やスピーチの速度変更、音声のボリュームの強弱など、テキストの音声が適切になるように、さまざまなカスタマイズを幅広く行うことができます。使用可能な SSML タグの詳細なリストについては、「サポートされている SSML タグ」を参照してください。

次の例では、SSML タグを使用して、短い段落を読み上げるときに「World Wide Web Consortium」を「W3C」に置き換えるように Amazon Polly に指示します。また、タグを使用して単語の一時停止およびウィスパーを導入します。この演習の結果を「コンソールでのレキシコンの適用 (音声合成)」の結果と比較します。

SSML の詳細と例については、「サポートされている SSML タグ」を参照してください。

SSML 拡張テキストから音声を合成するには (コンソール)

1. にサインイン AWS Management Console し、https://console.aws.amazon.com/polly/ で Amazon Polly コンソールを開きます。
2. 表示されていない場合は、[Text-to-Speech (テキスト読み上げ機能)] タブを選択します。
3. [SSML] を有効にします。

4. テキストボックスに次のテキストを入力するか貼り付けます。

```
<speak>
     He was caught up in the game.<break time="1s"/> In the middle of the
     10/3/2014 <sub alias="World Wide Web Consortium">W3C</sub> meeting,
     he shouted, "Nice job!" quite loudly. When his boss stared at him, he
 repeated
     <amazon:effect name="whispered">"Nice job,"</amazon:effect> in a
     whisper.
</speak>
```

SSML タグは Amazon Polly にテキストのレンダリング方法を指示します。

- `<break time="1s"/>` は、最初の 2 行の文章間で 1 秒間停止するように Amazon Polly に指示します。
- `<sub alias="World Wide Web Consortium">W3C</sub>` は、略語 W3C を World Wide Web Consortium に置き換えるように Amazon Polly に指示します。
- `<amazon:effect name="whispered">Nice job</amazon:effect>` は、「Nice job」の 2 番目のインスタンスをウィスパーするように Amazon Polly に指示します。

> **Note**
>
> を使用する場合は AWS CLI、入力テキストを引用符で囲み、周囲のコードと区別します。Amazon Polly コンソールにはコードは表示されないため、コンソールを使用する場合は入力テキストを引用符で囲みません。

5. [言語] で、[米国 (英語)] を選択し、音声を選択します。
6. 音声を聞くには、[聴く] を選択します。
7. 音声ファイルを保存するには、[ダウンロード] を選択します。別の形式で保存するには、[詳細設定] を展開して [音声ファイル形式の設定] を有効にし、目的の形式を選択してから、[ダウンロード] を選択します。

での SSML の使用 AWS CLI

を使用して SSML AWS CLI 入力テキストを合成できます。次の例では、 AWS CLIを使用して一般的なタスクを実行する方法を示します。

トピック

- Synthesize-Speech コマンドでの SSML の使用
- SSML 拡張ドキュメントの合成
- 一般的な Amazon Polly タスクに SSML を使用する

Synthesize-Speech コマンドでの SSML の使用

この例では、SSML 文字列を指定して、synthesize-speech コマンドを使用する方法を示しています。synthesize-speech コマンドを使用する場合、通常、以下を指定します。

- 入力テキスト (必須)
- 開始と終了のタグ (必須)
- 出力形式
- 音声

この例では、引用符と必須の開始と終了の <speak></speak> タグで簡単なテキスト文字列を指定します。

> **Important**
>
> Amazon Polly コンソールでは入力テキストの前後に引用符は必要ありませんが、 AWS CLI では使用する必要があります。また、入力テキストを囲む引用符と個々のタグで必要な引用符を区別することも重要です。
> たとえば、標準引用符 (") で入力テキストを囲み、一重引用符 (') を内部タグに使用できます。逆に使用することもできます。どちらのオプションを選んでも、Unix、Linux、および macOS 用に使用できます。ただし、Windows では、入力テキストは標準引用符で囲み、タグには一重引用符を使用する必要があります。
> すべてのオペレーティングシステムで、標準引用符 (") で入力テキストを囲み、一重引用符 (') を内部タグに使用できます。逆に使用することもできます。) 例:
>
> ```
> --text "<speak>Hello <break time='300ms'/> World</speak>"
> ```
>
> Unix、Linux、および macOS では、逆に一重引用符 (') で入力テキストを囲み、標準引用符 (") を内部タグに使用することもできます。
>
> ```
> --text '<speak>Hello <break time="300ms"/> World</speak>'
> ```

次の AWS CLI 例は、Unix、Linux、macOS 用にフォーマットされています。Windows の場合は、各行末のバックスラッシュ (\) Unix 連結文字をカレット (^) に置き換え、入力テキストは二重引用符 (") で囲み、内部タグは一重引用符 (') で囲みます。

```
aws polly synthesize-speech \
--text-type ssml \
--text '<speak>Hello world</speak>' \
--output-format mp3 \
--voice-id Joanna \
speech.mp3
```

合成された音声を聞くには、音声プレイヤーを使用して、生成された speech.mp3 ファイルを再生します。

SSML 拡張ドキュメントの合成

入力テキストが長い場合は、ファイルに SSML の内容を保存し、synthesize-speech コマンドでそのファイル名を指定する方が簡単です。たとえば、以下の内容を example.xml というファイルに保存します。

```
<?xml version="1.0"?>
<speak version="1.1"
       xmlns="http://www.w3.org/2001/10/synthesis"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:schemaLocation="http://www.w3.org/2001/10/synthesis http://www.w3.org/TR/
speech-synthesis11/synthesis.xsd"
       xml:lang="en-US">Hello World</speak>
```

xml:lang 属性は、入力テキストの言語として en-US (米国英語) を指定します。入力テキストの言語と選択されたボイスの言語が SynthesizeSpeech オペレーションに与える影響については、「外字の発音の改善」を参照してください。

SSML 拡張ファイルを実行するには

1. SSML をファイル (example.xml など) に保存します。
2. XML ファイルが保存されているパスから次の synthesize-speech コマンドを実行します。入力テキストの file:\\example.xml を置き換えて SSML ファイルを入力として指定しま

す。このコマンドは、実際に入力テキストを追加するのではなく、ファイルの場所を指定しているため、引用符は不要です。

> **Note**
>
> 次の AWS CLI 例は、Unix、Linux、macOS 用にフォーマットされています。Windows の場合は、各行末のバックスラッシュ (\) Unix 連結文字をキャレット (^) に置き換えてください。

```
aws polly synthesize-speech \
--text-type ssml \
--text file://example.xml \
--output-format mp3 \
--voice-id Joanna \
speech.mp3
```

3. 合成された音声を聞くには、音声プレイヤーを使用して、生成された speech.mp3 ファイルを再生します。

一般的な Amazon Polly タスクに SSML を使用する

次の例では、SSML タグを使用して一般的な Amazon Polly タスクを完了する方法を説明します。その他の SSML タグについては、「サポートされている SSML タグ」を参照してください。

次の例をテストするには、適切な SSML 拡張テキストを指定して次の synthesize-speech コマンドを使用します。

次の AWS CLI 例は、Unix、Linux、macOS 用にフォーマットされています。Windows の場合は、各行末のバックスラッシュ (\) Unix 連結文字をカレット (^) に置き換え、入力テキストは二重引用符 (") で囲み、内部タグは一重引用符 (') で囲みます。

```
aws polly synthesize-speech \
--text-type ssml \
--text '<speak>Hello <break time="300ms"/> World</speak>' \
--output-format mp3 \
--voice-id Joanna \
speech.mp3
```

一時停止を追加する

単語間に一時停止を追加するには、<break> 要素を使用します。次の SSML synthesize-speech コマンドは、<break> 要素を使用して、「Hello」と「World」の間に 300 ミリ秒のディレイを追加します。

```
<speak>
     Hello <break time="300ms"/> World.
</speak>
```

ボリューム、ピッチ、速度の制御

ピッチ、話す速度、話す音量を制御するには、<prosody> 要素を使用します。

- 次の synthesize-speech は <prosody> 要素を使用して音量を制御します。

```
<speak>
     <prosody volume="+20dB">Hello world</prosody>
</speak>
```

- 次の synthesize-speech コマンドは <prosody>要素を使用してピッチを制御します。

```
<speak>
     <prosody pitch="x-high">Hello world.</prosody>
</speak>
```

- 次の synthesize-speech コマンドは <prosody> 要素を使用して話す速度を指定します。

```
<speak>
     <prosody rate="x-fast">Hello world.</prosody>
</speak>
```

- 次の例に示すように、<prosody> 要素に複数の属性を指定できます。

```
<speak>
     <prosody volume="x-loud" pitch="x-high" rate="x-fast">Hello world.</prosody>
</speak>
```

ウィスパー

単語をささやくには、<amazon:effect name="whispered"> 要素を使用します。次の例では、<amazon:effect name="whispered"> 要素を使用して「little lamb」をウィスパー音声にするように Amazon Polly に指示します。

```
<speak>
     Mary has a <amazon:effect name="whispered">little lamb.</amazon:effect>
</speak>
```

この効果を高めるには、<prosody> 要素を使用して、ウィスパー音声をわずかに遅くします。

単語を強調する

単語または語句を強調するには、<emphasis> 要素を使用します。

```
<speak>
     <emphasis level="strong">Hello</emphasis> world how are you?
</speak>
```

特定の単語の発話方法の指定

読み上げられるテキストの種類に関する情報を提供するには、<say-as> 要素を使用します。

たとえば、次の SSML <say-as> は、テキスト 4/6 を日付として解釈する必要があることを示します。属性 interpret-as="date" format="dm" は、月/日の形式の日付として発声する必要があることを示します。

また、<say-as> 要素は、Amazon Polly に数字を分数、電話番号、測定単位などとして発声するよう指示するために使用することもできます。

```
<speak>
     Today is <say-as interpret-as="date" format="md" >4/6</say-as>
</speak>
```

生成される音声は「Today is June 4th」となります。<say-as> タグは、interpret-as 属性で追加コンテキストを指定することで、そのテキストをどのように解釈するかを説明します。

合成された音声を確認するには、生成された speech.mp3 ファイルを再生します。

この要素の詳細については、「特殊なタイプの単語の発声方法を制御する」を参照してください。

外字の発音の改善

Amazon Polly では、入力テキストは選択されたボイスと同じ言語であることを前提としています。入力テキスト内の外国語の発音を向上させるには、synthesize-speech 呼び出しで、xml:lang 属性を使用して対象言語を指定します。これにより、Amazon Polly はタグ付けされた外国語に対し異なる発音ルールを適用します。

次の例では、入力テキストの言語のさまざまな組み合わせを使用して、外国語の音声と発を指定する方法を説明します。使用可能な言語の完全なリストについては、「Amazon Polly の言語」を参照してください。

次の例では、音声 (Joanna) は米国英語の音声です。デフォルトでは、Amazon Polly では、入力テキストは選択されたボイスと同じ言語 (この場合は米国英語) であることを前提としています。xml:lang タグを使用すると、Amazon Polly はテキストをスペイン語として解釈し、テキストは外国語の発音ルールに基づいて選択した音声がスペイン語を発音する場合のように発音されます。このタグがない場合、テキストは選択された音声の発音ルールを使用して発声されます。

```
<speak>
     That restaurant is terrific. <lang xml:lang="es-ES">Mucho gusto.</lang>
</speak>
```

入力テキストの言語は英語のため、Amazon Polly は生成されたスペイン語音素をもっとも近い英語音素にマッピングします。その結果、Joanna は、スペイン語を正しく発音しているが米国英語のアクセントがある米国英語のネイティブスピーカーとして発声します。

> **Note**
>
> より似通っている言語同士の組み合わせであれば、他の組み合わせよりうまく機能します。

サポートされている SSML タグ

Amazon Polly は、次の SSML タグをサポートしています。

アクション	SSML タグ	ニューラル音声での可用性	ロングフォーム音声での可用性	生成音声での可用性
一時停止を追加する	<break>	完全に利用可能	完全に利用可能	完全に利用可能
単語を強調する	<emphasis>	利用不可	利用不可	利用不可
特定の単語に別の言語を指定する	<lang>	完全に利用可能	完全に利用可能	完全に利用可能
テキストにカスタムタグを配置する	<mark>	完全に利用可能	完全に利用可能	完全に利用可能
段落間に一時停止を追加する	<p>	完全に利用可能	完全に利用可能	完全に利用可能
発音記号を使用する	<phoneme>	完全に利用可能	完全に利用可能	利用不可
音量、話す速度、ピッチを制御する	<prosody>	部分的に利用可能	部分的に利用可能	利用不可
合成音声の最大時間を設定する	<prosody amazon:max-duration>	利用不可	利用不可	利用不可
文章間に一時停止を追加する	<s>	完全に利用可能	完全に利用可能	完全に利用可能
特殊なタイプの単語の発声方法を制御する	<say-as>	部分的に利用可能	部分的に利用可能	部分的に利用可能
SSML 拡張テキストを特定する	<speak>	完全に利用可能	完全に利用可能	完全に利用可能
頭字語や略語を発音する	<sub>	完全に利用可能	完全に利用可能	完全に利用可能

アクション	SSML タグ	ニューラル音声での可用性	ロングフォーム音声での可用性	生成音声での可用性
品詞を指定して発音を向上させる	<w>	完全に利用可能	完全に利用可能	完全に利用可能
呼吸音を追加する	<amazon:auto-breaths>	利用不可	利用不可	利用不可
ニュースキャスターの話し方	<amazon:domain name=news>	ニューラル音声のみを選択	利用不可	利用不可
ダイナミックレンジ圧縮を追加する	<amazon:effect name="drc">	完全に利用可能	完全に利用可能	利用不可
柔らかく発声する	<amazon:effect phonation="soft">	利用不可	利用不可	利用不可
声質を制御する	<Amazon: 効果 vocal-tract-length >	利用不可	利用不可	利用不可
ウィスパー	<amazon:effect name="whispered">	利用不可	利用不可	利用不可

> **Note**
>
> サポートされていない SSML タグを標準形式、ニューラル形式、またはロングフォーム形式で使用すると、エラーが発生します。

SSML 拡張テキストを特定する

<speak>

このタグは、生成形式、ロングフォーム形式、ニューラル形式、標準 TTS 形式でサポートされています。

<speak> タグは、すべての Amazon Polly SSML テキストのルート要素です。SSML 拡張テキストはすべて <speak> タグで囲まれている必要があります。

```
<speak>Mary had a little lamb.</speak>
```

一時停止を追加する

<break>

このタグは、生成形式、ロングフォーム形式、ニューラル形式、標準 TTS 形式でサポートされています。

テキストに一時停止を追加するには、<break> タグを使用します。強度に応じて一時停止を設定することができます (カンマ、文章、段落後の一時停止と同等)。または、特定の時間 (秒またはミリ秒単位) に設定できます。一時停止の長さを指定する属性を指定しない場合、Amazon Polly はデフォルト値である `<break strength="medium"/>` を使用します。この場合は、カンマの一時停止の長さが追加されます。

`strength` 属性値:

- none: 一時停止しません。none は、ピリオドの後などに通常発生する一時停止を削除する場合に使用します。
- `x-weak`: none と同じ強度であり、一時停止しません。
- `weak`: カンマ後と同じ長さの一時停止を設定します。
- `medium`: weak と同じ強度です。
- `strong`: 文の後と同じ長さの一時停止を設定します。
- `x-strong`: 段落後と同じ長さの一時停止を設定します。

`time` 属性値:

- *`[number]`*`s`: 一時停止の継続時間 (秒)。最大期間は `10s` です。
- *`[number]`*`ms`: 一時停止の時間 (ミリ秒)。最大期間は `10000ms` です。

例:

```
<speak>
     Mary had a little lamb <break time="3s"/>Whose fleece was white as snow.
```

```
</speak>
```

break タグの属性を使用しない場合、結果はテキストによって異なります。

- break タグの横に他に句読点がない場合は、<break strength="medium"/> (カンマの長さの一時停止) が作成されます。
- タグがカンマの横にある場合は、タグは <break strength="strong"/> (文の長さの一時停止) にアップグレードされます。
- タグがピリオドの横にある場合は、タグは <break strength="x-strong"/> (段落の長さの一時停止) にアップグレードされます。

単語を強調する

<emphasis>

このタグは、標準の TTS 形式でのみサポートされています。

単語を強調するには、<emphasis> タグを使用します。単語を強調すると、話す速度と音量が変わります。強調を大きくするほど、Amazon Polly はテキストを大きくゆっくり発声します。強調を小さくするほど、小さく速く発声します。強調の度合いを指定するには、level 属性を使用します。

level 属性値:

- Strong: 音量が大きくなり、速度が低下するため、読み上げが大きくて遅くなります。
- Moderate: 音量が大きくなり、速度が低下しますが、strong ほどではありません。Moderate がデフォルトです。
- Reduced: 音量が小さくなり、速度が上がります。発声が柔らかくて速くなります。

> **Note**
>
> 音声の通常の読み上げ速度およびボリュームは、moderate レベルから reduced レベルまでの間に収まります。

例:

```
<speak>
```

```
    I already told you I <emphasis level="strong">really like</emphasis> that person.
</speak>
```

特定の単語に別の言語を指定する

<lang>

このタグは、生成形式、ロングフォーム形式、ニューラル形式、標準 TTS 形式でサポートされています。

<lang> タグを使用して、特定の単語、語句、または文に別の言語を指定します。<lang> タグ内に囲うと、一般的に外国語の語句の発声がよくなります。言語を指定するには、xml:lang 属性を使用します。使用可能な言語の完全なリストについては、「Amazon Polly の言語」を参照してください。

<lang> タグを適用しない限り、入力テキストのすべての単語は、voice-id で指定された音声の言語で発声されます。<lang> タグを適用すると、その単語はその言語で発声されます。

例えば、voice-id が Joanna (米国英語の話者) の場合、Amazon Polly は以下をフランス語のアクセントを使用せずに Joanna の音声で発声します。

```
<speak>
    Je ne parle pas français.
</speak>
```

<lang> タグを使用して Joanna の音声を使用すると、Amazon Polly は文章をアメリカのアクセントがあるフランス語として発声します。

```
<speak>
    <lang xml:lang="fr-FR">Je ne parle pas français.</lang>.
</speak>
```

Joanna はネイティブのフランス語音声ではないため、発音は、ネイティブである米国英語に基づきます。たとえば、français という単語の完ぺきなフランス語の発音では口蓋垂ふるえ音の /R/ ですが、Joanna の米国英語の発音では、この音素に対応する /r/ が使用されます。

イタリア語を話す Giorgio の voice-id を使用する場合、次のテキストでは、Amazon Polly は Giorgio の音声でイタリア語発音を使用して文を発声します。

```
<speak>
     Mi piace Bruce Springsteen.
</speak>
```

同じ音声を以下の <lang> タグとともに使用すると、Amazon Polly は Bruce Springsteen をイタリア語のアクセントがある英語で発音します。

```
<speak>
     Mi piace <lang xml:lang="en-US">Bruce Springsteen.</lang>
</speak>
```

このタグは、音声を合成するときにオプションの DefaultLangCodeオプションの代わりに使用することもできます。ただし、これを行うには、テキストを SSML を使用してフォーマットする必要があります。

テキストにカスタムタグを配置する

<mark>

このタグは、生成形式、ロングフォーム形式、ニューラル形式、標準 TTS 形式でサポートされています。

テキスト内にカスタムタグを配置するには、<mark> タグを使用します。Amazon Polly はタグに対してアクションを行いませんが、SSML メタデータ内でタグの位置を返します。このタグは、次の形式を保持している限り、任意に呼び出すことができます。

```
<mark name="tag_name"/>
```

たとえば、タグの名前が「animal」の場合、入力テキストは次のようになります。

```
<speak>
     Mary had a little <mark name="animal"/>lamb.
</speak>
```

Amazon Polly により、次の SSML メタデータが返されます。

```
{"time":767,"type":"ssml","start":25,"end":46,"value":"animal"}
```

段落間に一時停止を追加する

<p>

このタグは、生成形式、ロングフォーム形式、ニューラル形式、標準 TTS 形式でサポートされています。

テキストの段落間に一時停止を追加するには、<p> タグを使用します。このタグを使用すると、通常ネイティブスピーカーがカンマや文章の終わりで一時停止するよりも長く一時停止できます。<p> タグを使用して段落を囲みます。

```
<speak>
     <p>This is the first paragraph. There should be a pause after this text is
 spoken.</p>
     <p>This is the second paragraph.</p>
</speak>
```

これは <break strength="x-strong"/> を使用した一時停止の指定と同等です。

発音記号を使用する

<phoneme>

このタグは、ロングフォーム形式、ニューラル形式、および標準の TTS 形式でサポートされています。

Amazon Polly で特定のテキストに発音記号を使用するには、<phoneme> タグを使用します。

<phoneme> タグには、次の 2 つの属性が必要です。これらは、Amazon Polly が使用する音声記号と訂正された発音の発音記号を示します。

- `alphabet`
  - `ipa`— 国際音声記号 (IPA) が使用されることを表します。
  - `x-sampa`— 拡張 SAM 音声記号 (X-SAMPA) システムが使用されることを表します。
- ph
  - 発音の発音記号を指定します。詳細については、サポートされている言語の音素およびビゼームテーブルを参照してください。

<phoneme> タグを使用すると、Amazon Polly は選択されたボイスが使用する言語にデフォルトで関連付けられた標準の発音ではなく、ph 属性で指定された発音を使用します。

たとえば、「pecan」という単語には、発音方法が 2 種類あります。次の例では、「pecan」という単語には、異なる発音が各行に割り当てられています。Amazon Polly は、デフォルトの発音ではなく、ph 属性で指定されたとおりに pecan を発音します。

国際音声記号 (IPA)

```
<speak>
     You say, <phoneme alphabet="ipa" ph="p##k##n">pecan</phoneme>.
     I say, <phoneme alphabet="ipa" ph="#pi.kæn">pecan</phoneme>.
</speak>
```

拡張 SAM 音声記号 (X-SAMPA)

```
<speak>
     You say, <phoneme alphabet='x-sampa' ph='pI"kA:n'>pecan</phoneme>.
     I say, <phoneme alphabet='x-sampa' ph='"pi.k{n'>pecan</phoneme>.
</speak>
```

標準中国語では、音声発音に拼音が使用されます。

拼音

```
<speak>
     ## <phoneme alphabet="x-amazon-pinyin" ph="bo2">#</phoneme>#
     ## <phoneme alphabet="x-amazon-pinyin" ph="bao2">#</phoneme>#
</speak>
```

日本語では読み仮名と発音仮名を使います。

読み仮名

```
<speak>
     ###<phoneme alphabet="x-amazon-yomigana" ph="####">##</phoneme>###
     ###<phoneme alphabet="x-amazon-yomigana" ph="####">##</phoneme>###
     ###<phoneme alphabet="x-amazon-yomigana" ph="Hirokazu">##</phoneme>###
</speak>
```

発音仮名

```
<speak>
     ###<phoneme alphabet="x-amazon-pron-kana" ph="##'##">##</phoneme>###
</speak>
```

音量、話す速度、ピッチを制御する

<prosody>

Prosody タグ属性は、標準の TTS 音声によって完全にサポートされています。ニューラル音声と口ングフォーム音声は、volume および rate 属性をサポートしますが、pitch 属性はサポートしません。

選択した音声の音量、速度、ピッチを制御するには、prosody タグを使用します。

音量、ピッチ、話す速度は、選択された特定の音声に依存します。異なる言語の音声間で異なるだけではなく、同じ言語を話す個々の音声にも違いがあります。このため、属性はすべての言語でもほぼ共通していますが、言語によって明らかな差異があり、絶対値はありません。

prosody タグには 3 つの属性が含まれており、それぞれに属性を設定する複数の値があります。各属性には同じ構文が使用されます。

```
<prosody attribute="value"></prosody>
```

- volume
  - default: ボリュームを現在の音声のデフォルトレベルにリセットします。
  - silent、x-soft、soft、medium、loud、x-loud: ボリュームを、現在の音声の事前定義された値に設定します。
  - +ndB、-ndB: 現在の音量レベルと比較して音量を変更します。値 +0dB は、ボリュームを変更しない、+6dB は、現在のボリュームの約 2 倍、-6dB は、現在のボリュームの約半分であることを表します。

  たとえば、次のように文章に音量を設定します。

  ```
  <speak>
       Sometimes it can be useful to <prosody volume="loud">increase the volume
       for a specific speech.</prosody>
  </speak>
  ```

または、次のように設定することもできます。

```
<speak>
     And sometimes a lower volume <prosody volume="-6dB">is a more effective way of
     interacting with your audience.</prosody>
</speak>
```

- `rate`
  - `x-slow`、`slow`、`medium`、`fast`、`x-fast`: ピッチを選択音声用に事前定義した値に設定します。
  - n%: 話す速度を正のパーセンテージで変更します。たとえば、100% では話す速度は変更されず、200% の値は話す速度がデフォルトの速度の 2 倍の速度、50% の値はデフォルトの話す速度の半分の速度を意味します。この値の範囲は 20 ~ 200% です。

  たとえば、次のように文章に話す速度を設定します。

```
<speak>
     For dramatic purposes, you might wish to <prosody rate="slow">slow up the
 speaking
     rate of your text.</prosody>
</speak>
```

  または、次のように設定することもできます。

```
<speak>
     Although in some cases, it might help your audience to <prosody rate="85%">slow
     the speaking rate slightly to aid in comprehension.</prosody>
</speak>
```

- `pitch`
  - `default`: ピッチを現在の音声のデフォルトレベルにリセットします。
  - `x-low`、`low`、`medium`、`high`、`x-high`: ピッチを現在音声用に事前定義した値に設定します。
  - `+n%` または `-n%`: 相対パーセンテージを使用してピッチを調整します。たとえば、値 `+0%` は、ベースラインピッチを変更しない、`+5%` は、ベースラインピッチを少し上げる、`-5%` は、ベースラインピッチを少し下げることを意味します。

  たとえば、次のように文章にピッチを設定します。

```
<speak>
     Do you like sythesized speech <prosody pitch="high">with a pitch that is higher
     than normal?</prosody>
</speak>
```

または、次のように設定することもできます。

```
<speak>
     Or do you prefer your speech <prosody pitch="-10%">with a somewhat lower pitch?
</prosody>
</speak>
```

<prosody> タグには属性が少なくとも 1 つ含まれている必要がありますが、同じタグ内に複数含めることができます。

```
<speak>
     Each morning when I wake up, <prosody volume="loud" rate="x-slow">I speak
     quite slowly and deliberately until I have my coffee.</prosody>
</speak>
```

次のように、ネストされたタグを組み合わせることもできます。

```
<speak>
     <prosody rate="85%">Sometimes combining attributes <prosody pitch="-10%">can
     change the impression your audience has of a voice</prosody> as well.</prosody>

</speak>
```

合成音声の最大時間を設定する

<prosody amazon:max-duration>

このタグは現在、標準の TTS 形式でのみサポートされています。

音声が合成されるときの時間を制御するには、<prosody> 属性を持つ amazon:max-duration タグを使用します。

選択した音声により、合成された音声の時間はわずかに異なります。これにより、合成音声を正確なタイミングを必要とするビジュアルやその他のアクティビティと一致させることが難しくなります。

特定のフレーズが異なる言語で大きく異なることがあるため、この問題は、翻訳アプリケーションによって拡大されています。

<prosody amazon:max-duration> タグは合成された音声の時間 (長さ) と一致します。

このタグでは次の構文を使用します。

```
<prosody amazon:max-duration="time duration">
```

<prosody amazon:max-duration> タグを使用して、長さを秒またはミリ秒で指定できます。

- *n*s。最大時間 (秒)。
- *n*ms。最大時間 (ミリ秒)。

たとえば、次の音声テキストの最大時間は 2 秒です。

```
<speak>
     <prosody amazon:max-duration="2s">
          Human speech is a powerful way to communicate.
     </prosody>
</speak>
```

タグ内に配置されたテキストは、指定された時間を超えません。選択した音声または言語が通常その時間より長くかかる場合は、Amazon Polly は指定された時間に収まるように音声の速度を上げます。

指定された時間が、通常のレートでテキストを読み上げる時間より長い場合、Amazon Polly は通常通りに音声を読み上げます。音声の速度を下げたり、無音が追加されることはありませんので、生成される音声はリクエストされたものよりも短くなります。

> **Note**
>
> Amazon Polly は通常のレートの 5 倍以上に速度を上げません。これより速くテキストを読み上げることには、通常意味がありません。最大の速さまで速度を上げても指定された時間内に収まらない場合は、速度が上げられますが、音声は指定された時間よりも長くなります。

<prosody amazon:max-duration> タグ内に 1 つの文または複数の文を含めることができ、テキスト内で複数の <prosody amazon:max-duration> タグを使用できます。

例:

```
<speak>
     <prosody amazon:max-duration="2400ms">
        Human speech is a powerful way to communicate.
     </prosody>
     <break strength="strong"/>
     <prosody amazon:max-duration="5100ms">
        Even a simple ‘Hello’ can convey a lot of information depending on the pitch,
 intonation, and tempo.
     </prosody>
     <break strength="strong"/>
     <prosody amazon:max-duration="8900ms">
        We naturally understand this information, which is why speech is ideal for
 creating applications where
        a screen isn’t practical or possible, or simply isn’t convenient.
     </prosody>
</speak>
```

<prosody amazon:max-duration> タグを使用して、Amazon Polly が合成された音声を返す際のレイテンシーを向上させることができます。レイテンシーの度合いは、文章およびその長さによって異なります。比較的短い文章で構成されたテキストを使用することをお勧めします。

制約事項

<prosody amazon:max-duration> タグの使用方法、および他の SSML タグがどのように動作するかの両方で、それぞれ制限があります。

- <prosody amazon:max-duration> タグ内のテキストは 1500 文字より長くすることはできません。
- <prosody amazon:max-duration> タグを入れ子にすることはできません。別のタグの内部に <prosody amazon:max-duration> タグを配置すると、Amazon Polly では内側のタグが無視されます。

  たとえば、次の場合には、<prosody amazon:max-duration="5s"> タグが無視されます。

```
<speak>
     <prosody amazon:max-duration="16s">
          Human speech is a powerful way to communicate.
```

```
          <prosody amazon:max-duration="5s">
               Even a simple ‘Hello’ can convey a lot of information depending on the
 pitch, intonation, and tempo.
          </prosody>

          We naturally understand this information, which is why speech is ideal for
 creating applications where a screen isn’t practical or possible, or simply isn’t
 convenient.
     </prosody>
</speak>
```

- <prosody> タグ内で、rate タグの <prosody amazon:max-duration> 属性を使用することはできません。これは、両方がテキストを読み上げる際の速度に影響するためです。

  例えば、次の例で Amazon Polly では、<prosody rate="2"> タグが無視されます。

```
<speak>
     <prosody amazon:max-duration="7500ms">
          Human speech is a powerful way to communicate.

          <prosody rate="2">
               Even a simple ‘Hello’ can convey a lot of information depending on the
 pitch, intonation, and tempo.
          </prosody>
     </prosody>
</speak>
```

一時停止および **max-duration**

max-duration タグを使用する場合は、テキスト内に一時停止を挿入できます。ただし、Amazon Polly は、音声の最大時間を計算する際に一時停止の長さを含めます。さらに、Amazon Polly では、文章内にカンマおよびピリオドがある場合に発生する短い一時停止が保持され、最長時間に含められます。

たとえば、次のブロックでは、8 秒内の音声で、600 ミリ秒の休止と、カンマおよびピリオドが原因で発生する休止が発生します。

```
<speak>
     <prosody amazon:max-duration="8s">
          Human speech is a powerful way to communicate.
```

```
          <break time="600ms"/>
          Even a simple ‘Hello’ can convey a lot of information depending on the pitch,
 intonation, and tempo.
     </prosody>
</speak>
```

文章間に一時停止を追加する

<s>

このタグは、生成形式、ロングフォーム形式、ニューラル形式、標準 TTS 形式でサポートされています。

テキストの行または文章間に一時停止を追加するには、<s> タグを使用します。このタグは、以下と同じ効果があります。

- ピリオド (.) を使用した文の終了
- <break strength="strong"/> による一時停止の指定

<break> タグとは異なり、<s> タグは文を囲みます。これは、文ではなく、詩のように複数の行で編成された音声合成に便利です。

次の例では、<s> タグは、1 文目と 2 文目の後に、それぞれ短い一時停止を作成します。最後の文には、<s> タグはありませんが、ピリオドで終わっているため、その後に短い一時停止があります。

```
<speak>
     <s>Mary had a little lamb</s>
     <s>Whose fleece was white as snow</s>
     And everywhere that Mary went, the lamb was sure to go.
</speak>
```

特殊なタイプの単語の発声方法を制御する

<say-as>

characters オプションを除き、 <say-as> タグは生成形式、ロングフォーム形式、ニューラル形式、標準 TTS 形式でサポートされています。Amazon Polly で二ューラル音声を使用し、実行時に

characters オプションが設定された <say-as> タグが発生した場合、影響を受ける文は関連する標準音声を使用して合成されます。ただし、影響を受けた文は、ニューラル音声を使用しているかのように請求されます。

interpret-as 属性を指定した <say-as> タグを使用して、特定の文字、単語、および数字を発声する方法を Amazon Polly に指示します。これにより、コンテキストを追加して、Amazon Polly でのテキストのレンダリングのあいまいさを排除できます。

<say-as> タグは interpret-as という 1 つの属性を使用します。この属性にはさまざまな値を使用できます。それぞれには同じ構文が使用されます。

```
<say-as interpret-as="value">[text to be interpreted]</say-as>
```

次の値を interpret-as で使用できます。

- characters または spell-out: のように、テキストの各文字をスペルアウトします a-b-c。

> **Note**
>
> このオプションは現在、ニューラル音声に対してはサポートされていません。ニューラル音声を使用していて、この SSML コードが実行時に Amazon Polly によって発生した場合、影響を受ける文は関連する標準音声を使用して合成されます。ただし、この文は、それでもニューラル音声を使用している場合と同様に課金されることに注意してください。

- cardinal または number: 数値テキストを基数 (例: 1,234) として解釈します。
- ordinal: 数値テキストを序数 (例: 1,234 番目) として解釈します。
- digits: 各桁を個別 (例: 1-2-3-4) にスペルアウトします。
- fraction: 数値テキストを小数として解釈します。この属性は、分数 (例: 3/20) と帯分数 (例: 2½) のいずれにも適用されます。詳細については、以下を参照してください。
- unit: 数値テキストを測定値として解釈します。この値は、数字または小数の後に単位 (間にスペースは不要) を続けるか (例: 1/2inch)、単位のみが続きます (例: 1meter)。
- date: 日付としてテキストを解釈します。日付の形式は、形式属性で指定する必要があります。詳細については、以下を参照してください。
- time: 分単位および秒単位の時間として数値テキストを解釈します (例: 1'21")。
- address: 住所の一部としてテキストを解釈します。
- expletive: タグ内に含まれるコンテンツを「ピー音」で消します。

- telephone: 7 桁または 10 桁の電話番号として数値テキストを解釈します (例: 2025551212)。この値を使用して、内線電話も処理できます (例: 2025551212x345)。詳細については、以下を参照してください。

> **Note**
>
> 現在、telephone オプションはすべての言語で使用できるわけではありません。英語変種 (en-AU、en-GB、en-IN、en-US、en-GB-WLS)、スペイン語変種 (es-ES、es-MX、es-US)、フランス語変種 (fr-FR および fr-CA)、ポルトガル語変種 (pt-BR および pt-PT)、ドイツ語 (de-DE)、イタリア語 (it-IT)、日本語 (ja-JP)、ロシア語 (ru-RU) の音声では使用できます。また、場合によっては、アラビア語 (arb) などの言語が電話番号として設定された番号を自動的に処理するため、実際に telephone SSML タグを実装しないことに注意してください。

分数

Amazon Polly は interpret-as="fraction" 属性を持つ say-as タグ内の値を一般的な分数として解釈します。次に、分数の構文を示します。

- 分数

  構文: ##/## (2/9 など)。

  例: <say-as interpret-as="fraction">2/9</say-as> は「two ninths」と発音されます。
- 負でない混合数字

  構文: ## + ##/## (例: 3+1/2)。

  例: <say-as interpret-as="fraction">3+1/2</say-as> は「three and a half」と発音されます。

> **Note**
>
> 「3」と「1/2」の間に + を入れる必要があります。Amazon Polly では、+ を含まない混合数字 (例: 3 1/2) はサポートされていません。

日付

interpret-as が date に設定されている場合は、上記と合わせて、日付の形式を示す必要があります。

次の構文を使用します。

```
<say-as interpret-as="date" format="format">[date]</say-as>
```

例:

```
<speak>
     I was born on <say-as interpret-as="date" format="mdy">12-31-1900</say-as>.
</speak>
```

以下の形式は、date 属性で使用できます。

- mdy: Month-day-year。
- dmy: D ay-month-year。
- ymd: Y ear-month-day。
- md: 月-日。
- dm: 日-月。
- ym: 年-月。
- my: 月-年。
- d: 日。
- m: 月。
- y: 年。
- yyyymmdd: Y ear-month-day。この形式を使用する場合、疑問符を使用して Amazon Polly に日付の部分をスキップさせることができます。

  例えば、Amazon Polly は、以下を「9 月 22 日」としてレンダリングします。

  ```
  <say-as interpret-as="date">????0922</say-as>
  ```

  Format は必要ありません。

電話

Amazon Polly では、<say-as> タグが使用されていなくても、テキストの形式に基づいて正しく入力されたテキストとして解釈するよう試みます。例えば、テキストに「202-555-1212」が含まれている場合、Amazon Polly はこれを 10 桁の電話番号として解釈し、各桁を個別に読み上げ、ダッシュ部分には短い一時停止を挟みます。この場合、<say-as interpret-as="telephone"> を使用する必要はありません。ただし、「2025551212」と入力し、Amazon Polly に電話番号として読み上げさせたい場合は、<say-as interpret-as="telephone"> を指定する必要があります。

各要素を解釈するロジックは、言語固有です。たとえば、米国英語と英国英語は、電話番号の発音が異なります (英国英語では、同一数字の連続はまとめて発音される。「double five」や「triple four」など)。違いを確認するには、次の例をアメリカの音声とイギリスの音声でテストします。

```
<speak>
     Richard's number is <say-as interpret-as="telephone">2122241555</say-as>
</speak>
```

頭字語や略語を発音する

<sub>

このタグは、生成形式、ロングフォーム形式、ニューラル形式、標準 TTS 形式でサポートされています。

<sub> 属性を指定した alias タグを使用して、頭字語や略語など指定のテキストを別の単語 (または発音) に置き換えます。

次の構文を使用します。

```
<sub alias="new word">abbreviation</sub>
```

次の例では、「Mercury」という名前が要素の化学記号に置き換えられ、音声コンテンツがより明確になります。

```
<speak>
     My favorite chemical element is <sub alias="Mercury">Hg</sub>, because it looks so
 shiny.
</speak>
```

品詞を指定して発音を向上させる

<w>

このタグは、生成形式、ロングフォーム形式、ニューラル形式、標準 TTS 形式でサポートされています。

<w> タグを使用して単語の発音をカスタマイズするには、単語の品詞または別の意味を指定します。その際、role 属性を使用します。

このタグでは次の構文を使用します。

```
<w role="attribute">text</w>
```

以下の値は、role 属性に使用できます。

品詞を指定するには:

- amazon:VB: 動詞 (現在形) として語句を解釈します。
- amazon:VBD: 動詞の過去形として語句を解釈します。
- amazon:DT: 限定詞として語句を解釈します。
- amazon:IN: 単語を前置詞として解釈します。
- amazon:JJ: 形容詞として語句を解釈します。
- amazon:NN: 名詞として語句を解釈します。

たとえば、品詞で解釈する場合、米国英語の「read」という単語の発音はタグに応じて変わります。

```
<speak>
     The word <say-as interpret-as="characters">read</say-as> may be interpreted
     as either the present simple form <w role="amazon:VB">read</w>, or the past
     participle form <w role="amazon:VBD">read</w>.
</speak>
```

特定の意味を指定するには

- amazon:DEFAULT: 単語のデフォルトの意味を使用します。
- amazon:SENSE_1: 単語のデフォルト以外の意味を使用します。たとえば、「bass」という名詞の発音は、その意味に応じて異なります。この単語のデフォルトの意味は、音楽的な範囲の低音

部を表します。別の意味に「淡水魚」があり、「bass」と呼ばれますが、発音は異なります。<w role="amazon:SENSE_1">bass</w> を使用すると、デフォルトでない発音 (淡水魚) をレンダリングして音声テキストを作成します。

次のように合成すると、その発音と意味の違いを確認できます。

```
<speak>
    Depending on your meaning, the word <say-as interpret-as="characters">bass</say-
as>
    may be interpreted as either a musical element: bass, or as its alternative
 meaning,
    a freshwater fish <w role="amazon:SENSE_1">bass</w>.
</speak>
```

> **Note**
>
> 言語によっては、サポートされる品詞が異なる場合があります。

呼吸音を追加する

<amazon:breath> および <amazon:auto-breaths>

このタグは、標準の TTS 形式でのみサポートされています。

自然な音声には、正しく発話された単語と呼吸音の両方が含まれています。合成音声に呼吸音を追加することで、自然音に近づけることができます。<amazon:breath> タグと <amazon:auto-breaths> タグは呼吸音を提供します。次のオプションがあります。

- 手動モード: テキスト内の呼吸音の場所、長さ、ボリュームを設定します。
- 自動モード: Amazon Polly で音声出力に呼吸音を自動的に挿入します。
- 混合モード: お客様と Amazon Polly の両方が呼吸音を追加します。

手動モード

手動モードでは、入力テキスト内の呼吸音の挿入先に <amazon:breath/> タグを配置します。呼吸音の長さとボリュームは、duration 属性と volume 属性でそれぞれカスタマイズできます。

- duration: 呼吸音の長さを制御します。有効な値は、default、x-short、short、medium、long、x-long です。デフォルト値は、mediumです。
- volume: 呼吸音の音量を制御します。有効な値は、default、x-soft、soft、medium、loud、x-loud です。デフォルト値は、mediumです。

> **Note**
>
> 各属性値の正確な長さとボリュームは、使用する特定の Amazon Polly 音声に依存します。

デフォルト値を使用して呼吸音を設定するには、属性を設定しないで <amazon:breath/> を使用します。

たとえば、属性を使用して呼吸音の長さとボリュームを medium に設定するには、次のように属性を設定します。

```
<speak>
     Sometimes you want to insert only <amazon:breath duration="medium" volume="x-
loud"/>a single breath.
</speak>
```

デフォルト値を使用するには、次のタグのみを使用します。

```
<speak>
     Sometimes you need <amazon:breath/>to insert one or more average breaths
 <amazon:breath/> so that the
     text sounds correct.
</speak>
```

複数の個別の呼吸音を節内に追加するには、次のように設定します。

```
<speak>
     <amazon:breath duration="long" volume="x-loud"/> <prosody rate="120%"> <prosody
 volume="loud">
     Wow! <amazon:breath duration="long" volume="loud"/> </prosody> That was quite
 fast. <amazon:breath
     duration="medium" volume="x-loud"/> I almost beat my personal best time on this
 track. </prosody>
</speak>
```

自動モード

自動モードでは、<amazon:auto-breaths> タグを使用して、適切な間隔で呼吸音を自動的に作成するように Amazon Polly に指示します。間隔の頻度、ボリューム、および長さを設定できます。自動の呼吸音の適用先であるテキストの先頭に </amazon:auto-breaths> タグを配置し、最後に終了タグを配置します。

> **Note**
>
> 手動モードのタグである <amazon:breath/> とは異なり、<amazon:auto-breaths> タグには終了タグ (</amazon:auto-breaths>) が必要です。

<amazon:auto-breaths> タグでは、以下のオプションの属性を使用できます。

- volume: 呼吸音の音量を制御します。有効な値は、default、x-soft、soft、medium、loud、x-loud です。デフォルト値は、mediumです。
- frequency: テキスト内で呼吸音を発生させる回数を制御します。有効な値は、default、x-low、low、medium、high、x-high です。デフォルト値は、mediumです。
- duration: 呼吸音の長さを制御します。有効な値は、default、x-short、short、medium、long、x-long です。デフォルト値は、mediumです。

デフォルトでは、呼吸音の頻度は入力テキストに依存します。ただし、通常、呼吸音はカンマやピリオドの後に発生します。

以下の例では、<amazon:auto-breaths> タグの使用方法を示します。コンテンツに使用するオプションを決定するには、該当する例を Amazon Polly コンソールにコピーし、差異を聞き比べます。

- オプションのパラメータを設定しない自動モードの使用。

```
<speak>
     <amazon:auto-breaths>Amazon Polly is a service that turns text into lifelike
 speech,
     allowing you to create applications that talk and build entirely new categories
 of speech-
     enabled products. Amazon Polly is a text-to-speech service that uses advanced
 deep learning
     technologies to synthesize speech that sounds like a human voice. With dozens of
 lifelike
```

```
    voices across a variety of languages, you can select the ideal voice and build
 speech-
    enabled applications that work in many different countries.</amazon:auto-
breaths>
</speak>
```

- ボリュームを制御した自動モードの使用。指定しないパラメータ (duration と frequency) は、デフォルト値 (medium) に設定されます。

```
<speak>
    <amazon:auto-breaths volume="x-soft">Amazon Polly is a service that turns text
 into lifelike
    speech, allowing you to create applications that talk and build entirely new
 categories of
    speech-enabled products. Amazon Polly is a text-to-speech service, that uses
 advanced deep
    learning technologies to synthesize speech that sounds like a human voice. With
 dozens of
    lifelike voices across a variety of languages, you can select the ideal voice
 and build speech-
    enabled applications that work in many different countries.</amazon:auto-
breaths>
</speak>
```

- 頻度を制御した自動モードの使用。指定しないパラメータ (duration と volume) は、デフォルト値 (medium) に設定されます。

```
<speak>
    <amazon:auto-breaths frequency="x-low">Amazon Polly is a service that turns text
 into lifelike
    speech, allowing you to create applications that talk and build entirely new
 categories of
    speech-enabled products. Amazon Polly is a text-to-speech service, that uses
 advanced deep
    learning technologies to synthesize speech that sounds like a human voice. With
 dozens of
    lifelike voices across a variety of languages, you can select the ideal voice
 and build speech-
    enabled applications that work in many different countries.</amazon:auto-
breaths>
</speak>
```

- 複数のパラメータを設定した自動モードの使用。指定しない Duration パラメータは、Amazon Polly でデフォルト値 (medium) に設定されます。

```
<speak>
     <amazon:auto-breaths volume="x-loud" frequency="x-low">Amazon Polly is a service
 that turns
     text into lifelike speech, allowing you to create applications that talk and
 build entirely new
     categories of speech-enabled products. Amazon Polly is a text-to-speech service,
 that uses
     advanced deep learning technologies to synthesize speech that sounds like a
 human voice. With
     dozens of lifelike voices across a variety of languages, you can select the
 ideal voice and build
     speech-enabled applications that work in many different countries.</amazon:auto-
breaths>
</speak>
```

ニュースキャスターの話し方

<amazon:domain name=news>

ニュースキャスタースタイルは、アメリカ英語 (en-US) の Matthew または Joanna の音声、スペイン語 (米国) (es-ES) の Lupe の音声、およびイギリス英語の (en-GB) の Amy の音声で利用できます。このスタイルは Neural 形式を使用する場合にのみサポートされます。

ニュースキャスタースタイルを使用するには、SSML タグと次の構文を使用します。

```
<amazon:domain name="news">text</amazon:domain>
```

例えば、以下のようにニュースキャスタースタイルを Amy の音声で利用できます。

```
<speak>
<amazon:domain name="news">
From the Tuesday, April 16th, 1912 edition of The Guardian newspaper:

The maiden voyage of the White Star liner Titanic, the largest ship ever launched, has
 ended in disaster.
```

```
The Titanic started her trip from Southampton for New York on Wednesday. Late on Sunday
 night she struck
an iceberg off the Grand Banks of Newfoundland. By wireless telegraphy she sent out
 signals of distress,
and several liners were near enough to catch and respond to the call.
</amazon:domain>
</speak>
```

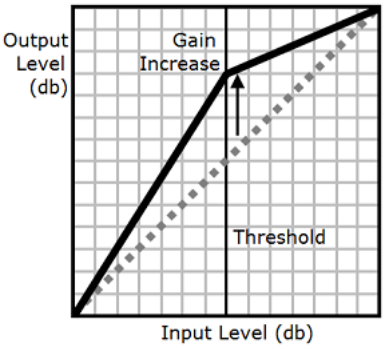
ダイナミックレンジ圧縮を追加する

<amazon:effect name="drc">

このタグは、ロングフォーム形式、ニューラル形式、および標準の TTS 形式でサポートされています。

オーディオファイルで使用されるテキスト、言語、および音声に応じて、音はソフトなものから大音量なものまでさまざまです。移動する車両の音などの環境音は、しばしばよりソフトな音を遮蔽することがあり、それによってオーディオトラックがはっきりと聞こえにくくなります。オーディオファイルの特定の音量を上げるには、ダイナミックレンジ圧縮 (drc) タグを使用します。

drc タグは、オーディオのミッドレンジの「ラウドネス」しきい値を設定し、そのしきい値前後のサウンドの音量 (ゲイン) を上げます。これは、しきい値に最も近いゲインを最大にして、しきい値から遠いゲインを最小にします。



これにより、ノイズの多い環境でミドルレンジのサウンドが聞き取りやすくなり、オーディオファイル全体がより明確になります。

drc タグはブールパラメータです (存在するかどうか)。これは <amazon:effect name="drc"> 構文を使用し、</amazon:effect> で閉じます。

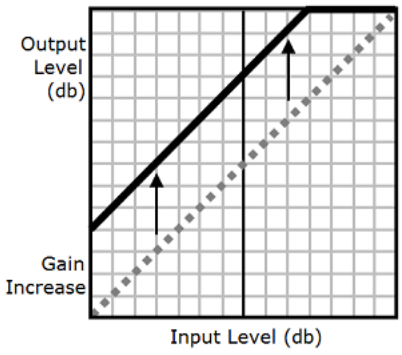
drc タグは、Amazon Polly でサポートされている任意の音声または言語で使用できます。録音のセクション全体に適用することも、数語だけに適用することもできます。例:

```
<speak>
     Some audio is difficult to hear in a moving vehicle, but <amazon:effect
 name="drc"> this audio
     is less difficult to hear in a moving vehicle.</amazon:effect>
</speak>
```

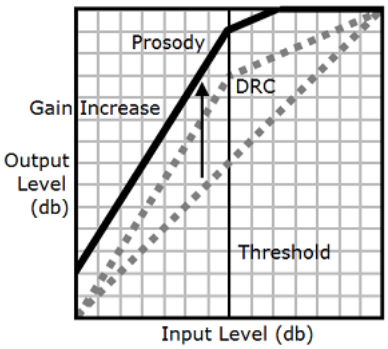
> **Note**
>
> 構文で「drc」を使用する場合、大文字と小文字が区別されます。amazon:effect

drc を prosody volume タグで使用する

次の図に示すように、prosody volume タグは、オーディオファイル全体の音量を元のレベル (点線) から調整レベル (実線) に均等に上げます。ファイルの特定の部分の音量をさらに上げるには、drc タグを prosody volume タグとともに使用します。タグを組み合わせても prosody volume タグの設定には影響しません。



drc タグと prosody volume タグを一緒に使用すると、Amazon Polly は drc タグを最初に適用し、中音域 (しきい値付近) の音量を上げます。次に、prosody volume タグを適用し、オーディオトラック全体の音量をさらに上げます。



タグを一緒に使用するには、一方を他方の内側に入れます。例:

```
<speak>
     <prosody volume="loud">This text needs to be understandable and loud.
 <amazon:effect name="drc">
     This text also needs to be more understandable in a moving car.</amazon:effect></
prosody>
</speak>
```

このテキストでは、prosody volume タグは、全体の音量を「大音量」に上げます。drc タグは、2 番目の文の中音域の音量を上げます。

> **Note**
>
> drc タグと prosody volumeタグを一緒に使用する場合は、入れ子タグに標準の XML プラクティスを使用します。

柔らかく発声する

<amazon:effect phonation="soft">

このタグは現在、標準の TTS 形式でのみサポートされています。

入力テキストを softer-than-normal 音声で話すように指定するには、<amazon:effect phonation "soft"> タグを使用します。

次の構文を使用します。

```
<amazon:effect phonation="soft">text</amazon:effect>
```

たとえば、Matthew の音声でこのタグを次のように使用してみます。

```
<speak>
     This is Matthew speaking in my normal voice. <amazon:effect phonation="soft">This
     is Matthew speaking in my softer voice.</amazon:effect>
</speak>
```

声質を制御する

<Amazon: 効果 vocal-tract-length >

このタグは現在、標準の TTS 形式でのみサポートされています。

声質 (Timbre) は、同じ音程とラウドネスを持っている場合でも、音声の違いを伝えるのに役立つ音声の音質です。音声の音色に影響する最も重要な生理機能の 1 つは、声道の長さです。声道は、声帯上部から唇の端までにおよぶ空気の通り道です。

Amazon Polly の出力音声の音色を制御するには、vocal-tract-length タグを使用します。このタグには、話し手の声道の長さを変更して、話し手の声の大きさが変化したように聞こえる効果があります。vocal-tract-length を大きくすると、話し手の声が物理的に大きく聞こえます。このタグを小さくすると、話し手の声も小さく聞こえます。このタグは、Amazon Polly のテキスト読み上げポートフォリオのいずれの声にも使用できます。

声質を変更するには、次の値を使用します。

- +n% または -n%: 現在の声で、相対割合 (%) の変更により声道の長さを調整します。たとえば、+4% または -2% などです。有効な値の範囲は +100% ~ -50% です。この範囲外の値は切り捨てられます。たとえば、+111% は +100% のように聞こえ、-60% は -50% のように聞こえます。
- n%: 声道の長さを現在の声の声道の長さの絶対割合値 (%) に調整します。たとえば、110% または 75% などです。110% の絶対値は +10% の相対値に相当します。絶対値 100% は、現在の声のデフォルト値と同じです。

次の例は、声帯の長さを変更して音色を変更する方法を示しています。

```
<speak>
     This is my original voice, without any modifications. <amazon:effect vocal-tract-
length="+15%">
     Now, imagine that I am much bigger. </amazon:effect> <amazon:effect vocal-tract-
length="-15%">
     Or, perhaps you prefer my voice when I'm very small. </amazon:effect> You can also
 control the
     timbre of my voice by making minor adjustments. <amazon:effect vocal-tract-
length="+10%">
     For example, by making me sound just a little bigger. </
amazon:effect><amazon:effect
     vocal-tract-length="-10%"> Or, making me sound only somewhat smaller. </
amazon:effect>
</speak>
```

複数のタグの組み合わせ

vocal-tract-length タグは、Amazon Polly でサポートされている他の SSML タグと組み合わせることができます。声質 (声道の長さ) とピッチが密接に結びついているので、vocal-tract-length タグと <prosody pitch> タグの両方を使用すると、最良の結果が得られます。最もリアルな音声を生成するために、2 つのタグの変化のパーセンテージを使用することをお勧めします。さまざまな組み合わせを試して、必要な結果を得ます。

次の例は、タグを結合する方法を示しています。

```
<speak>
     The pitch and timbre of a person's voice are connected in human speech.
     <amazon:effect vocal-tract-length="-15%"> If you are going to reduce the vocal
 tract length,
     </amazon:effect><amazon:effect vocal-tract-length="-15%"> <prosody pitch="+20%">
 you
     might consider increasing the pitch, too. </prosody></amazon:effect>
     <amazon:effect vocal-tract-length="+15%"> If you choose to lengthen the vocal
 tract,
     </amazon:effect> <amazon:effect vocal-tract-length="+15%"> <prosody pitch="-10%">
     you might also want to lower the pitch. </prosody></amazon:effect>
</speak>
```

ウィスパー

<amazon:effect name="whispered">

このタグは現在、標準の TTS 形式でのみサポートされています。

このタグは、入力テキストを通常の音声ではなく、ささやき声で読み上げることを表します。このタグは、Amazon Polly のテキスト読み上げポートフォリオのいずれの声にも適用できます。

次の構文を使用します。

```
<amazon:effect name="whispered">text</amazon:effect>
```

例:

```
<speak>
     <amazon:effect name="whispered">If you make any noise, </amazon:effect>
     she said, <amazon:effect name="whispered">they will hear us.</amazon:effect>
</speak>
```

この場合、合成された音声はささやき声で読み上げられますが、「she said」というフレーズは、選択された Amazon Polly 音声の通常の合成音声で発声されます。

必要に応じて、話速を最大 10% 低下させることで、「ささやき」効果を強めることができます。

例:

```
<speak>
     When any voice is made to whisper, <amazon:effect name="whispered">
     <prosody rate="-10%">the sound is slower and quieter than normal speech
     </prosody></amazon:effect>
</speak>
```

ささやかれた音声のスピーチマークを生成するときは、音声ストリームには、スピーチマークが音声ストリームに一致するよう、ささやかれた音声を含める必要があります。

レキシコンの管理

発音レキシコンを使用すると、単語の発音をカスタマイズできます。Amazon Polly には、レキシコンを AWS リージョンに保存するために使用できる API オペレーションが用意されています。これらのレキシコンは、その特定のリージョンに固有です。SynthesizeSpeech オペレーションを使用してテキストを合成するとき、そのリージョンから 1 つ以上のレキシコンを使用できます。これにより、合成が始まる前に、入力テキストに指定されたレキシコンが適用されます。詳細については、「SynthesizeSpeech」を参照してください。

> **Note**
>
> これらのレキシコンは、Pronunciation Lexicon Specification (PLS) の W3C 推奨事項に準拠する必要があります。詳細については、W3C ウェブサイトの「Pronunciation Lexicon Specification (PLS) バージョン 1.0」を参照してください。

以下は、音声合成エンジンでレキシコンを使用する方法の例です。

- 一般的な単語は、「g3t sm4rt」 (get smart) のように文字を数字に置き換えたスタイルで書かれる場合があります。人間はこれらの単語を正しく読むことができます。ただし、テキスト読み上げ機能 (TTS) エンジンでは文字をそのまま読み上げ、名前をスペルどおりに発音します。このような場合、Amazon Polly でレキシコンを活用し、合成された音声をカスタマイズできます。この例では、レキシコンで「g3t sm4rt」という単語に「get smart」というエイリアスを指定できます。
- テキスト内に W3C などの頭文字が使用されている場合があります。レキシコンを使用して W3C のエイリアスを定義すると、W3C は完全な形の「World Wide Web Consortium」に置き換えられて読み上げられます。

レキシコンにより、選択した言語で一般的でない単語を Amazon Polly が発音する方法をさらにコントロールできます。たとえば、音声記号を使用して発音を指定できます。詳細については、W3C ウェブサイトの「Pronunciation Lexicon Specification (PLS) バージョン 1.0」を参照してください。

トピック

- 複数のレキシコンの適用
- Amazon Polly コンソールでのレキシコンの管理
- でのレキシコンの管理 AWS CLI

複数のレキシコンの適用

テキストに最大 5 つのレキシコンを適用できます。テキストに適用する複数のレキシコンに同じ書記素がある場合、適用順序により音声の結果が異なります。例えば、「Hello, my name is Bob」というテキストがあり、異なるレキシコンで 2 つの語彙素が同じ書記素 Bob を使用しているとします。

LexA

```
<lexeme>
   <grapheme>Bob</grapheme>
   <alias>Robert</alias>
</lexeme>
```

LexB

```
<lexeme>
   <grapheme>Bob</grapheme>
   <alias>Bobby</alias>
</lexeme>
```

レキシコンが LexA の後に LexB という順番になっている場合、合成された音声は、「Hello, my name is Robert.」となります。LexB の後に LexA という順番になっている場合、合成された音声は、「Hello, my name is Bobby.」となります。

Example – LexB の前に LexA を適用する

```
aws polly synthesize-speech \
--lexicon-names LexA LexB \
--output-format mp3 \
--text 'Hello, my name is Bob' \
--voice-id Justin \
bobAB.mp3
```

音声出力:「Hello, my name is Robert.」

Example – LexA の前に LexB を適用する

```
aws polly synthesize-speech \
--lexicon-names LexB LexA \
--output-format mp3 \
--text 'Hello, my name is Bob' \
```

```
--voice-id Justin \
bobBA.mp3
```

音声出力:「Hello, my name is Bobby.」

Amazon Polly コンソールを使用したレキシコンの適用の詳細については、コンソールでのレキシコンの適用 (音声合成) を参照してください。

Amazon Polly コンソールでのレキシコンの管理

Amazon Polly コンソールを使用して、レキシコンのアップロード、ダウンロード、適用、フィルタリング、および削除ができます。以下の手順で、これらの各プロセスを示します。

コンソールでのレキシコンのアップロード

発音レキシコンを使用するには、最初にアップロードする必要があります。コンソールにはレキシコンをアップロードできる場所が 2 つあります。[Text-to-Speech] タブ、および [Lexicons] タブです。

以下のプロセスでは、選択した言語で一般的でない語句がどのように発音されるかをカスタマイズするのに使用できるレキシコンを追加する方法を説明します。

レキシコンを [Lexicons] タブから追加するには

1. にサインイン AWS Management Console し、https://console.aws.amazon.com/polly/ で Amazon Polly コンソールを開きます。
2. [Lexicons] タブを選択します。
3. [Upload lexicon] (レキシコンのアップロード) を選択します。
4. レキシコンの名前を入力し、[Choose a lexicon file] (レキシコンファイルの選択) を使用して、アップロードするレキシコンを見つけます。アップロードできるのは、.pls または.xml 拡張子を持つ PLS ファイルのみです。
5. [Upload lexicon] (レキシコンのアップロード) を選択します。同じ名前でレキシコンが (.pls または .xml ファイルにかかわりなく) 既に存在する場合、レキシコンをアップロードすると既存のレキシコンは上書きされます。

text-to-Speech タブからレキシコンを追加するには

1. にサインイン AWS Management Console し、https://console.aws.amazon.com/polly/ で Amazon Polly コンソールを開きます。

2. [Text-to-Speech (テキスト読み上げ機能)] タブを選択します。
3. [Additional settings] (詳細設定) を展開し、[Customize pronunciation] (発音のカスタマイズ) を有効にしてから、[Upload lexicon] (レキシコンのアップロード) を選択します。
4. レキシコンの名前を入力し、[Choose a lexicon file] (レキシコンファイルの選択) を使用して、アップロードするレキシコンを見つけます。.pls および .xml 拡張子を持つ PLS ファイルのみ使用できます。
5. [Upload lexicon] (レキシコンのアップロード) を選択します。同じ名前のレキシコンが (.pls または .xml ファイルにかかわりなく) 既に存在する場合、レキシコンをアップロードすると既存のレキシコンは上書きされます。

コンソールでのレキシコンの適用 (音声合成)

次の手順では、「W3C」を「World Wide Web Consortium」に置き換える `W3c.pls` レキシコンを適用することで、入力テキストにレキシコンを適用する方法を示します。テキストに複数のレキシコンを適用する場合、最初の一致が後の一致より優先する上から下の順序で適用されます。レキシコンで指定した言語が選択した言語と同じ場合、レキシコンはテキストにのみ適用されます。

レキシコンはプレーンテキストまたは SSML 入力に適用できます。

Example – W3C.pls レキシコンを適用する

この演習に必要なレキシコンを作成するには、PutLexicon オペレーションの使用 を参照してください。トピックの一番上に示されている W3C.pls レキシコンを作成するには、空のテキストファイルを使用します。このファイルを保存する場所を忘れないでください。

W3C.pls レキシコンを入力に適用するには

この例では、「W3C」を「World Wide Web Consortium」に置き換えるレキシコンを説明します。この実習結果を、英語と別の言語の両方で「コンソールでの SSML の使用」の例と比較してみてください。

1. にサインイン AWS Management Console し、https://console.aws.amazon.com/polly/ で Amazon Polly コンソールを開きます。
2. 次のいずれかを行います。

   - [SSML] を無効にし、テキスト入力ボックスにこのテキストを入力するか貼り付けます。

```
He was caught up in the game.
```

```
In the middle of the 10/3/2014 W3C meeting
he shouted, "Score!" quite loudly.
```

- [SSML] を有効にし、テキスト入力ボックスにこのテキストを入力するか貼り付けます。

```
<speak>He wasn't paying attention.<break time="1s"/>
In the middle of the 10/3/2014 W3C meeting
he shouted, "Score!" quite loudly.</speak>
```

3. [Language] (言語) リストから [English US] を選択し、このテキストに使用する音声を選択します。
4. [Additional settings] (詳細設定) を展開し、[Customize pronunciation] (発音のカスタマイズ) を有効にします。
5. レキシコンのリストから、`W3C (English, US)` を選択します。

   `W3C (English, US)` レキシコンがリストにない場合、[Upload lexicon] を選択してアップロードし、リストから選択します。このレキシコンの作成については、「PutLexicon オペレーションの使用」を参照してください。
6. 音声をすぐに聞くには、[Listen] (聴く) を選択します。
7. 音声をファイルに保存するには、

   a. [ダウンロード] を選択します。

   b. 別のファイル形式に変更するには、[Speech file format settings] (音声ファイル形式の設定) を有効にし、目的のファイル形式を選択して、[Download] (ダウンロード) を選択します。

前の手順を繰り返しますが、異なる言語を選択し、出力の違いを確認してください。

コンソールでのレキシコンリストのフィルタリング

次の手順では、選択した言語のレキシコンのみが表示されるようにレキシコンのリストをフィルタリングする方法を説明します。

言語ごとにリスト表示されるレキシコンをフィルタリングするには

1. にサインイン AWS Management Console し、https://console.aws.amazon.com/polly/ で Amazon Polly コンソールを開きます。
2. [Lexicons] タブを選択します。
3. [Any language] (任意の言語) を選択します。

4. 言語のリストから、フィルタリングする言語を選択します。

   選択した言語のレキシコンのみがリストに表示されます。

コンソールでのレキシコンのダウンロード

以下のプロセスでは、1 つ以上のレキシコンをダウンロードする方法を説明します。ファイル内のレキシコンエントリを追加、削除、または変更し、再度アップロードしてレキシコン を保持できます up-to-date。

1 つ以上のレキシコンをダウンロードするには

1. にサインイン AWS Management Console し、https://console.aws.amazon.com/polly/ で Amazon Polly コンソールを開きます。
2. [Lexicons] タブを選択します。
3. ダウンロードするレキシコンを 1 つ、または複数選択します。
   a. 1 つのレキシコンをダウンロードするには、リストから名前を選択します。
   b. 単一の圧縮アーカイブファイルとして複数のレキシコンをダウンロードするには、リストにあるダウンロードする各エントリの横にあるチェックボックスを選択します。
4. [ダウンロード] を選択します。
5. レキシコンをダウンロードするフォルダを開きます。
6. [保存] を選択します。

コンソールでのレキシコンの削除

レキシコンを削除するには

以下のプロセスでは、レキシコンを削除する方法を説明します。レキシコンを削除した後は、もう一度レキシコンを使用する前に再度追加する必要があります。個々のレキシコンの横にあるチェックボックスを選択して、1 つ以上のレキシコンを同時に削除できます。

1. にサインイン AWS Management Console し、https://console.aws.amazon.com/polly/ で Amazon Polly コンソールを開きます。
2. [Lexicons] タブを選択します。
3. 削除するレキシコンをリストから 1 つ以上選択します。

4. [削除] を選択します。
5. 確認テキストを入力し、リージョンからレキシコンを削除するには [Delete] (削除) を、そのままにするには [Cancel] (キャンセル) を選択します。

でのレキシコンの管理 AWS CLI

以下のトピックでは、発音レキシコンの管理に必要な AWS CLI コマンドについて説明します。

トピック

- PutLexicon オペレーションの使用
- GetLexicon オペレーションの使用
- ListLexicons オペレーションの使用
- DeleteLexicon オペレーションの使用

PutLexicon オペレーションの使用

Amazon Polly では、 PutLexiconを使用して、アカウントの特定の AWS リージョンに発音レキシコンを保存できます。次に、サービスがテキストを合成し始める前に適用する SynthesizeSpeech リクエストに保存した 1 つ以上のレキシコンを指定できます。詳細については、「レキシコンの管理」を参照してください。

このセクションでは、レキシコンの例と、レキシコンの保存とテスト step-by-step の手順について説明します。

> **Note**
>
> これらのレキシコンは、Pronunciation Lexicon Specification (PLS) の W3C 推奨事項に準拠する必要があります。詳細については、W3C ウェブサイトの「Pronunciation Lexicon Specification (PLS) バージョン 1.0」を参照してください。

例 1: 1 つの lexeme を持つレキシコン

次の W3C PLS 準拠のレキシコンについて考えます。

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<lexicon version="1.0"
      xmlns="http://www.w3.org/2005/01/pronunciation-lexicon"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://www.w3.org/2005/01/pronunciation-lexicon
        http://www.w3.org/TR/2007/CR-pronunciation-lexicon-20071212/pls.xsd"
      alphabet="ipa"
      xml:lang="en-US">
  <lexeme>
    <grapheme>W3C</grapheme>
    <alias>World Wide Web Consortium</alias>
  </lexeme>
</lexicon>
```

次の点に注意してください。

- <lexicon> 要素で指定された 2 つの属性:
  - xml:lang 属性は、レキシコンが適用される言語コード en-US を指定します。Amazon Polly は、SynthesizeSpeech コールで指定された音声が同じ言語コード (en-US) であれば、この例のレキシコンを使用できます。

    > **Note**
    >
    > 音声に関連付けられている言語コードを探すため、DescribeVoices オペレーションを使用できます。

  - alphabet 属性は IPA を指定します。つまり、国際音声記号 (IPA) のアルファベットが発音に使用されます。IPA は、発音を表記するためのアルファベットの 1 つです。Amazon Polly は、Extended Speech Assessment Methods Phonetic Alphabet (X-SAMPA) もサポートしています。

- <lexeme> 要素は、<grapheme>(単語のテキスト表記) と <alias> の間のマッピングを説明します。

このレキシコンをテストするには、以下を実行します。

1. example.pls という名前でレキシコンを保存します。

2. put-lexicon AWS CLI コマンドを実行して、レキシコン (という名前w3c) を us-east-2 リージョンに保存します。

```
aws polly put-lexicon \
--name w3c \
--content file://example.pls
```

3. synthesize-speech コマンドを実行してサンプルテキストを音声ストリーム (speech.mp3) に合成し、オプションの lexicon-name パラメーターを指定します。

```
aws polly synthesize-speech \
--text 'W3C is a Consortium' \
--voice-id Joanna \
--output-format mp3 \
--lexicon-names="w3c" \
speech.mp3
```

4. 結果の speech.mp3 を再生し、テキスト内の W3C という単語が World Wide Web Consortium に置き換わっていることを確認します。

前の例のレキシコンでは、エイリアスを使用します。レキシコンで説明されている IPA のアルファベットは使用されません。以下のレキシコンでは、IPA のアルファベットで <phoneme> 要素を使用して音声発音を指定します。

```
<?xml version="1.0" encoding="UTF-8"?>
<lexicon version="1.0"
      xmlns="http://www.w3.org/2005/01/pronunciation-lexicon"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://www.w3.org/2005/01/pronunciation-lexicon
        http://www.w3.org/TR/2007/CR-pronunciation-lexicon-20071212/pls.xsd"
      alphabet="ipa"
      xml:lang="en-US">
  <lexeme>
    <grapheme>pecan</grapheme>
    <phoneme>p##k##n</phoneme>
  </lexeme>
</lexicon>
```

このレキシコンをテストするため、同じ手順に従います。「pecan」 という単語がある入力テキストを必ず指定します (例「Pecan pie is delicious」)。

例 2: 複数の lexemes を持つレキシコン

この例では、レキシコンで指定する語彙素は合成の入力テキストのみに適用されます。次のレキシコンについて考えます。

```
<?xml version="1.0" encoding="UTF-8"?>
<lexicon version="1.0"
      xmlns="http://www.w3.org/2005/01/pronunciation-lexicon"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://www.w3.org/2005/01/pronunciation-lexicon
        http://www.w3.org/TR/2007/CR-pronunciation-lexicon-20071212/pls.xsd"
      alphabet="ipa" xml:lang="en-US">

  <lexeme>
    <grapheme>W3C</grapheme>
    <alias>World Wide Web Consortium</alias>
  </lexeme>
  <lexeme>
    <grapheme>W3C</grapheme>
    <alias>WWW Consortium</alias>
  </lexeme>
  <lexeme>
    <grapheme>Consortium</grapheme>
    <alias>Community</alias>
  </lexeme>
</lexicon>
```

レキシコンが 3 つの語彙素を指定し、そのうち 2 つは書記素 W3C のエイリアスを次のとおり定義します。

- 最初の <lexeme> 要素はエイリアス (World Wide Web Consortium) を定義します。
- 2 つめの <lexeme> は、代替エイリアス (WWW Consortium) を定義します。

Amazon Polly は、レキシコンにあるどの書記素にも最初の置き換えを使用します。

3 つめの <lexeme> は、Consortium という単語に対する置き換え (Community) を定義します。

まず、このレキシコンをテストします。次のサンプルテキストを音声ファイル (speech.mp3) に合成する場合、SynthesizeSpeech への呼び出しでレキシコンを指定します。

```
The W3C is a Consortium
```

SynthesizeSpeech は、まずレキシコンを次のように適用します。

- 最初の語彙素では、W3C という単語は World Wide Web Consortium に変更されます。変更したテキストは次のように表示されます。

```
The World Wide Web Consortium is a Consortium
```

- 3 番目の語彙素で定義したエイリアスは、元のテキストの一部にある Consortium という単語にのみ適用され、以下のテキストの結果となります。

```
The World Wide Web Consortium is a Community.
```

これは、 AWS CLI 次のように を使用してテストできます。

1. example.pls という名前でレキシコンを保存します。
2. us-east-2 リージョンに w3c の名前でレキシコンを保存するため put-lexicon コマンドを実行します。

```
aws polly put-lexicon \
--name w3c \
--content file://example.pls
```

3. w3c レキシコンは返されたレキシコンのリストにあることを確認するために、list-lexicons コマンドを実行します。

```
aws polly list-lexicons
```

4. synthesize-speech コマンドを実行してサンプルテキストを音声ファイル (speech.mp3) に合成し、オプションの lexicon-name パラメーターを指定します。

```
aws polly synthesize-speech \
--text 'W3C is a Consortium' \
--voice-id Joanna \
--output-format mp3 \
--lexicon-names="w3c" \
speech.mp3
```

5. 生成された speech.mp3 ファイルを再生して、合成された音声がテキストの変更を反映していることを確認します。

例 3: 複数のレキシコンを指定する

SynthesizeSpeech への呼び出しでは、複数のレキシコンを指定できます。この場合、最初に指定したレキシコン (左から右の順序) により、他のすべてのレキシコンがオーバーライドされます。

次の 2 つのレキシコンを考えます。各レキシコンは、同じ書記素 W3C の異なるエイリアスを示していることに注意してください。

- レキシコン 1: w3c.pls

```
<?xml version="1.0" encoding="UTF-8"?>
<lexicon version="1.0"
      xmlns="http://www.w3.org/2005/01/pronunciation-lexicon"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://www.w3.org/2005/01/pronunciation-lexicon
        http://www.w3.org/TR/2007/CR-pronunciation-lexicon-20071212/pls.xsd"
      alphabet="ipa" xml:lang="en-US">
  <lexeme>
    <grapheme>W3C</grapheme>
    <alias>World Wide Web Consortium</alias>
  </lexeme>
</lexicon>
```

- レキシコン 2: w3cAlternate.pls

```
<?xml version="1.0" encoding="UTF-8"?>
<lexicon version="1.0"
      xmlns="http://www.w3.org/2005/01/pronunciation-lexicon"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://www.w3.org/2005/01/pronunciation-lexicon
        http://www.w3.org/TR/2007/CR-pronunciation-lexicon-20071212/pls.xsd"
      alphabet="ipa" xml:lang="en-US">

  <lexeme>
    <grapheme>W3C</grapheme>
    <alias>WWW Consortium</alias>
  </lexeme>
</lexicon>
```

これらのレキシコンを w3c および w3cAlternate としてそれぞれ保存するとします。w3c 呼び出しでレキシコンの順序を指定 (w3cAlternate の次に SynthesizeSpeech) すると、最初のレキシ

コンで指定した W3C のエイリアスは 2 番目より優先されます。レキシコンをテストするには、以下を実行します。

1. レキシコンを w3c.pls および w3cAlternate.pls という名前でローカルにファイルとして保存します。
2. put-lexicon AWS CLI コマンドを使用して、これらのレキシコンをアップロードします。
   - w3c.pls レキシコンをアップロードし、w3c として保存します。

     ```
     aws polly put-lexicon \
     --name w3c \
     --content file://w3c.pls
     ```

   - w3cAlternate.pls レキシコンをサービスに w3cAlternate としてアップロードします。

     ```
     aws polly put-lexicon \
     --name w3cAlternate \
     --content file://w3cAlternate.pls
     ```

3. synthesize-speech コマンドを実行してサンプルテキストを音声ストリーム (speech.mp3) に合成し、lexicon-name パラメーターを使用して両方のレキシコンを指定します。

   ```
   aws polly synthesize-speech \
   --text 'PLS is a W3C recommendation' \
   --voice-id Joanna \
   --output-format mp3 \
   --lexicon-names '["w3c","w3cAlternative"]' \
   speech.mp3
   ```

4. speech.mp3 の結果のテスト 以下のようになるはずです。

   ```
   PLS is a World Wide Web Consortium recommendation
   ```

PutLexicon API の追加コードサンプル

- Java サンプル: PutLexicon
- Python (Boto3) サンプル: PutLexicon

GetLexicon オペレーションの使用

Amazon Polly は、特定のリージョンのアカウントに保存した発音レキシコンのコンテンツを取得するために、GetLexicon API オペレーションを提供します。

次のget-lexicon AWS CLI コマンドは、exampleレキシコンのコンテンツを取得します。

```
aws polly get-lexicon \
--name example
```

まだアカウントに保存したレキシコンがないなら、PutLexicon オペレーションを使用してレキシコンを保存できます。詳細については、「PutLexicon オペレーションの使用」を参照してください。

レスポンスの例を次に示します。レキシコンのコンテンツに加えて、レスポンスでは、レキシコンが適用される言語コード、レキシコンで定義されている語彙素の数、リソースの Amazon リソースネーム (ARN)、レキシコンのサイズ (バイト単位) などのメタデータを返します。LastModified 値は、Unix タイムスタンプです。

```
{
    "Lexicon": {
        "Content": "lexicon content in plain text PLS format",
        "Name": "example"
    },
    "LexiconAttributes": {
        "LanguageCode": "en-US",
        "LastModified": 1474222543.989,
        "Alphabet": "ipa",
        "LexemesCount": 1,
        "LexiconArn": "arn:aws:polly:us-east-2:account-id:lexicon/example",
        "Size": 495
    }
}
```

GetLexicon API の追加コードサンプル

- Java サンプル: GetLexicon
- Python (Boto3) サンプル: GetLexicon

ListLexicons オペレーションの使用

Amazon Polly には、特定の AWS リージョンのアカウントで発音レキシコンのリストを取得するために使用できる ListLexicons API オペレーションが用意されています。次の呼び出しは、us-east-2 リージョンのアカウントのレキシコンを AWS CLI 一覧表示します。

```
aws polly list-lexicons
```

次はレスポンスの例で、w3c と tomato という 2 つのレキシコンを表示しています。各レキシコンで、レスポンスは、レキシコンが適用される言語コード、レキシコンで定義されている語彙素の数、バイト単位のサイズなどのメタデータを返します。言語コードは、レキシコンで定義されている語彙素が適用される言語および口ケールを示します。

```
{
    "Lexicons": [
        {
            "Attributes": {
                "LanguageCode": "en-US",
                "LastModified": 1474222543.989,
                "Alphabet": "ipa",
                "LexemesCount": 1,
                "LexiconArn": "arn:aws:polly:aws-region:account-id:lexicon/w3c",
                "Size": 495
            },
            "Name": "w3c"
        },
        {
            "Attributes": {
                "LanguageCode": "en-US",
                "LastModified": 1473099290.858,
                "Alphabet": "ipa",
                "LexemesCount": 1,
                "LexiconArn": "arn:aws:polly:aws-region:account-id:lexicon/tomato",
                "Size": 645
            },
            "Name": "tomato"
        }
    ]
}
```

ListLexicon API の追加コードサンプル

- Java サンプル: ListLexicons
- Python (Boto3) サンプル: ListLexicon

DeleteLexicon オペレーションの使用

Amazon Polly は、アカウント内の特定の AWS リージョンから発音レキシコンを削除する DeleteLexicon API オペレーションを提供します。以下では、指定されたレキシコン AWS CLI を削除します。

次の AWS CLI 例は、Unix、Linux、macOS 用にフォーマットされています。Windows の場合は、各行末のバックスラッシュ (\) Unix 連結文字をカレット (^) に置き換え、入力テキストは二重引用符 (") で囲み、内部タグは一重引用符 (') で囲みます。

```
aws polly delete-lexicon \
--name example
```

DeleteLexicon API の追加コードサンプル

- Java サンプル: DeleteLexicon
- Python (Boto3) サンプル: DeleteLexicon

長いオーディオファイルの作成

大規模なテキストの TTS ファイルを作成するには、Amazon Polly の非同期の合成機能を使用します。これには 3 つの SpeechSynthesisTask API を使用します。

- StartSpeechSynthesisTask: 新しい合成タスクを開始します。
- GetSpeechSynthesisTask: 以前に送信された合成タスクの詳細を返します。
- ListSpeechSynthesisTasks: 送信された合成タスクを一覧表示します。

SynthesizeSpeech オペレーションはほぼリアルタイムにオーディオを生成し、ほとんどの場合、レイテンシーは比較的わずかです。これを行う場合、このオペレーションで合成できるのは 3000 文字だけです。

Amazon Polly の非同期の合成機能は、ドキュメントが合成され返される方法を変更することにより、より大きなテキストドキュメントを処理するという課題を克服します。StartSpeechSynthesisTask を使用して入力テキストを送信することにより合成リクエストが行われた場合、Amazon Polly ではリクエストがキューに入れられ、システムリソースが利用可能になるとすぐに非同期で処理されます。Amazon Polly は作成されたスピーチまたはスピーチマークストリームを Amazon Simple Storage Service (Amazon S3) バケット (必須) に直接アップロードして、完了したファイルの可用性について SNS トピック (オプション) を通じて通知します。

このようにして、ほぼリアルタイムの処理を除くすべての機能で利用できる長さは、最大で請求可能な 100,000 文字 (または合計 200,000 文字) です。

このメソッドを使用してドキュメントを合成するには、オーディオファイルを保存できる書き込み可能な Amazon S3 バケットを用意する必要があります。オプションの SNS トピック ID を指定して音声の合成の準備が完了したときに通知を受け取ることができます。合成タスクが完了すると、Amazon Polly は、そのトピックにメッセージを発行します。合成タスクが成功しなかった場合、このメッセージには役立つエラー情報が含まれていることもあります。これを行うには、合成タスクを作成して、SNS トピックに発行できるユーザーを確認します。作成する方法の詳細と SNS トピックへのサブスクライブの詳細については、Amazon SNS ドキュメントを参照してください。

暗号化

必要に応じて出力ファイルを暗号化された形式で S3 バケットに保存できます。これを行うには、最強のブロック暗号の一つである、256 ビットの高度暗号化規格 (AES-256) を使用する Amazon S3 バケットの暗号化を有効にします。

トピック

非同期合成用の IAM ポリシーの設定

非同期の合成機能を使用するには、次のことを許可する IAM ポリシーが必要になります。

- 新しい Amazon Polly オペレーションの使用
- 出力 S3 バケットへの書き込み
- ステータス SNS トピックへの発行 (オプション)

次のポリシーでは、非同期の合成に必要なアクセス許可のみが付与され、IAM ユーザーにアタッチすることができます。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "polly:StartSpeechSynthesisTask",
        "polly:GetSpeechSynthesisTask",
        "polly:ListSpeechSynthesisTasks"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "s3:PutObject",
      "Resource": "arn:aws:s3:::bucket-name/*"
    },
    {
      "Effect": "Allow",
      "Action": "sns:Publish",
      "Resource": "arn:aws:sns:region:account:topic"
    }
  ]
```

```
}
```

コンソールでの長いオーディオファイルの作成

Amazon Polly コンソールでは、非同期の合成と、 AWS CLIで使用できるのと同じ機能を使用して、長い音声を作成できます。これを行うには、他の合成とほぼ同じように、[Text-to-Speech (テキスト読み上げ機能)] タブを使用します。

もう 1 つの非同期の合成機能は、コンソールでも利用できます。[S3 synthesis tasks (S3 合成タスク)] タブには、`ListSpeechSynthesisTasks` 機能が反映され、S3 バケットに保存されたすべてのタスクが表示され、必要な場合にはフィルタを使用できます。特定の 1 つのタスクをクリックすると、`GetSpeechSynthesisTask` 機能が反映された詳細が表示されます。

Amazon Polly コンソールを使用して大きなテキストを合成するには

1. にサインイン AWS Management Console し、https://console.aws.amazon.com/polly/ で Amazon Polly コンソールを開きます。
2. [Text-to-Speech (テキスト読み上げ機能)] タブを選択します。必要に応じて、エンジンとして [ロングフォーム] を選択します。
3. SSML をオンまたはオフにして、テキストを入力ボックスに入力するか、または貼り付けます。
4. テキストの言語、リージョン、および音声を選択します。
5. [Save to S3] (S3 に保存) を選択します。

   > **Note**
   >
   > テキストの長さがリアルタイム`SynthesizeSpeech`オペレーションの 3,000 文字の制限を超える場合、ダウンロードオプションとリッスンオプションの両方がグレー表示されます。

6. コンソールでフォームが開き、出力ファイルの保存先を選択できます。

   a. 宛先 Amazon S3 バケットの名前を入力します。

   b. 必要に応じて、出力のプレフィックスキーを入力します。

      > **Note**
      >
      > 出力 S3 バケットは書き込み可能である必要があります。

c. 合成タスクが完了したら通知を受け取る場合は、オプションの SNS トピック ID を提供します。

> **Note**
>
> 現在のコンソールユーザーがこのオプションを使用する場合、SNS は発行するために開いている必要があります。詳細については、「Amazon Simple Notification Service (SNS)」を参照してください。

d. [Save to S3] (S3 に保存) を選択します。

音声合成タスクの情報を取得するには

1. コンソールで、[S3 Synthesis Tasks (S3 合成タスク)] タブを選択します。
2. タスクは、日付順に表示されます。タスクをステータスでフィルタリングするには、[All statuses] (すべてのステータス) を選択して、使用するステータスを選択します。
3. 特定のタスクの詳細を表示するには、リンクされた [Task ID (タスク ID)] を選択します。

での長いオーディオファイルの作成 AWS CLI

Amazon Polly の非同期の合成機能は、3 つの `SpeechSynthesisTask` API を使用して大量のテキストを操作します。

- `StartSpeechSynthesisTask`: 新しい合成タスクを開始します。
- `GetSpeechSynthesisTask`: 以前に送信された合成タスクの詳細を返します。
- `ListSpeechSynthesisTasks`: 送信された合成タスクを一覧表示します。

大量のテキストの合成 (**`StartSpeechSynthesisTask`**)

リアルタイム `SynthesizeSpeech` で作成できるオーディオファイルよりも大きなオーディオファイルを作成する場合、`StartSpeechSynthesisTask` オペレーションを使用します。`SynthesizeSpeech` オペレーションに必要な引数に加えて、`StartSpeechSynthesisTask` には Amazon S3 バケットの名前も必要です。タスクに関するステータス通知を受信する場合、他の 2 つのオプションの引数 (出力ファイルのキープレフィックス、および SNS トピックの ARN) を指定することもできます。

- OutputS3BucketName: 合成をアップロードする Amazon S3バケット の名前。このバケットは Amazon Polly サービスと同じリージョンで作成される必要があります。さらに、呼び出しに使用されている IAM ユーザーにはこのバケットへのアクセス権が必要です。[必須]
- OutputS3KeyPrefix: 出力ファイルのキープレフィックス。お使いのバケット内のカスタムディレクトリのようなキーに出力音声ファイルを保存する場合は、このパラメータを使用します。[オプション]
- SnsTopicArn: タスクのステータスに関する通知を受け取る場合に使用する SNS トピックの ARN。この SNS トピックは Amazon Polly サービスと同じリージョンで作成される必要があります。さらに、呼び出しに使用されている IAM ユーザーにはこのトピックへのアクセス権が必要です。[オプション]

例えば、次の例を使用して、米国東部 (オハイオ) リージョンで start-speech-synthesis-task AWS CLI コマンドを実行できます。

次の AWS CLI 例は、Unix、Linux、macOS 用にフォーマットされています。Windows の場合は、各行末のバックスラッシュ (\) Unix 連結文字をカレット (^) に置き換え、入力テキストは二重引用符 (") で囲み、内部タグは一重引用符 (') で囲みます。

```
aws polly start-speech-synthesis-task \
  --region us-east-2 \
  --endpoint-url "https://polly.us-east-2.amazonaws.com/" \
  --output-format mp3 \
  --output-s3-bucket-name your-bucket-name \
  --output-s3-key-prefix optional/prefix/path/file \
  --voice-id Joanna \
  --text file://text_file.txt
```

これにより、次のような応答が得られます。

```
"SynthesisTask":
{
     "OutputFormat": "mp3",
     "OutputUri": "https://s3.us-east-2.amazonaws.com/your-bucket-name/optional/prefix/
path/file.<task_id>.mp3",
     "TextType": "text",
     "CreationTime": [..],
     "RequestCharacters": [..],
     "TaskStatus": "scheduled",
```

```
    "TaskId": [task_id],
    "VoiceId": "Joanna"
}
```

start-speech-synthesis-task オペレーションは、複数の新しいフィールドを返します。

- OutputUri: 出力音声ファイルの場所。
- TaskId: Amazon Polly で生成された音声合成タスクの一意の識別子。
- CreationTime: タスクが最初に送信された時刻のタイムスタンプ。
- RequestCharacters: タスク内の請求可能な文字の数。
- TaskStatus: 送信されたタスクのステータスに関する情報を提供します。

  タスクが送信されると、最初のステータス scheduled が表示されます。Amazon Polly がタスクの処理を開始すると、ステータスが inProgress に変わり、さらに completed または failed へと変化します。タスクが失敗した場合、 GetSpeechSynthesisTask または ListSpeechSynthesisTasks オペレーションを呼び出すと、エラーメッセージが返されます。

タスクが完了すると、OutputUri で指定した場所で、音声ファイルが利用できます。

音声合成タスクの情報の取得

エラー、ステータスなどのタスクの情報を取得するには、GetSpeechSynthesisTask オペレーションを使用します。これを行うには、task-id によって返される StartSpeechSynthesisTask が必要です。

例えば、次の例を使用して get-speech-synthesis-task AWS CLI コマンドを実行できます。

```
aws polly get-speech-synthesis-task \
--region us-east-2 \
--endpoint-url "https:// polly.us-east-2.amazonaws.com/" \
--task-id task identifier
```

ListSpeechSynthesisTasks オペレーションを使用して、現在のリージョンで実行したすべての音声合成タスクを一覧表示することもできます。

例えば、次の例を使用して list-speech-synthesis-tasks AWS CLI コマンドを実行できます。

```
aws polly list-speech-synthesis-tasks \
--region us-east-2 \
--endpoint-url "https:// polly.us-east-2.amazonaws.com/"
```

コードとアプリケーションの例

このセクションでは、Amazon Polly の学習に使用できるコードとアプリケーションの例を示します。

トピック

- 「サンプルコード」
- サンプルアプリケーション

サンプルコードのトピックには、プログラミング言語で編成されたコードスニペットが含まれており、さまざまな Amazon Polly 機能の例に分かれています。サンプルアプリケーションのトピックには、Amazon Polly を学習するために独立して使用できる、プログラミング言語で編成されたアプリケーションが含まれています。

これらの例を使用し始める前に、まず「Amazon Polly の仕組み」を読んで「Amazon Polly の開始方法」で説明されているステップに従うことをお勧めします。

「サンプルコード」

このトピックには、Amazon Polly を学習するために使用できるさまざまな機能のコードサンプルが含まれています。

プログラミング言語によるサンプルコード

- Java サンプル
- Python サンプル

Java サンプル

次のコードサンプルは、Java ベースのアプリケーションを使用して Amazon Polly でさまざまなタスクを実行する方法を示しています。これらのサンプルは完全な例ではありませんが、AWS SDK for Java を使用する場合よりも大きな Java アプリケーションに含めることができます。

コードスニペット

- DeleteLexicon
- DescribeVoices

- GetLexicon
- ListLexicons
- PutLexicon
- StartSpeechSynthesisTask
- スピーチマーク
- SynthesizeSpeech

DeleteLexicon

次の Java コードサンプルは、Java ベースのアプリケーションを使用して AWS リージョンに格納されている特定のレキシコンを削除する方法を示しています。削除されたレキシコンは、音声合成には使用できません。また、`GetLexicon` または `ListLexicon` API を使用して取得することもできません。

このオペレーションの詳細については、`DeleteLexicon` API リファレンスを参照してください。

```
package com.amazonaws.polly.samples;

import com.amazonaws.services.polly.AmazonPolly;
import com.amazonaws.services.polly.AmazonPollyClientBuilder;
import com.amazonaws.services.polly.model.DeleteLexiconRequest;

public class DeleteLexiconSample {
    private String LEXICON_NAME = "SampleLexicon";

    AmazonPolly client = AmazonPollyClientBuilder.defaultClient();

    public void deleteLexicon() {
        DeleteLexiconRequest deleteLexiconRequest = new
 DeleteLexiconRequest().withName(LEXICON_NAME);

        try {
            client.deleteLexicon(deleteLexiconRequest);
        } catch (Exception e) {
            System.err.println("Exception caught: " + e);
        }
    }
}
```

DescribeVoices

次の Java コードサンプルは、Javaベースのアプリケーションを使用して、音声合成を要求するときに使用できる音声のリストを生成する方法を示しています。必要に応じて、使用可能な音声をフィルタリングする言語コードを指定できます。たとえば、en-US を指定すると、使用可能なすべてのアメリカ英語の音声のリストが返されます。

このオペレーションの詳細については、DescribeVoices API リファレンスを参照してください。

```
package com.amazonaws.polly.samples;

import com.amazonaws.services.polly.AmazonPolly;
import com.amazonaws.services.polly.AmazonPollyClientBuilder;
import com.amazonaws.services.polly.model.DescribeVoicesRequest;
import com.amazonaws.services.polly.model.DescribeVoicesResult;

public class DescribeVoicesSample {
    AmazonPolly client = AmazonPollyClientBuilder.defaultClient();

    public void describeVoices() {
        DescribeVoicesRequest allVoicesRequest = new DescribeVoicesRequest();
        DescribeVoicesRequest enUsVoicesRequest = new
 DescribeVoicesRequest().withLanguageCode("en-US");

        try {
            String nextToken;
            do {
                DescribeVoicesResult allVoicesResult =
 client.describeVoices(allVoicesRequest);
                nextToken = allVoicesResult.getNextToken();
                allVoicesRequest.setNextToken(nextToken);

                System.out.println("All voices: " + allVoicesResult.getVoices());
            } while (nextToken != null);

            do {
                DescribeVoicesResult enUsVoicesResult =
 client.describeVoices(enUsVoicesRequest);
                nextToken = enUsVoicesResult.getNextToken();
                enUsVoicesRequest.setNextToken(nextToken);

                System.out.println("en-US voices: " + enUsVoicesResult.getVoices());
            } while (nextToken != null);
```

```
        } catch (Exception e) {
            System.err.println("Exception caught: " + e);
        }
    }
}
```

GetLexicon

次の Java コードサンプルは、Java ベースのアプリケーションを使用して、 AWS リージョンに保存されている特定の発音レキシコンのコンテンツを生成する方法を示しています。

このオペレーションの詳細については、GetLexicon API リファレンスを参照してください。

```
package com.amazonaws.polly.samples;

import com.amazonaws.services.polly.AmazonPolly;
import com.amazonaws.services.polly.AmazonPollyClientBuilder;
import com.amazonaws.services.polly.model.GetLexiconRequest;
import com.amazonaws.services.polly.model.GetLexiconResult;

public class GetLexiconSample {
    private String LEXICON_NAME = "SampleLexicon";

    AmazonPolly client = AmazonPollyClientBuilder.defaultClient();

    public void getLexicon() {
        GetLexiconRequest getLexiconRequest = new
 GetLexiconRequest().withName(LEXICON_NAME);

        try {
            GetLexiconResult getLexiconResult = client.getLexicon(getLexiconRequest);
            System.out.println("Lexicon: " + getLexiconResult.getLexicon());
        } catch (Exception e) {
            System.err.println("Exception caught: " + e);
        }
    }
}
```

ListLexicons

次の Java コードサンプルは、Java ベースのアプリケーションを使用して、 AWS リージョンに保存されている発音レキシコンのリストを生成する方法を示しています。

このオペレーションの詳細については、ListLexicons API リファレンスを参照してください。

```
package com.amazonaws.polly.samples;

import com.amazonaws.services.polly.AmazonPolly;
import com.amazonaws.services.polly.AmazonPollyClientBuilder;
import com.amazonaws.services.polly.model.LexiconAttributes;
import com.amazonaws.services.polly.model.LexiconDescription;
import com.amazonaws.services.polly.model.ListLexiconsRequest;
import com.amazonaws.services.polly.model.ListLexiconsResult;

public class ListLexiconsSample {
    AmazonPolly client = AmazonPollyClientBuilder.defaultClient();

    public void listLexicons() {
        ListLexiconsRequest listLexiconsRequest = new ListLexiconsRequest();

        try {
            String nextToken;
            do {
                ListLexiconsResult listLexiconsResult =
 client.listLexicons(listLexiconsRequest);
                nextToken = listLexiconsResult.getNextToken();
                listLexiconsRequest.setNextToken(nextToken);

                for (LexiconDescription lexiconDescription :
 listLexiconsResult.getLexicons()) {
                    LexiconAttributes attributes = lexiconDescription.getAttributes();
                    System.out.println("Name: " + lexiconDescription.getName()
                            + ", Alphabet: " + attributes.getAlphabet()
                            + ", LanguageCode: " + attributes.getLanguageCode()
                            + ", LastModified: " + attributes.getLastModified()
                            + ", LexemesCount: " + attributes.getLexemesCount()
                            + ", LexiconArn: " + attributes.getLexiconArn()
                            + ", Size: " + attributes.getSize());
                }
            } while (nextToken != null);
        } catch (Exception e) {
            System.err.println("Exception caught: " + e);
        }
    }
}
```

PutLexicon

次の Java コードサンプルは、Java ベースのアプリケーションを使用して発音レキシコンを AWS リージョンに格納する方法を示しています。

このオペレーションの詳細については、PutLexicon API リファレンスを参照してください。

```
package com.amazonaws.polly.samples;

import com.amazonaws.services.polly.AmazonPolly;
import com.amazonaws.services.polly.AmazonPollyClientBuilder;
import com.amazonaws.services.polly.model.PutLexiconRequest;

public class PutLexiconSample {
    AmazonPolly client = AmazonPollyClientBuilder.defaultClient();

    private String LEXICON_CONTENT = "<?xml version=\"1.0\" encoding=\"UTF-8\"?>" +
            "<lexicon version=\"1.0\" xmlns=\"http://www.w3.org/2005/01/pronunciation-
lexicon\" xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\" " +
            "xsi:schemaLocation=\"http://www.w3.org/2005/01/pronunciation-lexicon
 http://www.w3.org/TR/2007/CR-pronunciation-lexicon-20071212/pls.xsd\" " +
            "alphabet=\"ipa\" xml:lang=\"en-US\">" +
            "<lexeme><grapheme>test1</grapheme><alias>test2</alias></lexeme>" +
            "</lexicon>";
    private String LEXICON_NAME = "SampleLexicon";

    public void putLexicon() {
        PutLexiconRequest putLexiconRequest = new PutLexiconRequest()
                .withContent(LEXICON_CONTENT)
                .withName(LEXICON_NAME);

        try {
            client.putLexicon(putLexiconRequest);
        } catch (Exception e) {
            System.err.println("Exception caught: " + e);
        }
    }
}
```

StartSpeechSynthesisTask

次の Java コードサンプルは、Java ベースのアプリケーションを使用して長い音声 (最大 100,000 課金対象文字) を合成し、Amazon S3 バケットに直接保存する方法を示しています。

詳細については、StartSpeechSynthesisTask API リファレンスを参照してください。

```
package com.amazonaws.parrot.service.tests.speech.task;

import com.amazonaws.parrot.service.tests.AbstractParrotServiceTest;
import com.amazonaws.services.polly.AmazonPolly;
import com.amazonaws.services.polly.model.*;
import org.awaitility.Duration;

import java.util.concurrent.TimeUnit;

import static org.awaitility.Awaitility.await;

public class StartSpeechSynthesisTaskSample {

    private static final int SYNTHESIS_TASK_TIMEOUT_SECONDS = 300;
    private static final AmazonPolly AMAZON_POLLY_CLIENT =
 AmazonPollyClientBuilder.defaultClient();
    private static final String PLAIN_TEXT = "This is a sample text to be
 synthesized.";
    private static final String OUTPUT_FORMAT_MP3 = OutputFormat.Mp3.toString();
    private static final String OUTPUT_BUCKET = "synth-books-buckets";
    private static final String SNS_TOPIC_ARN = "arn:aws:sns:eu-
west-2:123456789012:synthesize-finish-topic";
    private static final Duration SYNTHESIS_TASK_POLL_INTERVAL = Duration.FIVE_SECONDS;
    private static final Duration SYNTHESIS_TASK_POLL_DELAY = Duration.TEN_SECONDS;

    public static void main(String... args) {
        StartSpeechSynthesisTaskRequest request = new StartSpeechSynthesisTaskRequest()
                .withOutputFormat(OUTPUT_FORMAT_MP3)
                .withText(PLAIN_TEXT)
                .withTextType(TextType.Text)
                .withVoiceId(VoiceId.Amy)
                .withOutputS3BucketName(OUTPUT_BUCKET)
                .withSnsTopicArn(SNS_TOPIC_ARN)
                .withEngine("neural");

        StartSpeechSynthesisTaskResult result =
 AMAZON_POLLY_CLIENT.startSpeechSynthesisTask(request);
        String taskId = result.getSynthesisTask().getTaskId();

        await().with()
                .pollInterval(SYNTHESIS_TASK_POLL_INTERVAL)
                .pollDelay(SYNTHESIS_TASK_POLL_DELAY)
```

```
                .atMost(SYNTHESIS_TASK_TIMEOUT_SECONDS, TimeUnit.SECONDS)
                .until(
                        () ->
 getSynthesisTaskStatus(taskId).equals(TaskStatus.Completed.toString())
                );
    }

    private static SynthesisTask getSynthesisTask(String taskId) {
        GetSpeechSynthesisTaskRequest getSpeechSynthesisTaskRequest = new
 GetSpeechSynthesisTaskRequest()
                .withTaskId(taskId);
        GetSpeechSynthesisTaskResult result
 =AMAZON_POLLY_CLIENT.getSpeechSynthesisTask(getSpeechSynthesisTaskRequest);
        return result.getSynthesisTask();
    }

    private static String getSynthesisTaskStatus(String taskId) {
        GetSpeechSynthesisTaskRequest getSpeechSynthesisTaskRequest = new
 GetSpeechSynthesisTaskRequest()
                .withTaskId(taskId);
        GetSpeechSynthesisTaskResult result
 =AMAZON_POLLY_CLIENT.getSpeechSynthesisTask(getSpeechSynthesisTaskRequest);
        return result.getSynthesisTask().getTaskStatus();
    }

}
```

スピーチマーク

次のコードサンプルは、Java ベースのアプリケーションを使用して入力テキストのスピーチマークを合成する方法を示しています。この機能は SynthesizeSpeech API を使用します。

この機能の詳細については、「スピーチマーク」を参照してください。

API の詳細については、SynthesizeSpeech API リファレンスを参照してください。

```
package com.amazonaws.polly.samples;

import com.amazonaws.services.polly.AmazonPolly;
import com.amazonaws.services.polly.AmazonPollyClientBuilder;
```

```
import com.amazonaws.services.polly.model.OutputFormat;
import com.amazonaws.services.polly.model.SpeechMarkType;
import com.amazonaws.services.polly.model.SynthesizeSpeechRequest;
import com.amazonaws.services.polly.model.SynthesizeSpeechResult;
import com.amazonaws.services.polly.model.VoiceId;

import java.io.File;
import java.io.FileOutputStream;
import java.io.InputStream;

public class SynthesizeSpeechMarksSample {
    AmazonPolly client = AmazonPollyClientBuilder.defaultClient();

    public void synthesizeSpeechMarks() {
        String outputFileName = "/tmp/speechMarks.json";

        SynthesizeSpeechRequest synthesizeSpeechRequest = new SynthesizeSpeechRequest()
                .withOutputFormat(OutputFormat.Json)
                .withSpeechMarkTypes(SpeechMarkType.Viseme, SpeechMarkType.Word)
                .withVoiceId(VoiceId.Joanna)
                .withText("This is a sample text to be synthesized.");

        try (FileOutputStream outputStream = new FileOutputStream(new
 File(outputFileName))) {
            SynthesizeSpeechResult synthesizeSpeechResult =
 client.synthesizeSpeech(synthesizeSpeechRequest);
            byte[] buffer = new byte[2 * 1024];
            int readBytes;

            try (InputStream in = synthesizeSpeechResult.getAudioStream()){
                while ((readBytes = in.read(buffer)) > 0) {
                    outputStream.write(buffer, 0, readBytes);
                }
            }
        } catch (Exception e) {
            System.err.println("Exception caught: " + e);
        }
    }
}
```

SynthesizeSpeech

次の Java コードサンプルは、Java ベースのアプリケーションを使用してほぼリアルタイムの処理で短いテキストを合成する方法を示しています。

詳細については、SynthesizeSpeech API リファレンスを参照してください。

```
package com.amazonaws.polly.samples;

import com.amazonaws.services.polly.AmazonPolly;
import com.amazonaws.services.polly.AmazonPollyClientBuilder;
import com.amazonaws.services.polly.model.OutputFormat;
import com.amazonaws.services.polly.model.SynthesizeSpeechRequest;
import com.amazonaws.services.polly.model.SynthesizeSpeechResult;
import com.amazonaws.services.polly.model.VoiceId;

import java.io.File;
import java.io.FileOutputStream;
import java.io.InputStream;

public class SynthesizeSpeechSample {
    AmazonPolly client = AmazonPollyClientBuilder.defaultClient();

    public void synthesizeSpeech() {
        String outputFileName = "/tmp/speech.mp3";

        SynthesizeSpeechRequest synthesizeSpeechRequest = new SynthesizeSpeechRequest()
                .withOutputFormat(OutputFormat.Mp3)
                .withVoiceId(VoiceId.Joanna)
                .withText("This is a sample text to be synthesized.")
                .withEngine("neural");

        try (FileOutputStream outputStream = new FileOutputStream(new
 File(outputFileName))) {
            SynthesizeSpeechResult synthesizeSpeechResult =
 client.synthesizeSpeech(synthesizeSpeechRequest);
            byte[] buffer = new byte[2 * 1024];
            int readBytes;

            try (InputStream in = synthesizeSpeechResult.getAudioStream()){
                while ((readBytes = in.read(buffer)) > 0) {
                    outputStream.write(buffer, 0, readBytes);
                }
```

```
            }
        } catch (Exception e) {
            System.err.println("Exception caught: " + e);
        }
    }
}
```

Python サンプル

次のコードサンプルは、Python (boto3) ベースのアプリケーションを使用して Amazon Polly でさまざまなタスクを実行する方法を示しています。これらのサンプルは完全な例を意図したものではありませんが、AWS SDK for Python (Boto) を使用するより大きな Python アプリケーションに含めることができます。

コードスニペット

- DeleteLexicon
- GetLexicon
- ListLexicon
- PutLexicon
- StartSpeechSynthesisTask
- SynthesizeSpeech

DeleteLexicon

次の Python コード例では、 を使用して AWS SDK for Python (Boto) 、ローカル AWS 設定で指定されたリージョンのレキシコンを削除します。例では、指定したレキシコンのみを削除します。実際にレキシコンを削除する前に、処理を進めることを確認するよう求めます。

次のコード例では、 AWS SDK 設定ファイルに保存されているデフォルトの認証情報を使用しています。設定ファイルを作成する方法については、「ステップ 2.1: をセットアップする AWS CLI」を参照してください。

このオペレーションの詳細については、`DeleteLexicon` API リファレンスを参照してください。

```
from argparse import ArgumentParser
from sys import version_info

from boto3 import Session
from botocore.exceptions import BotoCoreError, ClientError
```

```
# Define and parse the command line arguments
cli = ArgumentParser(description="DeleteLexicon example")
cli.add_argument("name", type=str, metavar="LEXICON_NAME")
arguments = cli.parse_args()

# Create a client using the credentials and region defined in the adminuser
# section of the AWS credentials and configuration files
session = Session(profile_name="adminuser")
polly = session.client("polly")

# Request confirmation
prompt = input if version_info >= (3, 0) else raw_input
proceed = prompt((u"This will delete the \"{0}\" lexicon,"
                  " do you want to proceed? [y,n]: ").format(arguments.name))

if proceed in ("y", "Y"):
    print(u"Deleting {0}...".format(arguments.name))

    try:
        # Request deletion of a lexicon by name
        response = polly.delete_lexicon(Name=arguments.name)
    except (BotoCoreError, ClientError) as error:
        # The service returned an error, exit gracefully
        cli.error(error)

    print("Done.")
else:
    print("Cancelled.")
```

GetLexicon

次の Python コードは、 AWS SDK for Python (Boto) を使用して、 AWS リージョンに保存されているすべてのレキシコンを取得します。例では、レキシコン名をコマンドラインのパラメーターとして受け入れて、そのレキシコンのみを取得し、ローカルに保存された場所の tmp パスを印刷します。

次のコード例では、 AWS SDK 設定ファイルに保存されているデフォルトの認証情報を使用しています。設定ファイルを作成する方法については、「ステップ 2.1: をセットアップする AWS CLI」を参照してください。

このオペレーションの詳細については、GetLexicon API リファレンスを参照してください。

```
from argparse import ArgumentParser
from os import path
from tempfile import gettempdir

from boto3 import Session
from botocore.exceptions import BotoCoreError, ClientError

# Define and parse the command line arguments
cli = ArgumentParser(description="GetLexicon example")
cli.add_argument("name", type=str, metavar="LEXICON_NAME")
arguments = cli.parse_args()

# Create a client using the credentials and region defined in the adminuser
# section of the AWS credentials and configuration files
session = Session(profile_name="adminuser")
polly = session.client("polly")

print(u"Fetching {0}...".format(arguments.name))

try:
    # Fetch lexicon by name
    response = polly.get_lexicon(Name=arguments.name)
except (BotoCoreError, ClientError) as error:
    # The service returned an error, exit gracefully
    cli.error(error)

# Get the lexicon data from the response
lexicon = response.get("Lexicon", {})

# Access the lexicon's content
if "Content" in lexicon:
    output = path.join(gettempdir(), u"%s.pls" % arguments.name)
    print(u"Saving to %s..." % output)

    try:
        # Save the lexicon contents to a local file
        with open(output, "w") as pls_file:
            pls_file.write(lexicon["Content"])
    except IOError as error:
        # Could not write to file, exit gracefully
        cli.error(error)
else:
    # The response didn't contain lexicon data, exit gracefully
```

```
    cli.error("Could not fetch lexicons contents")

print("Done.")
```

ListLexicon

次の Python コード例では AWS SDK for Python (Boto) 、 を使用して、ローカル AWS 設定で指定されたリージョンのアカウントのレキシコンを一覧表示します。設定ファイルを作成する方法については、「ステップ 2.1: をセットアップする AWS CLI」を参照してください。

このオペレーションの詳細については、`ListLexicons` API リファレンスを参照してください。

```
import sys

from boto3 import Session
from botocore.exceptions import BotoCoreError, ClientError

# Create a client using the credentials and region defined in the adminuser
# section of the AWS credentials and configuration files
session = Session(profile_name="adminuser")
polly = session.client("polly")

try:
    # Request the list of available lexicons
    response = polly.list_lexicons()
except (BotoCoreError, ClientError) as error:
    # The service returned an error, exit gracefully
    print(error)
    sys.exit(-1)

# Get the list of lexicons in the response
lexicons = response.get("Lexicons", [])
print("{0} lexicon(s) found".format(len(lexicons)))

# Output a formatted list of lexicons with some of the attributes
for lexicon in lexicons:
    print((u" - {Name} ({Attributes[LanguageCode]}), "
           "{Attributes[LexemesCount]} lexeme(s)").format(**lexicon))
```

PutLexicon

次のコードサンプルは、Python (boto3) ベースのアプリケーションを使用して発音レキシコンを AWS リージョンに保存する方法を示しています。

このオペレーションの詳細については、PutLexicon API リファレンスを参照してください。

次の点に注意してください。

- ローカルのレキシコンファイル名と保存済みのレキシコンの名前を指定してコードを更新する必要があります。
- 例では、pls というサブディレクトリで作成したレキシコンファイルがあると想定しています。必要に応じてパスを更新する必要があります。

次のコード例では、 AWS SDK 設定ファイルに保存されているデフォルトの認証情報を使用しています。設定ファイルを作成する方法については、「ステップ 2.1: をセットアップする AWS CLI」を参照してください。

このオペレーションの詳細については、PutLexicon API リファレンスを参照してください。

```
from argparse import ArgumentParser

from boto3 import Session
from botocore.exceptions import BotoCoreError, ClientError

# Define and parse the command line arguments
cli = ArgumentParser(description="PutLexicon example")
cli.add_argument("path", type=str, metavar="FILE_PATH")
cli.add_argument("-n", "--name", type=str, required=True,
                 metavar="LEXICON_NAME", dest="name")
arguments = cli.parse_args()

# Create a client using the credentials and region defined in the adminuser
# section of the AWS credentials and configuration files
session = Session(profile_name="adminuser")
polly = session.client("polly")

# Open the PLS lexicon file for reading
try:
    with open(arguments.path, "r") as lexicon_file:
```

```
        # Read the pls file contents
        lexicon_data = lexicon_file.read()

        # Store the PLS lexicon on the service.
        # If a lexicon with that name already exists,
        # its contents will be updated
        response = polly.put_lexicon(Name=arguments.name,
                                     Content=lexicon_data)
except (IOError, BotoCoreError, ClientError) as error:
    # Could not open/read the file or the service returned an error,
    # exit gracefully
    cli.error(error)

print(u"The \"{0}\" lexicon is now available for use.".format(arguments.name))
```

StartSpeechSynthesisTask

次の Python コード例では AWS SDK for Python (Boto) 、 を使用して、ローカル AWS 設定で指定されたリージョンのアカウントのレキシコンを一覧表示します。設定ファイルを作成する方法については、「ステップ 2.1: をセットアップする AWS CLI」を参照してください。

詳細については、StartSpeechSynthesisTask API リファレンスを参照してください。

```
import boto3
import time

polly_client = boto3.Session(
                aws_access_key_id='',
    aws_secret_access_key='',
    region_name='eu-west-2').client('polly')

response = polly_client.start_speech_synthesis_task(VoiceId='Joanna',
                OutputS3BucketName='synth-books-buckets',
                OutputS3KeyPrefix='key',
                OutputFormat='mp3',
                Text='This is a sample text to be synthesized.',
                Engine='neural')

taskId = response['SynthesisTask']['TaskId']

print( "Task id is {} ".format(taskId))
```

```
task_status = polly_client.get_speech_synthesis_task(TaskId = taskId)

print(task_status)
```

SynthesizeSpeech

次の Python コード例では、 AWS SDK for Python (Boto) を使用してほぼリアルタイムの処理で短いテキストを合成する方法を示しています。詳細については、SynthesizeSpeech オペレーションのリファレンスを参照してください。

この例では、プレーンテキストの短い文字列を使用します。SSML テキストを使用して、出力をより詳細に制御できます。詳細については、「SSML ドキュメントからの音声の生成」を参照してください。

```
import boto3

polly_client = boto3.Session(
                aws_access_key_id=,
    aws_secret_access_key=,
    region_name='us-west-2').client('polly')

response = polly_client.synthesize_speech(VoiceId='Joanna',
                OutputFormat='mp3',
                Text = 'This is a sample text to be synthesized.',
                Engine = 'neural')

file = open('speech.mp3', 'wb')
file.write(response['AudioStream'].read())
file.close()
```

サンプルアプリケーション

このセクションには、Amazon Polly を学習するために使用できるサンプルアプリケーションの形での追加の例が含まれています。

プログラミング言語によるアプリケーションの例

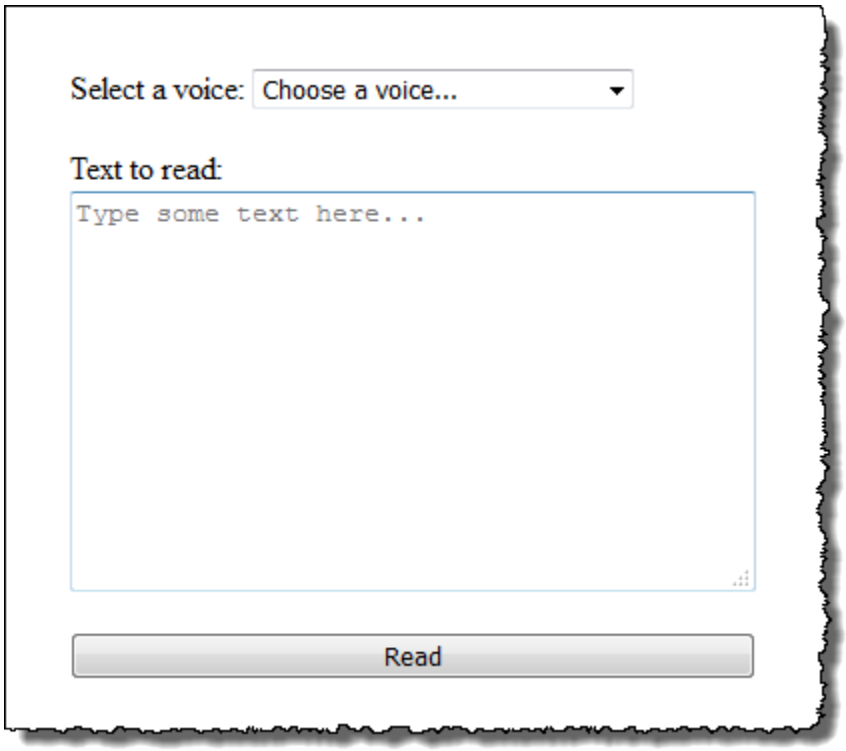
- Python の例 (HTML5 クライアントと Python サーバー)
- Java の例
- iOS の例

- Android の例

Python の例 (HTML5 クライアントと Python サーバー)

このサンプルアプリケーションは、以下の内容で構成されています。

- HTTP チャンク転送コードを使用する HTTP 1.1 サーバー (「Chunked Transfer Coding」を参照)
- HTTP 1.1 サーバーとやり取りするシンプルな HTML5 ユーザーインターフェイス (以下を参照):



この例の目的は、Amazon Polly を使用してブラウザベースの HTML5 アプリケーションから音声をストリーミングする方法を示すことです。テキストの合成と同時に Amazon Polly により生成された音声ストリームを使用する方法は、応答性が重要な要素であるユースケース (ダイアログシステムやスクリーンリーダーなど) に推奨されるアプローチです。

このサンプルアプリケーションを実行するには、以下のものが必要です。

- HTML5 および EcmaScript5 標準に準拠したウェブブラウザ (Chrome 23.0 以降、Firefox 21.0 以降、Internet Explorer 9.0 以降など)
- Python バージョン 3.0 以上

アプリケーションをテストするには

1. サーバーコードを server.py として保存します。コードについては、「Python の例: Python サーバーコード (server.py)」を参照してください。
2. HTML5 クライアントコードを index.html として保存します。コードについては、「Python の例: HTML5 ユーザーインターフェイス (index.html)」を参照してください。
3. server.py を保存したパスから次のコマンドを実行し、アプリケーションを起動します (システムによっては、コマンドの実行時に python ではなく python3 を使用する必要があります)。

   ```
   $ python  server.py
   ```

   アプリケーションが起動すると、URL がターミナルに表示されます。
4. ターミナルに表示された URL をウェブブラウザで開きます。

   アプリケーションサーバーのアドレスとポートを渡して、server.py のパラメーターとして使用することができます。詳細については、python server.py -h を実行してください。
5. 音声を聞くには、リストから音声を選択してテキストを入力し、[Read] を選択します。Amazon Polly が使用可能な最初の音声データチャンクを転送するとすぐに、音声の再生が始まります。
6. アプリケーションのテストが終わって Python サーバーを停止するには、サーバーが実行されているターミナルで Ctrl+C を押します。

> **Note**
>
> サーバーにより、 AWS SDK for Python (Boto)を使用して Boto3 が作成されます。クライアントは、コンピュータの設定 AWS ファイルに保存されている認証情報を使用して、Amazon Polly へのリクエストに署名して認証します。 AWS 設定ファイルの作成と認証情報の保管の方法については、AWS Command Line Interface ユーザーガイドのAWS Command Line Interfaceの設定を参照してください。

Python の例: HTML5 ユーザーインターフェイス (index.html)

このセクションでは、「Python の例 (HTML5 クライアントと Python サーバー)」で説明されている HTML5 クライアントのコードを示します。

```
<html>
```

```
<head>
    <title>Text-to-Speech Example Application</title>
    <script>
        /*
         * This sample code requires a web browser with support for both the
         * HTML5 and ECMAScript 5 standards; the following is a non-comprehensive
         * list of compliant browsers and their minimum version:
         *
         * - Chrome 23.0+
         * - Firefox 21.0+
         * - Internet Explorer 9.0+
         * - Edge 12.0+
         * - Opera 15.0+
         * - Safari 6.1+
         * - Android (stock web browser) 4.4+
         * - Chrome for Android 51.0+
         * - Firefox for Android 48.0+
         * - Opera Mobile 37.0+
         * - iOS (Safari Mobile and Chrome) 3.2+
         * - Internet Explorer Mobile 10.0+
         * - Blackberry Browser 10.0+
         */

        // Mapping of the OutputFormat parameter of the SynthesizeSpeech API
        // and the audio format strings understood by the browser
        var AUDIO_FORMATS = {
            'ogg_vorbis': 'audio/ogg',
            'mp3': 'audio/mpeg',
            'pcm': 'audio/wave; codecs=1'
        };

        /**
         * Handles fetching JSON over HTTP
         */
        function fetchJSON(method, url, onSuccess, onError) {
            var request = new XMLHttpRequest();
            request.open(method, url, true);
            request.onload = function () {
                // If loading is complete
                if (request.readyState === 4) {
                    // if the request was successful
                    if (request.status === 200) {
                        var data;
```

```
                    // Parse the JSON in the response
                    try {
                        data = JSON.parse(request.responseText);
                    } catch (error) {
                        onError(request.status, error.toString());
                    }

                    onSuccess(data);
                } else {
                    onError(request.status, request.responseText)
                }
            }
        };

        request.send();
    }

    /**
     * Returns a list of audio formats supported by the browser
     */
    function getSupportedAudioFormats(player) {
        return Object.keys(AUDIO_FORMATS)
            .filter(function (format) {
                var supported = player.canPlayType(AUDIO_FORMATS[format]);
                return supported === 'probably' || supported === 'maybe';
            });
    }

    // Initialize the application when the DOM is loaded and ready to be
    // manipulated
    document.addEventListener("DOMContentLoaded", function () {
        var input = document.getElementById('input'),
            voiceMenu = document.getElementById('voice'),
            text = document.getElementById('text'),
            player = document.getElementById('player'),
            submit = document.getElementById('submit'),
            supportedFormats = getSupportedAudioFormats(player);

        // Display a message and don't allow submitting the form if the
        // browser doesn't support any of the available audio formats
        if (supportedFormats.length === 0) {
            submit.disabled = true;
            alert('The web browser in use does not support any of the' +
                  ' available audio formats. Please try with a different' +
```

```
                ' one.');
        }

        // Play the audio stream when the form is submitted successfully
        input.addEventListener('submit', function (event) {
            // Validate the fields in the form, display a message if
            // unexpected values are encountered
            if (voiceMenu.selectedIndex <= 0 || text.value.length === 0) {
                alert('Please fill in all the fields.');
            } else {
                var selectedVoice = voiceMenu
                                        .options[voiceMenu.selectedIndex]
                                        .value;

                // Point the player to the streaming server
                player.src = '/read?voiceId=' +
                    encodeURIComponent(selectedVoice) +
                    '&text=' + encodeURIComponent(text.value) +
                    '&outputFormat=' + supportedFormats[0];
                player.play();
            }

            // Stop the form from submitting,
            // Submitting the form is allowed only if the browser doesn't
            // support Javascript to ensure functionality in such a case
            event.preventDefault();
        });

        // Load the list of available voices and display them in a menu
        fetchJSON('GET', '/voices',
            // If the request succeeds
            function (voices) {
                var container = document.createDocumentFragment();

                // Build the list of options for the menu
                voices.forEach(function (voice) {
                    var option = document.createElement('option');
                    option.value = voice['Id'];
                    option.innerHTML = voice['Name'] + ' (' +
                        voice['Gender'] + ', ' +
                        voice['LanguageName'] + ')';
                    container.appendChild(option);
                });
```

```
                    // Add the options to the menu and enable the form field
                    voiceMenu.appendChild(container);
                    voiceMenu.disabled = false;
                },
                // If the request fails
                function (status, response) {
                    // Display a message in case loading data from the server
                    // fails
                    alert(status + ' - ' + response);
                });
        });

    </script>
    <style>
        #input {
            min-width: 100px;
            max-width: 600px;
            margin: 0 auto;
            padding: 50px;
        }

        #input div {
            margin-bottom: 20px;
        }

        #text {
            width: 100%;
            height: 200px;
            display: block;
        }

        #submit {
            width: 100%;
        }
    </style>
</head>

<body>
    <form id="input" method="GET" action="/read">
        <div>
            <label for="voice">Select a voice:</label>
            <select id="voice" name="voiceId" disabled>
                <option value="">Choose a voice...</option>
            </select>
```

```
        </div>
        <div>
            <label for="text">Text to read:</label>
            <textarea id="text" maxlength="1000" minlength="1" name="text"
                    placeholder="Type some text here..."></textarea>
        </div>
        <input type="submit" value="Read" id="submit" />
    </form>
    <audio id="player"></audio>
</body>

</html>
```

Python の例: Python サーバーコード (server.py)

このセクションでは、「Python の例 (HTML5 クライアントと Python サーバー)」で説明されている Python サーバーのコードを示します。

```
"""
Example Python 2.7+/3.3+ Application

This application consists of a HTTP 1.1 server using the HTTP chunked transfer
coding (https://tools.ietf.org/html/rfc2616#section-3.6.1) and a minimal HTML5
user interface that interacts with it.

The goal of this example is to start streaming the speech to the client (the
HTML5 web UI) as soon as the first consumable chunk of speech is returned in
order to start playing the audio as soon as possible.
For use cases where low latency and responsiveness are strong requirements,
this is the recommended approach.

The service documentation contains examples for non-streaming use cases where
waiting for the speech synthesis to complete and fetching the whole audio stream
at once are an option.

To test the application, run 'python server.py' and then open the URL
displayed in the terminal in a web browser (see index.html for a list of
supported browsers). The address and port for the server can be passed as
parameters to server.py. For more information, run: 'python server.py -h'
"""
from argparse import ArgumentParser
from collections import namedtuple
from contextlib import closing
```

```
from io import BytesIO
from json import dumps as json_encode
import os
import sys

if sys.version_info >= (3, 0):
    from http.server import BaseHTTPRequestHandler, HTTPServer
    from socketserver import ThreadingMixIn
    from urllib.parse import parse_qs
else:
    from BaseHTTPServer import BaseHTTPRequestHandler, HTTPServer
    from SocketServer import ThreadingMixIn
    from urlparse import parse_qs

from boto3 import Session
from botocore.exceptions import BotoCoreError, ClientError

ResponseStatus = namedtuple("HTTPStatus",
                            ["code", "message"])

ResponseData = namedtuple("ResponseData",
                          ["status", "content_type", "data_stream"])

# Mapping the output format used in the client to the content type for the
# response
AUDIO_FORMATS = {"ogg_vorbis": "audio/ogg",
                 "mp3": "audio/mpeg",
                 "pcm": "audio/wave; codecs=1"}
CHUNK_SIZE = 1024
HTTP_STATUS = {"OK": ResponseStatus(code=200, message="OK"),
               "BAD_REQUEST": ResponseStatus(code=400, message="Bad request"),
               "NOT_FOUND": ResponseStatus(code=404, message="Not found"),
               "INTERNAL_SERVER_ERROR": ResponseStatus(code=500, message="Internal
 server error")}
PROTOCOL = "http"
ROUTE_INDEX = "/index.html"
ROUTE_VOICES = "/voices"
ROUTE_READ = "/read"


# Create a client using the credentials and region defined in the adminuser
# section of the AWS credentials and configuration files
session = Session(profile_name="adminuser")
polly = session.client("polly")
```

```
class HTTPStatusError(Exception):
    """Exception wrapping a value from http.server.HTTPStatus"""

    def __init__(self, status, description=None):
        """
        Constructs an error instance from a tuple of
        (code, message, description), see http.server.HTTPStatus
        """
        super(HTTPStatusError, self).__init__()
        self.code = status.code
        self.message = status.message
        self.explain = description


class ThreadedHTTPServer(ThreadingMixIn, HTTPServer):
    """An HTTP Server that handle each request in a new thread"""
    daemon_threads = True


class ChunkedHTTPRequestHandler(BaseHTTPRequestHandler):
    """"HTTP 1.1 Chunked encoding request handler"""
    # Use HTTP 1.1 as 1.0 doesn't support chunked encoding
    protocol_version = "HTTP/1.1"

    def query_get(self, queryData, key, default=""):
        """Helper for getting values from a pre-parsed query string"""
        return queryData.get(key, [default])[0]

    def do_GET(self):
        """Handles GET requests"""

        # Extract values from the query string
        path, _, query_string = self.path.partition('?')
        query = parse_qs(query_string)

        response = None

        print(u"[START]: Received GET for %s with query: %s" % (path, query))

        try:
            # Handle the possible request paths
            if path == ROUTE_INDEX:
```

```
                response = self.route_index(path, query)
            elif path == ROUTE_VOICES:
                response = self.route_voices(path, query)
            elif path == ROUTE_READ:
                response = self.route_read(path, query)
            else:
                response = self.route_not_found(path, query)

            self.send_headers(response.status, response.content_type)
            self.stream_data(response.data_stream)

        except HTTPStatusError as err:
            # Respond with an error and log debug
            # information
            if sys.version_info >= (3, 0):
                self.send_error(err.code, err.message, err.explain)
            else:
                self.send_error(err.code, err.message)

            self.log_error(u"%s %s %s - [%d] %s", self.client_address[0],
                           self.command, self.path, err.code, err.explain)

        print("[END]")

    def route_not_found(self, path, query):
        """Handles routing for unexpected paths"""
        raise HTTPStatusError(HTTP_STATUS["NOT_FOUND"], "Page not found")

    def route_index(self, path, query):
        """Handles routing for the application's entry point'"""
        try:
            return ResponseData(status=HTTP_STATUS["OK"], content_type="text_html",
                                # Open a binary stream for reading the index
                                # HTML file
                                data_stream=open(os.path.join(sys.path[0],
                                                              path[1:]), "rb"))
        except IOError as err:
            # Couldn't open the stream
            raise HTTPStatusError(HTTP_STATUS["INTERNAL_SERVER_ERROR"],
                                  str(err))

    def route_voices(self, path, query):
        """Handles routing for listing available voices"""
        params = {}
```

```
        voices = []

        while True:
            try:
                # Request list of available voices, if a continuation token
                # was returned by the previous call then use it to continue
                # listing
                response = polly.describe_voices(**params)
            except (BotoCoreError, ClientError) as err:
                # The service returned an error
                raise HTTPStatusError(HTTP_STATUS["INTERNAL_SERVER_ERROR"],
                                      str(err))

            # Collect all the voices
            voices.extend(response.get("Voices", []))

            # If a continuation token was returned continue, stop iterating
            # otherwise
            if "NextToken" in response:
                params = {"NextToken": response["NextToken"]}
            else:
                break

        json_data = json_encode(voices)
        bytes_data = bytes(json_data, "utf-8") if sys.version_info >= (3, 0) \
            else bytes(json_data)

        return ResponseData(status=HTTP_STATUS["OK"],
                            content_type="application/json",
                            # Create a binary stream for the JSON data
                            data_stream=BytesIO(bytes_data))

    def route_read(self, path, query):
        """Handles routing for reading text (speech synthesis)"""
        # Get the parameters from the query string
        text = self.query_get(query, "text")
        voiceId = self.query_get(query, "voiceId")
        outputFormat = self.query_get(query, "outputFormat")

        # Validate the parameters, set error flag in case of unexpected
        # values
        if len(text) == 0 or len(voiceId) == 0 or \
                outputFormat not in AUDIO_FORMATS:
            raise HTTPStatusError(HTTP_STATUS["BAD_REQUEST"],
```

```
                                "Wrong parameters")
        else:
            try:
                # Request speech synthesis
                response = polly.synthesize_speech(Text=text,
                                                   VoiceId=voiceId,
                                                   OutputFormat=outputFormat,
                                                   Engine="neural")
            except (BotoCoreError, ClientError) as err:
                # The service returned an error
                raise HTTPStatusError(HTTP_STATUS["INTERNAL_SERVER_ERROR"],
                                      str(err))

            return ResponseData(status=HTTP_STATUS["OK"],
                                content_type=AUDIO_FORMATS[outputFormat],
                                # Access the audio stream in the response
                                data_stream=response.get("AudioStream"))

    def send_headers(self, status, content_type):
        """Send out the group of headers for a successful request"""
        # Send HTTP headers
        self.send_response(status.code, status.message)
        self.send_header('Content-type', content_type)
        self.send_header('Transfer-Encoding', 'chunked')
        self.send_header('Connection', 'close')
        self.end_headers()

    def stream_data(self, stream):
        """Consumes a stream in chunks to produce the response's output'"""
        print("Streaming started...")

        if stream:
            # Note: Closing the stream is important as the service throttles on
            # the number of parallel connections. Here we are using
            # contextlib.closing to ensure the close method of the stream object
            # will be called automatically at the end of the with statement's
            # scope.
            with closing(stream) as managed_stream:
                # Push out the stream's content in chunks
                while True:
                    data = managed_stream.read(CHUNK_SIZE)
                    self.wfile.write(b"%X\r\n%s\r\n" % (len(data), data))

                    # If there's no more data to read, stop streaming
```

```
                    if not data:
                        break

                # Ensure any buffered output has been transmitted and close the
                # stream
                self.wfile.flush()

            print("Streaming completed.")
        else:
            # The stream passed in is empty
            self.wfile.write(b"0\r\n\r\n")
            print("Nothing to stream.")

# Define and parse the command line arguments
cli = ArgumentParser(description='Example Python Application')
cli.add_argument(
    "-p", "--port", type=int, metavar="PORT", dest="port", default=8000)
cli.add_argument(
    "--host", type=str, metavar="HOST", dest="host", default="localhost")
arguments = cli.parse_args()

# If the module is invoked directly, initialize the application
if __name__ == '__main__':
    # Create and configure the HTTP server instance
    server = ThreadedHTTPServer((arguments.host, arguments.port),
                                ChunkedHTTPRequestHandler)
    print("Starting server, use <Ctrl-C> to stop...")
    print(u"Open {0}://{1}:{2}{3} in a web browser.".format(PROTOCOL,
                                                           arguments.host,
                                                           arguments.port,
                                                           ROUTE_INDEX))

    try:
        # Listen for requests indefinitely
        server.serve_forever()
    except KeyboardInterrupt:
        # A request to terminate has been received, stop the server
        print("\nShutting down...")
        server.socket.close()
```

Java の例

この例は、Amazon Polly を使用して Java ベースのアプリケーションでスピーチをストリームする方法を示しています。この例は AWS SDK for Java を使用して、リストから選択された音声を使用して指定したテキストを読み上げています。

表示されているコードは主なタスクをカバーしていますが、最低限のエラーチェックのみを行います。Amazon Polly がエラーに遭遇した場合、アプリケーションは終了します。

このサンプルアプリケーションを実行するには、以下のものが必要です。

- Java 8 Java Development Kit (JDK)
- AWS SDK for Java
- Apache Maven

アプリケーションをテストするには

1. JAVA_HOME 環境変数が JDKに設定されていることを確認します。

   たとえば、Windows で JDK 1.8.0_121 を `C:\Program Files\Java\jdk1.8.0_121` にインストールした場合、コマンドプロンプトで以下を入力します。

   ```
   set JAVA_HOME=""C:\Program Files\Java\jdk1.8.0_121""
   ```

   Linux に JDK 1.8.0_121を `/usr/lib/jvm/java8-openjdk-amd64` にインストールした場合、コマンドプロンプトで以下を入力してください。

   ```
   export JAVA_HOME=/usr/lib/jvm/java8-openjdk-amd64
   ```

2. コマンドラインから Maven を実行するには Maven 環境変数を設定します。

   たとえば、Windows で Maven 3.3.9を `C:\Program Files\apache-maven-3.3.9` にインストールした場合、次を入力します。

   ```
   set M2_HOME=""C:\Program Files\apache-maven-3.3.9""
   set M2=%M2_HOME%\bin
   set PATH=%M2%;%PATH%
   ```

Linux で Maven 3.3.9 を /home/ec2-user/opt/apache-maven-3.3.9 にインストールした場合は、以下を入力します。

```
export M2_HOME=/home/ec2-user/opt/apache-maven-3.3.9
export M2=$M2_HOME/bin
export PATH=$M2:$PATH
```

3. polly-java-demo という名前のディレクトリを作成します。
4. polly-java-demo ディレクトリで、pom.xml という名前のファイルを作成し、次のコードを貼り付けます。

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
                      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/
maven-4.0.0.xsd">
 <modelVersion>4.0.0</modelVersion>
 <groupId>com.amazonaws.polly</groupId>
 <artifactId>java-demo</artifactId>
 <version>0.0.1-SNAPSHOT</version>

 <dependencies>
  <!-- https://mvnrepository.com/artifact/com.amazonaws/aws-java-sdk-polly -->
  <dependency>
   <groupId>com.amazonaws</groupId>
   <artifactId>aws-java-sdk-polly</artifactId>
   <version>1.11.77</version>
  </dependency>
  <!-- https://mvnrepository.com/artifact/com.googlecode.soundlibs/jlayer -->
  <dependency>
   <groupId>com.googlecode.soundlibs</groupId>
   <artifactId>jlayer</artifactId>
   <version>1.0.1-1</version>
  </dependency>

 </dependencies>
 <build>
  <plugins>
   <plugin>
    <groupId>org.codehaus.mojo</groupId>
    <artifactId>exec-maven-plugin</artifactId>
    <version>1.2.1</version>
    <executions>
```

```
     <execution>
      <goals>
       <goal>java</goal>
      </goals>
     </execution>
    </executions>
    <configuration>
     <mainClass>com.amazonaws.demos.polly.PollyDemo</mainClass>
    </configuration>
   </plugin>
  </plugins>
 </build>
</project>
```

5. polly という名前のディレクトリを src/main/java/com/amazonaws/demos に作成します。
6. polly ディレクトリで、PollyDemo.java という名前の新しい Java ソースファイルを作成し、次のコードを貼り付けます。

```
package com.amazonaws.demos.polly;

import java.io.IOException;
import java.io.InputStream;

import com.amazonaws.ClientConfiguration;
import com.amazonaws.auth.DefaultAWSCredentialsProviderChain;
import com.amazonaws.regions.Region;
import com.amazonaws.regions.Regions;
import com.amazonaws.services.polly.AmazonPollyClient;
import com.amazonaws.services.polly.model.DescribeVoicesRequest;
import com.amazonaws.services.polly.model.DescribeVoicesResult;
import com.amazonaws.services.polly.model.OutputFormat;
import com.amazonaws.services.polly.model.SynthesizeSpeechRequest;
import com.amazonaws.services.polly.model.SynthesizeSpeechResult;
import com.amazonaws.services.polly.model.Voice;

import javazoom.jl.player.advanced.AdvancedPlayer;
import javazoom.jl.player.advanced.PlaybackEvent;
import javazoom.jl.player.advanced.PlaybackListener;

public class PollyDemo {
```

```
private final AmazonPollyClient polly;
private final Voice voice;
private static final String SAMPLE = "Congratulations. You have successfully built
this working demo
of Amazon Polly in Java. Have fun building voice enabled apps with Amazon Polly
(that's me!), and always
look at the AWS website for tips and tricks on using Amazon Polly and other great
services from AWS";

public PollyDemo(Region region) {
 // create an Amazon Polly client in a specific region
 polly = new AmazonPollyClient(new DefaultAWSCredentialsProviderChain(),
 new ClientConfiguration());
 polly.setRegion(region);
 // Create describe voices request.
 DescribeVoicesRequest describeVoicesRequest = new DescribeVoicesRequest();

 // Synchronously ask Amazon Polly to describe available TTS voices.
 DescribeVoicesResult describeVoicesResult =
polly.describeVoices(describeVoicesRequest);
 voice = describeVoicesResult.getVoices().get(0);
}

public InputStream synthesize(String text, OutputFormat format) throws IOException
{
 SynthesizeSpeechRequest synthReq =
 new SynthesizeSpeechRequest().withText(text).withVoiceId(voice.getId())
   .withOutputFormat(format).withEngine("neural");
 SynthesizeSpeechResult synthRes = polly.synthesizeSpeech(synthReq);

 return synthRes.getAudioStream();
}

public static void main(String args[]) throws Exception {
 //create the test class
 PollyDemo helloWorld = new PollyDemo(Region.getRegion(Regions.US_EAST_1));
 //get the audio stream
 InputStream speechStream = helloWorld.synthesize(SAMPLE, OutputFormat.Mp3);

 //create an MP3 player
 AdvancedPlayer player = new AdvancedPlayer(speechStream,
   javazoom.jl.player.FactoryRegistry.systemRegistry().createAudioDevice());

 player.setPlayBackListener(new PlaybackListener() {
```

```
    @Override
    public void playbackStarted(PlaybackEvent evt) {
     System.out.println("Playback started");
     System.out.println(SAMPLE);
    }

    @Override
    public void playbackFinished(PlaybackEvent evt) {
     System.out.println("Playback finished");
    }
   });


   // play it!
   player.play();

  }
}
```

7. polly-java-demo ディレクトリに戻り、そのデモをクリーン、コンパイル、そして実行します。

```
mvn clean compile exec:java
```

iOS の例

次の例では、iOS SDK for Amazon Polly を使用して、音声のリストから選択された音声で指定されたテキストを読み上げます。

ここに示すコードは、主要なタスクを実行するものであり、エラーは処理しません。完全なコードについては、AWS Mobile SDK for iOS Amazon Polly のデモを参照してください。

Initialize

```
// Region of Amazon Polly.
let AwsRegion = AWSRegionType.usEast1

// Cognito pool ID. Pool needs to be unauthenticated pool with
// Amazon Polly permissions.
let CognitoIdentityPoolId = "YourCognitoIdentityPoolId"

// Initialize the Amazon Cognito credentials provider.
let credentialProvider = AWSCognitoCredentialsProvider(regionType: AwsRegion,
 identityPoolId: CognitoIdentityPoolId)

// Create an audio player
var audioPlayer = AVPlayer()
```

使用可能な音声のリストの取得

```
// Use the configuration as default
AWSServiceManager.default().defaultServiceConfiguration = configuration

// Get all the voices (no parameters specified in input) from Amazon Polly
// This creates an async task.
let task = AWSPolly.default().describeVoices(AWSPollyDescribeVoicesInput())

// When the request is done, asynchronously do the following block
// (we ignore all the errors, but in a real-world scenario they need
// to be handled)
task.continue(successBlock: { (awsTask: AWSTask) -> Any? in
      // awsTask.result is an instance of AWSPollyDescribeVoicesOutput in
      // case of the "describeVoices" method
      let voices = (awsTask.result! as AWSPollyDescribeVoicesOutput).voices
```

```
      return nil
})
```

音声の合成

```
// First, Amazon Polly requires an input, which we need to prepare.
// Again, we ignore the errors, however this should be handled in
// real applications. Here we are using the URL Builder Request,
// since in order to make the synthesis quicker we will pass the
// presigned URL to the system audio player.
let input = AWSPollySynthesizeSpeechURLBuilderRequest()

// Text to synthesize
input.text = "Sample text"

// We expect the output in MP3 format
input.outputFormat = AWSPollyOutputFormat.mp3

// Choose the voice ID
input.voiceId = AWSPollyVoiceId.joanna

// Create an task to synthesize speech using the given synthesis input
let builder = AWSPollySynthesizeSpeechURLBuilder.default().getPreSignedURL(input)

// Request the URL for synthesis result
builder.continueOnSuccessWith(block: { (awsTask: AWSTask<NSURL>) -> Any? in
 // The result of getPresignedURL task is NSURL.
 // Again, we ignore the errors in the example.
 let url = awsTask.result!

 // Try playing the data using the system AVAudioPlayer
 self.audioPlayer.replaceCurrentItem(with: AVPlayerItem(url: url as URL))
 self.audioPlayer.play()

 return nil
})
```

Android の例

次の例では、Android SDK for Amazon Polly を使用して、音声のリストから選択された音声で指定されたテキストを読み上げます。

ここに示すコードは、主要なタスクを実行するものであり、エラーは処理しません。完全なコードについては、AWS Mobile SDK for Android Amazon Polly のデモを参照してください。

Initialize

```
// Cognito pool ID. Pool needs to be unauthenticated pool with
// Amazon Polly permissions.
String COGNITO_POOL_ID = "YourCognitoIdentityPoolId";

// Region of Amazon Polly.
Regions MY_REGION = Regions.US_EAST_1;

// Initialize the Amazon Cognito credentials provider.
CognitoCachingCredentialsProvider credentialsProvider = new
 CognitoCachingCredentialsProvider(
        getApplicationContext(),
        COGNITO_POOL_ID,
        MY_REGION
);

// Create a client that supports generation of presigned URLs.
AmazonPollyPresigningClient client = new
 AmazonPollyPresigningClient(credentialsProvider);
```

使用可能な音声のリストの取得

```
// Create describe voices request.
DescribeVoicesRequest describeVoicesRequest = new DescribeVoicesRequest();

// Synchronously ask Amazon Polly to describe available TTS voices.
DescribeVoicesResult describeVoicesResult =
 client.describeVoices(describeVoicesRequest);
List<Voice> voices = describeVoicesResult.getVoices();
```

音声ストリームの URL の取得

```
// Create speech synthesis request.
SynthesizeSpeechPresignRequest synthesizeSpeechPresignRequest =
        new SynthesizeSpeechPresignRequest()
        // Set the text to synthesize.
        .withText("Hello world!")
        // Select voice for synthesis.
        .withVoiceId(voices.get(0).getId()) // "Joanna"
        // Set format to MP3.
        .withOutputFormat(OutputFormat.Mp3);

// Get the presigned URL for synthesized speech audio stream.
URL presignedSynthesizeSpeechUrl =
        client.getPresignedSynthesizeSpeechUrl(synthesizeSpeechPresignRequest);
```

合成された音声の再生

```
// Use MediaPlayer: https://developer.android.com/guide/topics/media/mediaplayer.html

// Create a media player to play the synthesized audio stream.
MediaPlayer mediaPlayer = new MediaPlayer();
mediaPlayer.setAudioStreamType(AudioManager.STREAM_MUSIC);

try {
    // Set media player's data source to previously obtained URL.
    mediaPlayer.setDataSource(presignedSynthesizeSpeechUrl.toString());
} catch (IOException e) {
    Log.e(TAG, "Unable to set data source for the media player! " + e.getMessage());
}

// Prepare the MediaPlayer asynchronously (since the data source is a network stream).
mediaPlayer.prepareAsync();

// Set the callback to start the MediaPlayer when it's prepared.
mediaPlayer.setOnPreparedListener(new MediaPlayer.OnPreparedListener() {
    @Override
    public void onPrepared(MediaPlayer mp) {
        mp.start();
    }
});

// Set the callback to release the MediaPlayer after playback is completed.
mediaPlayer.setOnCompletionListener(new MediaPlayer.OnCompletionListener() {
```

```
    @Override
    public void onCompletion(MediaPlayer mp) {
 mp.release();
    }
});
```

Amazon Polly のクォータ

Amazon Polly は、過剰なリクエストを拒否することでカスタマートラフィックにクォータを適用します。標準音声の SynthesizeSpeech リクエストのデフォルトクォータは、1 つの AWS アカウントで、1 つのリージョンにおいて 1 秒あたり 80 件のトランザクション (tps) です。制限が引き上げられず、標準の音声を使用して 1 秒あたり 100 件の SynthesizeSpeech リクエストを生成した場合、1 秒あたり 80 件のリクエストが成功し、1 秒あたり 20 件のリクエストが Amazon Polly によってスロットリングされます。これらのリクエストにより、HTTP ステータス 400 のレスポンスと、ThrottlingException を示すレスポンスヘッダーが返されます。また、Amazon Polly はリクエストレートに基づいてすべてのオペレーションへのトラフィックをスロットリングします。

音声合成制限の例

- 英語のアルファベットの最初の 24 文字を 1 文字ずつ合成する。各文字の合成にかかる時間が 50 ミリ秒未満で、オペレーションの制限が 8 tps の場合、24 文字の合成には少なくとも 3 秒かかります。その間、1 秒あたり最大 8 文字を合成できます。それ以降のリクエストはスロットリングされます。リクエストは短時間しか続かないため、重複することなく連続して合成されます。
- 16 段落のテキストを合成する。各段落が合成され、クライアント側で 2 秒以内にすべて受信され、同時リクエスト数のオペレーションの制限が 8 件の場合、16 件の記事すべてを合成するには少なくとも 4 秒かかります。最初の 1 秒で、最大 8 件のリクエストを開始できます。同時リクエスト中は、同時実行数の制限により、新しい合成を開始しようとしてもスロットリングされます。最初の 2 秒間、つまり最初のリクエストのバッチが終了した後に、残りの 8 つの段落を合成できるようになります。

Amazon Polly を使用するときは、以下の制限に注意してください。

トピック

サポートされるリージョン

Amazon Polly が利用可能な AWS リージョンのリストについては、「」のAmazon Polly エンドポイントとクォータ」を参照してくださいAmazon Web Services 全般のリファレンス。

- 生成音声をサポートするリージョンについては、「生成音声」を参照してください。
- ロングフォーム音声をサポートするリージョンについては、「ロングフォーム音声」を参照してください。
- ニューラル音声をサポートするリージョンについては、ニューラル TTS の「the section called “機能とリージョンの互換性”」を参照してください。

クォータとスロットルレート

次の表では、Amazon Polly オペレーションごとのスロットルレートが定義されています。を使用して AWS Management Console 、必要に応じて調整可能なクォータのクォータ引き上げをリクエストできます。

操作	制限
レキシコン	

`DeleteLexicon`
`PutLexicon`
`GetLexicon`
`ListLexicons`
	これらのオペレーションによる 2 トランザクション/秒 (tps) はすべて結合されます。
	最大許容バーストは 4 tps です。

音声	
`DescribeVoices`	80 tps で、バースト制限は 100 tps
`SynthesizeSpeech`	生成音声: 8 tps ロングフォーム音声: 8 tps、バースト制限は 10 tps ニューラル音声: 8 tps、バースト制限は 10 tps

操作	制限
	標準音声: 80 tps、バースト制限は 100 tps
StartSpeechSynthesisTask	生成音声: 1 tps ロングフォーム音声: 1 tps ニューラル音声: 1 tps 標準音声: 10 tps、バースト制限は 12 tps
GetSynthesizeSpeechTask および ListSynthesizeSpeechTask	組み合わせの最大許容は 10 tps

同時実行リクエスト

生成音声 の場合、Amazon Polly は最大 26 件の同時リクエストをサポートします。[ロングフォーム音声] の場合、Amazon Polly は最大 26 件の同時リクエストに対応しています。ニューラル音声の場合、Amazon Polly は 8 tps およびバースト制限の 10 tps で、最大 18 件の同時リクエストに対応しています。Amazon Polly は同時リクエストの制限もサポートしています。標準の音声の場合、Amazon Polly は 80 tps で最大 80 件の同時リクエストに対応しています。

スロットリングを軽減するためのベストプラクティス

- バックオフとジッターを使用してスロットリングを再試行することで、負荷を短期間で分散させ、可用性を損なうことなく使用量の予期しないピークに対処できます。 AWS Code Sample Catalog は多くのプログラミング言語でこれをデフォルトで行うように既に設定されています。詳細については、「機能のリトライ動作」を参照してください。
- Amazon Polly メトリクスを使用する。Amazon Polly は に自動的に発行して CloudWatch 、現在の使用状況を分析し、使用量の増加を予測します。

> **Note**
>
> クォータの増額をリクエストする前に (該当する場合)、このページのガイドラインに従って必要な TPS を計算してください。Amazon Polly は、コストを低く抑えるために、顧客の需要に応じて必要なコンピューティングリソースのみを確保します。

発音レキシコン

- アカウントにつき最大 100 個のレキシコンを保存できます。
- レキシコン名は、長さが最大 20 文字の英数字文字列です。
- 各レキシコンのサイズは最大 40,000 文字です。(レキシコンのサイズは SynthesizeSpeech オペレーションのレイテンシーに影響することに注意してください)。
- レキシコンの <phoneme> または <alias> は最大 100 文字と置き換えることができます。

レキシコンの使用については、「レキシコンの管理」を参照してください。

SynthesizeSpeech API オペレーション

SynthesizeSpeech の使用量を見積もるとき、Amazon Polly によって生成された音声は通常、特にインタラクティブアプリケーションを使用する場合、再生に少なくとも数秒かかることに注意してください。これにより、同時コンシューマー数が多い場合は、SynthesizeSpeech へのリクエストの速度が低下します。さらに、Amazon Polly では、合成する同時リクエストの数に応じて SynthesizeSpeech リクエストをスロットリングします。同時リクエストを個別に設定することはできません。同時リクエスト数の上限は常に、許容される tps 数と同じ値で、これに合わせてスケールされます。

短いストーリーのサンプルアプリケーション。Amazon Polly を使用すると、一連の短いストーリーを再生するアプリケーションを作成できます。この種類のアプリケーションでは、ユーザーがアプリケーションを終了するまでは、最初のストーリーが再生され、続いて後続のストーリーが再生されます。各ストーリーの合成には約 0.5 秒かかり、再生には 10 秒かかります。このシナリオでは、顧客がアプリケーションを使用して 10 秒経過するたびに SynthesizeSpeech が 1 回の呼び出されることを想定しています。これは、アプリケーションを同時に使用している顧客 10 人ごとに 1 秒あたり 1 回の呼び出しがあることになります。1000 人の顧客が同時にアプリケーションを使用している場合、SynthesizeSpeech への平均コールレートは 1 秒あたりのトランザクション数は 100 件程度になると予想できます。

SynthesizeSpeech API オペレーションの使用には、以下の制限が関連している点に注意してください。

- 入力テキストのサイズは、最大 3000 課金対象文字 (合計 6000 文字) です。SSML タグは、課金対象文字としてカウントされません。
- 入力テキストに適用する最大 5 個のレキシコンを指定できます。
- 出力オーディオストリーム (合成) は 10 分に制限されています。これに達した後は、残りの音声はカットオフされます。

詳細については、「SynthesizeSpeech」を参照してください。

> **Note**
>
> SynthesizeSpeech API オペレーションのいくつかの制限は、StartSythensizeSpeechTask API を使用して回避することができます。詳細については、「長いオーディオファイルの作成」を参照してください。

SpeechSynthesisTask API オペレーション

StartSpeechSynthesisTask、GetSpeechSynthesisTask、および ListSpeechSynthesisTasks API オペレーションの使用には、以下の制限が関連している点に注意してください。

- 入力テキストのサイズは、最大 100,000 課金対象文字 (合計 200,000 文字) です。SSML タグは、課金対象文字としてカウントされません。
- 入力テキストに適用する最大 5 個のレキシコンを指定できます。

音声合成マークアップ言語 (SSML)

SSML の使用には、以下の制限が関連している点に注意してください。

- <audio>、<lexicon>、<lookup>、および <voice> タグは、サポートされていません。
- <break> エレメントは、それぞれ最大 10 秒の時間を指定できます。
- <prosody> タグでは、-80% より小さいレート属性値はサポートされていません。

詳細については、「SSML ドキュメントからの音声の生成」を参照してください。

Amazon Polly のセキュリティ

のクラウドセキュリティが最優先事項 AWS です。お客様は AWS 、セキュリティを最も重視する組織の要件を満たすように構築されたデータセンターとネットワークアーキテクチャからメリットを得られます。

セキュリティは、 AWS とユーザーの間で共有される責任です。責任共有モデルではこれを、クラウドのセキュリティ、およびクラウド内でのセキュリティと説明しています:

- クラウドのセキュリティ — クラウドで AWS サービスを実行するインフラストラクチャを保護する責任 AWS は AWS にあります。 AWS また、 は、安全に使用できるサービスも提供します。コンプライアンスAWS プログラムコンプライアンスプログラム の一環として、サードパーティーの監査者は定期的にセキュリティの有効性をテストおよび検証。Amazon Polly に適用されるコンプライアンスプログラムの詳細については、「コンプライアンスプログラムAWS による対象範囲内のサービスコンプライアンスプログラム」を参照してください。
- クラウドのセキュリティ — お客様の責任は、使用する AWS サービスによって決まります。また、お客様は、お客様のデータの機密性、企業の要件、および適用可能な法律および規制などの他の要因についても責任を担います。

このドキュメントは、Amazon Polly 使用時における責任共有モデルの適用法を理解するのに役立ちます。以下のトピックでは、セキュリティやコンプライアンスに関する目的を果たせるように Amazon Polly を設定する方法について説明します。また、Amazon Polly リソースのモニタリングや保護に役立つ他の AWS のサービスの使用方法についても説明します。

トピック

Amazon Polly のデータ保護

Amazon Polly は AWS 、 責任共有モデル に準拠しています。 には、データ保護に関する規制とガイドラインが含まれています。 AWS は、すべての AWS サービスを実行するグローバルインフラストラクチャを保護する責任を負います。 AWS は、このインフラストラクチャでホストされるデータの制御を維持します。 これには、顧客コンテンツと個人データを処理するためのセキュリティ設定コントロール、 AWS 顧客および APN パートナーが含まれます。 データコントローラーまたはデータ処理者として動作し、 は、 AWS クラウドに保存する個人データについて責任を負います。

データ保護の目的で、 AWS アカウント認証情報を保護し、 AWS Identity and Access Management (IAM) を使用して個々のユーザーをセットアップして、各ユーザーに各自の職務を果たすために必要なアクセス許可のみを付与することをお勧めします。また、次の方法でデータを保護することもお勧めします:

- 各アカウントで多要素認証 (MFA) を使用します。
- SSL/TLS を使用して AWS リソースと通信します。
- で API とユーザーアクティビティのログ記録を設定します AWS CloudTrail。
- AWS 暗号化ソリューションと、 サービス内のすべての AWS デフォルトのセキュリティコントロールを使用します。

顧客のアカウント番号などの機密の識別情報は、[名前] フィールドなどの自由形式のフィールドに配置しないことを強くお勧めします。これは、コンソール、API、または AWS CLI SDK を使用して Amazon Polly または他の AWS のサービスを使用する場合も同様です。 AWS SDKs Amazon Polly や他のサービスに入力したすべてのデータは、診断ログに取り込まれる可能性があります。外部サーバーへの URL を指定するときは、そのサーバーへのリクエストを検証するための認証情報を URL に含めないでください。

データ保護の詳細については、AWS セキュリティブログ のブログ投稿「AWS の責任共有モデルと GDPR」を参照してください。

保管時の暗号化

Amazon Polly 音声合成の出力は独自のシステムに保存できます。Amazon Polly を呼び出して、選択した暗号化キーでファイルを暗号化し、Amazon Simple Storage Service (Amazon S3) または別の安全なストレージに保存することもできます。Amazon Polly の the section called “SynthesizeSpeech” オペレーションはステートレスであり、カスタマー ID と関連付けられていません。後で Amazon Polly から取得することはできません。

転送時の暗号化

すべてのテキスト提出物は、転送中に Secure Sockets Layer (SSL) によって保護され、Amazon Polly はテキスト提出物の内容を保持しません。

インターネットトラフィックのプライバシー

Amazon Polly へのアクセスは、 AWS コンソール、CLI、または SDKs。通信では、機密性のために Transport Layer Security (TLS) セッション暗号化を利用し、認証と整合性のためにデジタル署名を利用します。

Amazon Polly での Identity and Access Management

AWS Identity and Access Management (IAM) は、管理者が AWS リソースへのアクセスを安全に制御 AWS のサービス するのに役立つ です。IAM 管理者は、誰を認証 (サインイン) し、誰に Amazon Polly リソースの使用を承認する (アクセス許可を付与する) かを制御します。IAM は、追加料金なしで AWS のサービス 使用できる です。

トピック

- 対象者
- アイデンティティを使用した認証
- ポリシーを使用したアクセスの管理
- Amazon Polly で IAM を使用する方法
- Amazon Polly のアイデンティティベースポリシーの例
- Amazon Polly API アクセス許可: アクション、アクセス許可、リソースの参照
- Amazon Polly の ID とアクセスのトラブルシューティング

対象者

AWS Identity and Access Management (IAM) の使用方法は、Amazon Polly で行う作業によって異なります。

サービスユーザー – ジョブを実行するために Amazon Polly サービスを使用する場合は、管理者から必要なアクセス許可と認証情報が与えられます。業務のために使用する Amazon Polly 機能が増えるにつれて、追加の許可が必要になる可能性があります。アクセスの管理方法を理解しておくと、管理者に適切な許可をリクエストするうえで役立ちます。Amazon Polly の機能にアクセスできない場合は、「Amazon Polly の ID とアクセスのトラブルシューティング」を参照してください。

サービス管理者 – 社内の Amazon Polly リソースを担当している場合は、Amazon Polly に対する完全なアクセス権があると思われます。サービスのユーザーがどの Amazon Polly 機能やリソースにアクセスするかを決めるのは管理者の仕事です。その後、IAM 管理者にリクエストを送信して、サービスユーザーの権限を変更する必要があります。このページの情報を点検して、IAM の基本概念を理解してください。会社で Amazon Polly と IAM を併用する方法の詳細については、Amazon Polly で IAM を使用する方法を参照してください。

IAM 管理者 – IAM 管理者には、Amazon Polly へのアクセスを管理するポリシーの作成方法の詳細を理解することが推奨されます。IAM で使用できる Amazon Polly アイデンティティベースのポリシーの例を表示するには、Amazon Polly のアイデンティティベースポリシーの例 を参照してください。

アイデンティティを使用した認証

認証とは、ID 認証情報 AWS を使用して にサインインする方法です。として、IAM ユーザーとして AWS アカウントのルートユーザー、または IAM ロールを引き受けて認証（ にサインイン AWS) される必要があります。

ID ソースを介して提供された認証情報を使用して、フェデレーティッド ID AWS として にサインインできます。 AWS IAM Identity Center（ IAM Identity Center) ユーザー、会社のシングルサインオン認証、Google または Facebook の認証情報は、フェデレーション ID の例です。フェデレーティッドアイデンティティとしてサインインする場合、IAM ロールを使用して、前もって管理者により ID フェデレーションが設定されています。フェデレーション AWS を使用して にアクセスすると、間接的にロールを引き受けることになります。

ユーザーのタイプに応じて、 AWS Management Console または AWS アクセスポータルにサインインできます。へのサインインの詳細については AWS、「 ユーザーガイド」の「 へのサインイン AWS アカウント方法AWS サインイン 」を参照してください。

AWS プログラムで にアクセスする場合、 は Software Development Kit (SDK) とコマンドラインインターフェイス (CLI) AWS を提供し、認証情報を使用してリクエストに暗号で署名します。 AWS ツールを使用しない場合は、リクエストに自分で署名する必要があります。推奨される方法を使用してリクエストを自分で署名する方法の詳細については、IAM ユーザーガイドの API AWS リクエストの署名を参照してください。

使用する認証方法を問わず、追加セキュリティ情報の提供をリクエストされる場合もあります。例えば、 AWS では、アカウントのセキュリティを強化するために多要素認証 (MFA) を使用することをお勧めします。詳細については、『AWS IAM Identity Center ユーザーガイド』の「Multi-factor authentication」(多要素認証) および『IAM ユーザーガイド』の「AWSにおける多要素認証 (MFA) の使用」を参照してください。

AWS アカウント ルートユーザー

を作成するときは AWS アカウント、アカウントのすべての AWS のサービス およびリソースへの完全なアクセス権を持つ 1 つのサインインアイデンティティから始めます。この ID は AWS アカウント ルートユーザーと呼ばれ、アカウントの作成に使用した E メールアドレスとパスワードでサインインすることでアクセスできます。日常的なタスクには、ルートユーザーを使用しないことを強くお勧めします。ルートユーザーの認証情報は保護し、ルートユーザーでしか実行できないタスクを実行するときに使用します。ルートユーザーとしてサインインする必要があるタスクの完全なリストについては、『IAM ユーザーガイド』の「ルートユーザー認証情報が必要なタスク」を参照してください。

フェデレーティッドアイデンティティ

ベストプラクティスとして、管理者アクセスを必要とするユーザーを含む人間のユーザーに、一時的な認証情報を使用して にアクセスするための ID プロバイダーとのフェデレーションの使用を要求 AWS のサービス します。

フェデレーティッド ID は、エンタープライズユーザーディレクトリ、ウェブ ID プロバイダー、AWS Directory Service、アイデンティティセンターディレクトリのユーザー、または ID ソースを通じて提供された認証情報 AWS のサービス を使用して にアクセスするユーザーです。フェデレーティッド ID が にアクセスすると AWS アカウント、ロールを引き受け、ロールは一時的な認証情報を提供します。

アクセスを一元管理する場合は、 AWS IAM Identity Centerを使用することをお勧めします。IAM Identity Center でユーザーとグループを作成することも、独自の ID ソース内のユーザーとグループのセットに接続して同期して、すべての AWS アカウント とアプリケーションで使用することもできます。IAM Identity Center の詳細については、『AWS IAM Identity Center ユーザーガイド』の「What is IAM Identity Center?」(IAM Identity Center とは) を参照してください。

IAM ユーザーとグループ

IAM ユーザーは、単一のユーザーまたはアプリケーションに対して特定のアクセス許可 AWS アカウント を持つ 内のアイデンティティです。可能であれば、パスワードやアクセスキーなどの長期的な認証情報を保有する IAM ユーザーを作成する代わりに、一時認証情報を使用することをお勧めします。ただし、IAM ユーザーでの長期的な認証情報が必要な特定のユースケースがある場合は、アクセスキーをローテーションすることをお勧めします。詳細については、「IAM ユーザーガイド」の「長期的な認証情報を必要とするユースケースのためにアクセスキーを定期的にローテーションする」を参照してください。

IAM グループは、IAM ユーザーの集団を指定するアイデンティティです。グループとしてサインインすることはできません。グループを使用して、複数のユーザーに対して一度に権限を指定できます。多数のユーザーグループがある場合、グループを使用することで権限の管理が容易になります。例えば、IAMAdmins という名前のグループを設定して、そのグループに IAM リソースを管理する権限を与えることができます。

ユーザーは、ロールとは異なります。ユーザーは 1 人の人または 1 つのアプリケーションに一意に関連付けられますが、ロールはそれを必要とする任意の人が引き受けるようになっています。ユーザーには永続的な長期の認証情報がありますが、ロールでは一時的な認証情報が提供されます。詳細については、『IAM ユーザーガイド』の「IAM ユーザー (ロールではなく) の作成が適している場合」を参照してください。

IAM ロール

IAM ロールは、特定のアクセス許可 AWS アカウント を持つ 内のアイデンティティです。これは IAM ユーザーに似ていますが、特定のユーザーには関連付けられていません。ロール を切り替える AWS Management Console ことで、 で IAM ロールを一時的に引き受けることができます。ロールを引き受けるには、 または AWS API AWS CLI オペレーションを呼び出すか、カスタム URL を使用します。ロールを使用する方法の詳細については、「IAM ユーザーガイド」の「IAM ロールの使用」を参照してください。

IAM ロールと一時的な認証情報は、次の状況で役立ちます:

- フェデレーションユーザーアクセス – フェデレーティッドアイデンティティに権限を割り当てるには、ロールを作成してそのロールの権限を定義します。フェデレーティッドアイデンティティが認証されると、そのアイデンティティはロールに関連付けられ、ロールで定義されている権限が付与されます。フェデレーションの詳細については、『IAM ユーザーガイド』の「サードパーティーアイデンティティプロバイダー向けロールの作成」 を参照してください。IAM アイデンティティセンターを使用する場合、権限セットを設定します。アイデンティティが認証後にアクセスできるものを制御するため、IAM Identity Center は、権限セットを IAM のロールに関連付けます。権限セットの詳細については、『AWS IAM Identity Center ユーザーガイド』の「権限セット」を参照してください。
- 一時的な IAM ユーザー権限 - IAM ユーザーまたはロールは、特定のタスクに対して複数の異なる権限を一時的に IAM ロールで引き受けることができます。
- クロスアカウントアクセス - IAM ロールを使用して、自分のアカウントのリソースにアクセスすることを、別のアカウントの人物 (信頼済みプリンシパル) に許可できます。クロスアカウントアクセス権を付与する主な方法は、ロールを使用することです。ただし、一部の では AWS のサービス、 (プロキシとしてロールを使用する代わりに) リソースにポリシーを直接アタッチできま

す。クロスアカウントアクセスにおけるロールとリソースベースのポリシーの違いについては、『IAM ユーザーガイド』の「IAM ロールとリソースベースのポリシーとの相違点」を参照してください。

- クロスサービスアクセス — 一部の は、他の の機能 AWS のサービス を使用します AWS のサービス。例えば、あるサービスで呼び出しを行うと、通常そのサービスによって Amazon EC2 でアプリケーションが実行されたり、Amazon S3 にオブジェクトが保存されたりします。サービスでは、呼び出し元プリンシパルの権限、サービスロール、またはサービスにリンクされたロールを使用してこれを行う場合があります。
  - 転送アクセスセッション (FAS) – IAM ユーザーまたはロールを使用して でアクションを実行する場合 AWS、ユーザーはプリンシパルと見なされます。一部のサービスを使用する際に、アクションを実行することで、別のサービスの別のアクションがトリガーされることがあります。FAS は、 を呼び出すプリンシパルのアクセス許可を AWS のサービス、ダウンストリームサービス AWS のサービス へのリクエストのリクエストと組み合わせて使用します。FAS リクエストは、サービスが他の AWS のサービス またはリソースとのやり取りを完了する必要があるリクエストを受け取った場合にのみ行われます。この場合、両方のアクションを実行するためのアクセス許可が必要です。FAS リクエストを行う際のポリシーの詳細については、「転送アクセスセッション」を参照してください。
  - サービスロール - サービスがユーザーに代わってアクションを実行するために引き受ける IAM ロールです。IAM 管理者は、IAM 内からサービスロールを作成、変更、削除できます。詳細については、「IAM ユーザーガイド」の「AWS のサービスに権限を委任するロールの作成」を参照してください。
  - サービスにリンクされたロール – サービスにリンクされたロールは、 にリンクされたサービスロールの一種です AWS のサービス。サービスは、ユーザーに代わってアクションを実行するロールを引き受けることができます。サービスにリンクされたロールは に表示され AWS アカウント 、サービスによって所有されます。IAM 管理者は、サービスにリンクされたロールの権限を表示できますが、編集することはできません。
- Amazon EC2 で実行されているアプリケーション – IAM ロールを使用して、EC2 インスタンスで実行され、 AWS CLI または AWS API リクエストを行うアプリケーションの一時的な認証情報を管理できます。これは、EC2 インスタンス内でのアクセスキーの保存に推奨されます。 AWS ロールを EC2 インスタンスに割り当て、そのすべてのアプリケーションで使用できるようにするには、インスタンスにアタッチされたインスタンスプロファイルを作成します。インスタンスプロファイルにはロールが含まれ、EC2 インスタンスで実行されるプログラムは一時的な認証情報を取得できます。詳細については、『IAM ユーザーガイド』の「Amazon EC2 インスタンスで実行されるアプリケーションに IAM ロールを使用して権限を付与する」を参照してください。

IAM ロールと IAM ユーザーのどちらを使用するかについては、『IAM ユーザーガイド』の「(IAM ユーザーではなく) IAM ロールをいつ作成したら良いのか?」を参照してください。

ポリシーを使用したアクセスの管理

でアクセスを制御する AWS には、ポリシーを作成し、 AWS ID またはリソースにアタッチします。ポリシーは AWS 、ID またはリソースに関連付けられているときにアクセス許可を定義する のオブジェクトです。 は、プリンシパル (ユーザー、ルートユーザー、またはロールセッション) がリクエストを行うときに、これらのポリシー AWS を評価します。ポリシーでの権限により、リクエストが許可されるか拒否されるかが決まります。ほとんどのポリシーは JSON ドキュメント AWS としてに保存されます。JSON ポリシードキュメントの構造と内容の詳細については、「IAM ユーザーガイド」の「JSON ポリシー概要」を参照してください。

管理者は AWS JSON ポリシーを使用して、誰が何にアクセスできるかを指定できます。つまり、どのプリンシパルがどんなリソースにどんな条件でアクションを実行できるかということです。

デフォルトでは、ユーザーやロールに権限はありません。IAM 管理者は、リソースで必要なアクションを実行するための権限をユーザーに付与する IAM ポリシーを作成できます。その後、管理者はロールに IAM ポリシーを追加し、ユーザーはロールを引き継ぐことができます。

IAM ポリシーは、オペレーションの実行方法を問わず、アクションの権限を定義します。例えば、`iam:GetRole` アクションを許可するポリシーがあるとします。そのポリシーを持つユーザーは、 AWS Management Console、、 AWS CLIまたは AWS API からロール情報を取得できます。

アイデンティティベースのポリシー

アイデンティティベースポリシーは、IAM ユーザー、ユーザーのグループ、ロールなど、アイデンティティにアタッチできる JSON 権限ポリシードキュメントです。これらのポリシーは、ユーザーとロールが実行できるアクション、リソース、および条件をコントロールします。アイデンティティベースのポリシーを作成する方法については、「IAM ユーザーガイド」の「IAM ポリシーの作成」を参照してください。

アイデンティティベースポリシーは、さらにインラインポリシーまたはマネージドポリシーに分類できます。インラインポリシーは、単一のユーザー、グループ、またはロールに直接埋め込まれています。管理ポリシーは、 内の複数のユーザー、グループ、ロールにアタッチできるスタンドアロンポリシーです AWS アカウント。管理ポリシーには、 AWS 管理ポリシーとカスタマー管理ポリシーが含まれます。マネージドポリシーまたはインラインポリシーのいずれかを選択する方法については、『IAM ユーザーガイド』の「マネージドポリシーとインラインポリシーの比較」を参照してください。

リソースベースのポリシー

リソースベースのポリシーは、リソースに添付する JSON ポリシードキュメントです。リソースベースのポリシーには例として、IAM ロールの信頼ポリシーや Amazon S3 バケットポリシーがあげられます。リソースベースのポリシーをサポートするサービスでは、サービス管理者はポリシーを使用して特定のリソースへのアクセスを制御できます。ポリシーがアタッチされているリソースの場合、指定されたプリンシパルがそのリソースに対して実行できるアクションと条件は、ポリシーによって定義されます。リソースベースのポリシーでは、プリンシパルを指定する必要があります。プリンシパルには、アカウント、ユーザー、ロール、フェデレーティッドユーザー、または を含めることができます AWS のサービス。

リソースベースのポリシーは、そのサービス内にあるインラインポリシーです。リソースベースのポリシーでは、IAM の AWS マネージドポリシーを使用できません。

アクセスコントロールリスト (ACL)

アクセスコントロールリスト (ACL) は、どのプリンシパル (アカウントメンバー、ユーザー、またはロール) がリソースにアクセスするための権限を持つかをコントロールします。ACL はリソースベースのポリシーに似ていますが、JSON ポリシードキュメント形式は使用しません。

Amazon S3、 AWS WAF、および Amazon VPC は、ACLs。ACL の詳細については、『Amazon Simple Storage Service デベロッパーガイド』の「アクセスコントロールリスト (ACL) の概要」を参照してください。

その他のポリシータイプ

AWS は、一般的ではない追加のポリシータイプをサポートします。これらのポリシータイプでは、より一般的なポリシータイプで付与された最大の権限を設定できます。

- アクセス許可の境界 - アクセス許可の境界は、アイデンティティベースのポリシーによって IAM エンティティ (IAM ユーザーまたはロール) に付与できる権限の上限を設定する高度な機能です。エンティティにアクセス許可の境界を設定できます。結果として得られる権限は、エンティティのアイデンティティベースポリシーとそのアクセス許可の境界の共通部分になります。`Principal` フィールドでユーザーまたはロールを指定するリソースベースのポリシーでは、アクセス許可の境界は制限されません。これらのポリシーのいずれかを明示的に拒否した場合、権限は無効になります。許可の境界の詳細については、「IAM ユーザーガイド」の「IAM エンティティの許可の境界」を参照してください。
- サービスコントロールポリシー (SCPs) – SCPs は、 の組織または組織単位 (OU) に対する最大アクセス許可を指定する JSON ポリシーです AWS Organizations。 AWS Organizations は、 AWS

アカウント ビジネスが所有する複数の をグループ化して一元管理するためのサービスです。組織内のすべての機能を有効にすると、サービスコントロールポリシー (SCP) を一部またはすべてのアカウントに適用できます。SCP は、各 を含むメンバーアカウントのエンティティのアクセス許可を制限します AWS アカウントのルートユーザー。Organizations と SCP の詳細については、『AWS Organizations ユーザーガイド』の「SCP の仕組み」を参照してください。

- セッションポリシー - セッションポリシーは、ロールまたはフェデレーションユーザーの一時的なセッションをプログラムで作成する際にパラメータとして渡す高度なポリシーです。結果としてセッションの権限は、ユーザーまたはロールのアイデンティティベースポリシーとセッションポリシーの共通部分になります。また、リソースベースのポリシーから権限が派生する場合もあります。これらのポリシーのいずれかを明示的に拒否した場合、権限は無効になります。詳細については、「IAM ユーザーガイド」の「セッションポリシー」を参照してください。

複数のポリシータイプ

1 つのリクエストに複数のタイプのポリシーが適用されると、結果として作成される権限を理解するのがさらに難しくなります。複数のポリシータイプが関与する場合にリクエストを許可するかどうか AWS を決定する方法については、IAM ユーザーガイドの「ポリシー評価ロジック」を参照してください。

Amazon Polly で IAM を使用する方法

IAM を使用して Amazon Polly へのアクセスを管理する前に、Amazon Polly で使用できる IAM 機能について理解しておく必要があります。

Amazon Polly で使用できる IAM の機能

IAM 機能	Amazon Polly のサポート
アイデンティティベースのポリシー	Yes
リソースベースのポリシー	No
ポリシーアクション	Yes
ポリシーリソース	はい
ポリシー条件キー (サポート固有)	いいえ
ACL	No

IAM 機能	Amazon Polly のサポート
ABAC (ポリシー内のタグ)	いいえ
一時的な認証情報	はい
Amazon Polly の転送アクセスセッション (FAS)	はい
サービスロール	いいえ
サービスリンクロール	いいえ

Amazon Polly およびその他の AWS のサービスがほとんどの IAM 機能と連携する方法の概要を把握するには、「IAM ユーザーガイド」のAWS 「IAM と連携する のサービス」を参照してください。

Amazon Polly の ID ベースのポリシー

アイデンティティベースポリシーをサポートする	Yes

アイデンティティベースポリシーは、IAM ユーザー、ユーザーグループ、ロールなど、アイデンティティにアタッチできる JSON 権限ポリシードキュメントです。これらのポリシーは、ユーザーとロールが実行できるアクション、リソース、および条件をコントロールします。アイデンティティベースのポリシーを作成する方法については、『IAM ユーザーガイド』の「IAM ポリシーの作成」を参照してください。

IAM アイデンティティベースのポリシーでは、許可または拒否するアクションとリソース、およびアクションを許可または拒否する条件を指定できます。プリンシパルは、それが添付されているユーザーまたはロールに適用されるため、アイデンティティベースのポリシーでは指定できません。JSON ポリシーで使用できるすべての要素については、「IAM ユーザーガイド」の「IAM JSON ポリシーの要素のリファレンス」を参照してください。

Amazon Polly のアイデンティティベースポリシーの例

Amazon Polly でのアイデンティティベースのポリシーの例は、Amazon Polly のアイデンティティベースポリシーの例でご確認ください。

Amazon Polly 内のリソースベースのポリシー

リソースベースのポリシーのサポート	No

リソースベースのポリシーは、リソースに添付する JSON ポリシードキュメントです。リソースベースのポリシーには例として、IAM ロールの信頼ポリシーや Amazon S3 バケットポリシーがあげられます。リソースベースのポリシーをサポートするサービスでは、サービス管理者はポリシーを使用して特定のリソースへのアクセスを制御できます。ポリシーがアタッチされているリソースの場合、指定されたプリンシパルがそのリソースに対して実行できるアクションと条件は、ポリシーによって定義されます。リソースベースのポリシーでは、プリンシパルを指定する必要があります。プリンシパルには、アカウント、ユーザー、ロール、フェデレーティッドユーザー、または を含めることができます AWS のサービス。

クロスアカウントアクセスを有効にするには、アカウント全体、または別のアカウントの IAM エンティティをリソースベースのポリシーのプリンシパルとして指定します。リソースベースのポリシーにクロスアカウントのプリンシパルを追加しても、信頼関係は半分しか確立されない点に注意してください。プリンシパルとリソースが異なる にある場合 AWS アカウント、信頼されたアカウントの IAM 管理者は、プリンシパルエンティティ (ユーザーまたはロール) にリソースへのアクセス許可も付与する必要があります。IAM 管理者は、アイデンティティベースのポリシーをエンティティにアタッチすることで権限を付与します。ただし、リソースベースのポリシーで、同じアカウントのプリンシパルへのアクセス権が付与されている場合は、アイデンティティベースのポリシーを追加する必要はありません。詳細については、『IAM ユーザーガイド』の「IAM ロールとリソースベースのポリシーとの相違点」を参照してください。

Amazon Polly のポリシーアクション

ポリシーアクションに対するサポート	はい

管理者は AWS JSON ポリシーを使用して、誰が何にアクセスできるかを指定できます。つまり、どのプリンシパルがどんなリソースにどんな条件でアクションを実行できるかということです。

JSON ポリシーの`Action`要素には、ポリシー内のアクセスを許可または拒否するために使用できるアクションが記述されます。ポリシーアクションの名前は通常、関連付けられた AWS API オペレーションと同じです。一致する API オペレーションのない権限のみのアクションなど、いくつかの例外があります。また、ポリシーに複数アクションが必要なオペレーションもあります。これらの追加アクションは、 依存アクション と呼ばれます。

このアクションは、関連付けられたオペレーションを実行するための権限を付与するポリシーで使用されます。

Amazon Polly アクションのリストを確認するには、「サービス認証リファレンス」の「Actions defined by Amazon Polly」を参照してください。

Amazon Polly のポリシーアクションは、アクションの前にプレフィックスを使用します。

```
polly
```

単一のステートメントで複数のアクションを指定するには、アクションをカンマで区切ります。

```
"Action": [
      "polly:action1",
      "polly:action2"
         ]
```

Amazon Polly でのアイデンティティベースのポリシーの例は、Amazon Polly のアイデンティティベースポリシーの例でご確認ください。

Amazon Polly のポリシーリソース

ポリシーリソースに対するサポート	はい

管理者は AWS JSON ポリシーを使用して、誰が何にアクセスできるかを指定できます。つまり、どのプリンシパルがどのリソースにどのような条件でアクションを実行できるかということです。

Resource JSON ポリシー要素は、アクションが適用されるオブジェクトを指定します。ステートメントには、Resource または NotResource要素を含める必要があります。ベストプラクティスとして、Amazon リソースネーム (ARN) を使用してリソースを指定します。これは、リソースレベルの権限と呼ばれる特定のリソースタイプをサポートするアクションに対して実行できます。

オペレーションのリスト化など、リソースレベルの権限をサポートしないアクションの場合は、ステートメントがすべてのリソースに適用されることを示すために、ワイルドカード (*) を使用します。

```
"Resource": "*"
```

Amazon Polly リソースのタイプとその ARN のリストを確認するには、「サービス認可リファレンス」の「Amazon Polly で定義されるリソース」を参照してください。各リソースのARNを指定できるアクションについては、Amazon Polly で定義されるアクションを参照してください。

Amazon Polly でのアイデンティティベースのポリシーの例は、Amazon Polly のアイデンティティベースポリシーの例でご確認ください。

Amazon Polly のポリシー条件キー

サービス固有のポリシー条件キーのサポート	いいえ

管理者は AWS JSON ポリシーを使用して、誰が何にアクセスできるかを指定できます。つまり、どのプリンシパルがどんなリソースにどんな条件でアクションを実行できるかということです。

`Condition` 要素 (または `Condition` ブロック) を使用すると、ステートメントが有効な条件を指定できます。`Condition` 要素はオプションです。イコールや未満などの 条件演算子 を使用して条件式を作成することで、ポリシーの条件とリクエスト内の値を一致させることができます。

1 つのステートメントに複数の `Condition` 要素を指定するか、1 つの `Condition` 要素に複数のキーを指定すると、 AWS は `AND` 論理演算子を使用してそれらを評価します。1 つの条件キーに複数の値を指定すると、 は論理ORオペレーションを使用して条件 AWS を評価します。ステートメントの権限が付与される前にすべての条件が満たされる必要があります。

条件を指定する際にプレースホルダー変数も使用できます。例えば IAM ユーザーに、IAM ユーザー名がタグ付けされている場合のみリソースにアクセスできる権限を付与することができます。詳細については、『IAM ユーザーガイド』の「IAM ポリシーの要素: 変数およびタグ」を参照してください。

AWS は、グローバル条件キーとサービス固有の条件キーをサポートします。すべての AWS グローバル条件キーを確認するには、「IAM ユーザーガイド」のAWS 「 グローバル条件コンテキストキー」を参照してください。

Amazon Polly での条件キーの一覧については、「サービス認証リファレンス」の「Amazon Polly の条件キー」を参照してください。どのアクションおよびリソースと条件キーを使用できるかについては、「Amazon Polly で定義されるアクション」を参照してください。

Amazon Polly でのアイデンティティベースのポリシーの例は、Amazon Polly のアイデンティティベースポリシーの例でご確認ください。

Amazon Polly での ACL

ACL のサポート	No

アクセスコントロールリスト (ACL) は、どのプリンシパル (アカウントメンバー、ユーザー、またはロール) がリソースにアクセスするための権限を持つかを制御します。ACL はリソースベースのポリシーに似ていますが、JSON ポリシードキュメント形式は使用しません。

Amazon Polly での ABAC

ABAC (ポリシー内のタグ) のサポート	いいえ

属性ベースのアクセス制御 (ABAC) は、属性に基づいて権限を定義する認可戦略です。では AWS、これらの属性はタグ と呼ばれます。タグは、IAM エンティティ (ユーザーまたはロール) および多くの AWS リソースにアタッチできます。エンティティとリソースのタグ付けは、ABAC の最初の手順です。その後、プリンシパルのタグがアクセスしようとしているリソースのタグと一致した場合に操作を許可するように ABAC ポリシーを設計します。

ABAC は、急成長する環境やポリシー管理が煩雑になる状況で役立ちます。

タグに基づいてアクセスを管理するには、`aws:ResourceTag/key-name`、`aws:RequestTag/key-name`、または `aws:TagKeys` の条件キーを使用して、ポリシーの条件要素でタグ情報を提供します。

サービスがすべてのリソースタイプに対して 3 つの条件キーすべてをサポートする場合、そのサービスの値ははいです。サービスが一部のリソースタイプに対してのみ 3 つの条件キーのすべてをサポートする場合、値は「部分的」になります。

ABAC の詳細については、『IAM ユーザーガイド』の「ABAC とは?」を参照してください。ABAC をセットアップするステップを説明するチュートリアルについては、「IAM ユーザーガイド」の「属性に基づくアクセスコントロール (ABAC) を使用する」を参照してください。

Amazon Polly での一時的な認証情報の使用

一時的な認証情報のサポート	はい

一部の は、一時的な認証情報を使用してサインインすると機能 AWS のサービス しません。一時的な認証情報 AWS のサービス を使用する などの詳細については、IAM ユーザーガイドのAWS のサービス「IAM と連携する 」を参照してください。

ユーザー名とパスワード以外の AWS Management Console 方法で にサインインする場合、一時的な認証情報を使用します。例えば、会社の Single Sign-On (SSO) リンク AWS を使用して にアクセスすると、そのプロセスによって一時的な認証情報が自動的に作成されます。また、ユーザーとしてコンソールにサインインしてからロールを切り替える場合も、一時的な認証情報が自動的に作成されます。ロールの切り替えに関する詳細については、「IAM ユーザーガイド」の「ロールへの切り替え (コンソール)」を参照してください。

一時的な認証情報は、 AWS CLI または AWS API を使用して手動で作成できます。その後、これらの一時的な認証情報を使用して にアクセスします AWS。 AWS 長期的なアクセスキーを使用する代わりに、一時的な認証情報を動的に生成することをお勧めします。詳細については、「IAM の一時的セキュリティ認証情報」を参照してください。

Amazon Polly のクロスサービス転送アクセスセッション (FAS)

転送アクセスセッション (FAS) をサポート	はい

IAM ユーザーまたはロールを使用して でアクションを実行すると AWS、プリンシパルと見なされます。一部のサービスを使用する際に、アクションを実行することで、別のサービスの別のアクションがトリガーされることがあります。FAS は、 を呼び出すプリンシパルのアクセス許可を AWS のサービス、ダウンストリームサービス AWS のサービス へのリクエストのリクエストと組み合わせて使用します。FAS リクエストは、サービスが他の AWS のサービス またはリソースとのやり取りを完了する必要があるリクエストを受け取った場合にのみ行われます。この場合、両方のアクションを実行するためのアクセス許可が必要です。FAS リクエストを行う際のポリシーの詳細については、「転送アクセスセッション」を参照してください。

Amazon Polly のサービスロール

サービスロールのサポート	いいえ

サービスロールとは、サービスがユーザーに代わってアクションを実行するために引き受ける IAM ロールです。IAM 管理者は、IAM 内からサービスロールを作成、変更、削除できます。詳細につい

ては、「IAM ユーザーガイド」の「AWS のサービスに権限を委任するロールの作成」を参照してください。

> **Warning**
>
> サービスロールの許可を変更すると、Amazon Polly の機能が破損する可能性があります。Amazon Polly が指示する場合以外は、サービスロールを編集しないでください。

Amazon Polly のサービスにリンクされたロール

サービスにリンクされたロールのサポート	いいえ

サービスにリンクされたロールは、 にリンクされたサービスロールの一種です AWS のサービス。サービスは、ユーザーに代わってアクションを実行するロールを引き受けることができます。サービスにリンクされたロールは に表示され AWS アカウント 、サービスによって所有されます。IAM 管理者は、サービスにリンクされたロールの権限を表示できますが、編集することはできません。

サービスにリンクされたロールの作成または管理の詳細については、「IAM と提携するAWS のサービス」を参照してください。表の中から、[Service-linked role] (サービスにリンクされたロール) 列に Yes と記載されたサービスを見つけます。サービスにリンクされたロールに関するドキュメントをサービスで表示するには、[Yes] リンクを選択します。

Amazon Polly の IAM ロール

アイデンティティベースのアクセス権限ポリシーを IAM ロールにアタッチして、クロスアカウントアクセス権限を付与できます。例えば、アカウント A の管理者は、次のように別の AWS アカウント (アカウント B など) または AWS サービスにクロスアカウントアクセス許可を付与するロールを作成できます。

1. アカウント A の管理者は、IAM ロールを作成して、アカウント A のリソースに許可を付与するロールに許可ポリシーをアタッチします。
2. アカウント A の管理者は、アカウント B をそのロールを引き受けるプリンシパルとして識別するロールに、信頼ポリシーをアタッチします。
3. アカウント B の管理者は、アカウント B の任意のユーザーにロールを引き受けるアクセス許可を委任できます。これにより、アカウント B のユーザーは、アカウント A のリソースを作成または

アクセスできるようになります。ロールを引き受けるアクセス許可を AWS サービスに付与する場合は、信頼ポリシーのプリンシパルを AWS サービスプリンシパルにすることもできます。

IAM を使用した許可の委任の詳細については、「IAM ユーザーガイド」の「アクセス管理」を参照してください。

レキシコンを配置および取得するアクセス権限と現在使用可能なレキスコンを一覧表示するアクセス権限を付与するポリシーの例を以下に示します。

Amazon Polly は、アクションのアイデンティティベースのポリシーをリソースレベルでサポートしています。場合によっては、リソースが ARN によって制限されることがあります。これは SynthesizeSpeech、StartSpeechSynthesisTask、PutLexicon、GetLexicon、および DeleteLexicon オペレーションに当てはまります。これらの場合、Resource 値は ARN によって示されます。たとえば、Resource 値としての arn:aws:polly:*us-east-2*:*account-id*:lexicon/* は、us-east-2 リージョン内の所有するすべてのレキシコンに対するアクセス許可を指定します。

```
{
   "Version": "2012-10-17",
   "Statement": [{
      "Sid": "AllowPut-Get-ListActions",
      "Effect": "Allow",
      "Action": [
         "polly:PutLexicon",
         "polly:GetLexicon",
         "polly:ListLexicons"],
      "Resource": "arn:aws:polly:us-east-2:account-id:lexicon/*"
      }
   ]
}
```

ただし、すべてのオペレーションで ARN が使用されるわけはありません。これは、DescribeVoices、ListLexicons、GetSpeechSynthesisTasks、および ListSpeechSynthesisTasks オペレーションの場合です。

ユーザー、グループ、ロール、許可の詳細については、「IAM ユーザーガイド」の「アイデンティティ (ユーザー、グループ、ロール)」を参照してください。

Amazon Polly のアイデンティティベースポリシーの例

デフォルトでは、ユーザーおよびロールにはAmazon Polly リソースを作成または変更するアクセス許可はありません。また、、 AWS Command Line Interface （ AWS CLI ）AWS Management Console、または AWS API を使用してタスクを実行することはできません。IAM 管理者は、リソースで必要なアクションを実行するための権限をユーザーに付与する IAM ポリシーを作成できます。その後、管理者はロールに IAM ポリシーを追加し、ユーザーはロールを引き受けることができます。

これらサンプルの JSON ポリシードキュメントを使用して、IAM アイデンティティベースのポリシーを作成する方法については、『IAM ユーザーガイド』の「IAM ポリシーの作成」を参照してください。

Amazon Polly が定義するアクションとリソースタイプ (リソースタイプごとの ARN の形式を含む) の詳細については、サービス認証リファレンスの「Amazon Polly のアクション、リソース、および条件キー」を参照してください。

トピック

- ポリシーのベストプラクティス
- Amazon Polly コンソールの使用
- 自分の権限の表示をユーザーに許可する
- AWS Amazon Polly の マネージド (事前定義) ポリシー
- カスタマーマネージドポリシーの例

ポリシーのベストプラクティス

ID ベースのポリシーは、ユーザーのアカウント内で誰かが Amazon Polly リソースを作成、アクセス、または削除できるかどうかを決定します。これらのアクションを実行すると、 AWS アカウントに料金が発生する可能性があります。アイデンティティベースポリシーを作成したり編集したりする際には、以下のガイドラインと推奨事項に従ってください:

- AWS 管理ポリシーを開始し、最小特権のアクセス許可に移行する – ユーザーとワークロードにアクセス許可を付与するには、多くの一般的なユースケースにアクセス許可を付与する AWS 管理ポリシーを使用します。これらは で使用できます AWS アカウント。ユースケースに固有の AWS カスタマー管理ポリシーを定義して、アクセス許可をさらに減らすことをお勧めします。詳細については、『IAM ユーザーガイド』の「AWS マネージドポリシー」または「AWS ジョブ機能の管理ポリシー」を参照してください。

- 最小特権を適用する – IAM ポリシーで権限を設定するときは、タスクの実行に必要な権限のみを付与します。これを行うには、特定の条件下で特定のリソースに対して実行できるアクションを定義します。これは、最小特権権限とも呼ばれています。IAM を使用して権限を適用する方法の詳細については、『IAM ユーザーガイド』の「IAM でのポリシーと権限」を参照してください。
- IAM ポリシーで条件を使用してアクセスをさらに制限する - ポリシーに条件を追加して、アクションやリソースへのアクセスを制限できます。例えば、ポリシー条件を記述して、すべてのリクエストを SSL を使用して送信するように指定できます。条件を使用して、 などの特定の を介してサービスアクションが使用される場合に AWS のサービス、サービスアクションへのアクセスを許可することもできます AWS CloudFormation。詳細については、『IAM ユーザーガイド』の [IAM JSON policy elements: Condition] (IAM JSON ポリシー要素：条件) を参照してください。
- IAM Access Analyzer を使用して IAM ポリシーを検証し、安全で機能的な権限を確保する - IAM Access Analyzer は、新規および既存のポリシーを検証して、ポリシーが IAM ポリシー言語 (JSON) および IAM のベストプラクティスに準拠するようにします。IAM アクセスアナライザーは 100 を超えるポリシーチェックと実用的な推奨事項を提供し、安全で機能的なポリシーの作成をサポートします。詳細については、『IAM ユーザーガイド』の「IAM Access Analyzer ポリシーの検証」を参照してください。
- 多要素認証 (MFA) を要求する – で IAM ユーザーまたはルートユーザーを必要とするシナリオがある場合は AWS アカウント、セキュリティを強化するために MFA を有効にします。API オペレーションが呼び出されるときに MFA を必須にするには、ポリシーに MFA 条件を追加します。詳細については、『IAM ユーザーガイド』の「MFA 保護 API アクセスの設定」を参照してください。

IAM でのベストプラクティスの詳細については、『IAM ユーザーガイド』の「IAM でのセキュリティのベストプラクティス」を参照してください。

Amazon Polly コンソールの使用

Amazon Polly コンソールにアクセスするには、許可の最小限のセットが必要です。これらのアクセス許可により、 の Amazon Polly リソースの詳細を一覧表示および表示できます AWS アカウント。最小限必要な許可よりも制限が厳しいアイデンティティベースのポリシーを作成すると、そのポリシーを持つエンティティ (ユーザーまたはロール) に対してコンソールが意図したとおりに機能しません。

AWS CLI または AWS API のみを呼び出すユーザーには、最小限のコンソールアクセス許可を付与する必要はありません。代わりに、実行しようとしている API オペレーションに一致するアクションのみへのアクセスが許可されます。

ユーザーとロールが引き続き Amazon Polly コンソールを使用できるようにするには、エンティティに Amazon Polly *ConsoleAccess*または *ReadOnly* AWS 管理ポリシーもアタッチします。詳細については、「IAM ユーザーガイド」の「ユーザーへのアクセス許可の追加」を参照してください。

Amazon Polly コンソールを使用するには、すべての Amazon Polly API にアクセス許可を付与します。必要な追加のアクセス権限はありません。コンソールの完全な機能を得るには、以下のポリシーを使用できます。

```
{
  "Version": "2012-10-17",
     "Statement": [{
        "Sid": "Console-AllowAllPollyActions",
        "Effect": "Allow",
        "Action": [
           "polly:*"],
        "Resource": "*"
      }
   ]
}
```

自分の権限の表示をユーザーに許可する

この例では、ユーザーアイデンティティにアタッチされたインラインおよびマネージドポリシーの表示を IAM ユーザーに許可するポリシーの作成方法を示します。このポリシーには、コンソールで、または AWS CLI または AWS API を使用してプログラムでこのアクションを実行するアクセス許可が含まれています。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
```

```
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

AWS Amazon Polly の マネージド (事前定義) ポリシー

AWS は、 によって作成および管理されるスタンドアロン IAM ポリシーを提供することで、多くの一般的なユースケースに対処します AWS。これらの AWS 管理ポリシーは、一般的なユースケースに必要なアクセス許可を付与するため、どのアクセス許可が必要かを調査する必要がなくなります。詳細については、「IAM ユーザーガイド」の「AWS マネージドポリシー」を参照してください。

アカウントのユーザーにアタッチできる以下の AWS マネージドポリシーは、Amazon Polly に固有です。

- AmazonPollyReadOnlyAccess – リソースへの読み取り専用アクセスを付与し、レキシコンの一覧表示、レキシコンの取得、使用可能な音声の一覧表示、音声の合成 (合成された音声へのレキシコンの適用を含む) を許可します。
- AmazonPollyFullAccess – リソースとサポートされているすべてのオペレーションへのフルアクセスを許可します。

> **Note**
>
> IAM コンソールにサインインし、特定のポリシーを検索することで、これらの許可ポリシーを確認できます。

独自のカスタム IAM ポリシーを作成して、Amazon Polly アクションとリソースのための権限を許可することもできます。これらのカスタムポリシーは、それらのアクセス許可が必要な IAM ユーザーまたはグループにアタッチできます。

カスタマーマネージドポリシーの例

このセクションでは、さまざまな Amazon Polly アクションのアクセス権限を付与するユーザーポリシー例を示しています。これらのポリシーは、 AWS SDKsまたは を使用している場合に機能します AWS CLI。コンソールを使用する場合は、すべての Amazon Polly API APIs。

> **Note**
>
> すべての例で、us-east-2 リージョンを使用し、架空のアカウント ID を含めています。

例

- 例 1: すべての Amazon Polly アクションを許可する
- 例 2: を除くすべての Amazon Polly アクションを許可する DeleteLexicon
- 例 3: 許可 DeleteLexicon
- 例 4: 指定されたリージョンでレキシコンの削除を許可する
- 例 5: 指定されたレキシコン DeleteLexicon を許可する

例 1: すべての Amazon Polly アクションを許可する

サインアップ後 (「Amazon Polly のセットアップ」を参照)、ユーザーの作成やアクセス許可の管理など、アカウントを管理するための管理者ユーザーを作成します。

すべての Amazon Polly アクションに対するアクセス許可のあるユーザーを作成する場合があります。このユーザーは、Amazon Polly を使用するためのサービス固有の管理者と考えてください。このユーザーに以下のアクセス権限をアタッチできます。

```
{
   "Version": "2012-10-17",
   "Statement": [{
      "Sid": "AllowAllPollyActions",
      "Effect": "Allow",
      "Action": [
```

```
         "polly:*"],
      "Resource": "*"
      }
   ]
}
```

例 2: を除くすべての Amazon Polly アクションを許可する DeleteLexicon

以下のアクセス許可ポリシーは、DeleteLexicon を除くすべてのアクションを実行するアクセス許可をユーザーに付与します (削除アクセス許可がすべてのリージョンで明示的に拒否されます)。

```
{
   "Version": "2012-10-17",
   "Statement": [{
      "Sid": "AllowAllActions-DenyDelete",
      "Effect": "Allow",
      "Action": [
         "polly:DescribeVoices",
         "polly:GetLexicon",
         "polly:PutLexicon",
         "polly:SynthesizeSpeech",
         "polly:ListLexicons"],
      "Resource": "*"
      }
      {
      "Sid": "DenyDeleteLexicon",
      "Effect": "Deny",
      "Action": [
         "polly:DeleteLexicon"],
      "Resource": "*"
      }
   ]
}
```

例 3: 許可 DeleteLexicon

以下のアクセス許可ポリシーは、プロジェクトまたはプロジェクトが所在するリージョンにかかわらず所有するレキシコンを削除するアクセス許可をユーザーに付与します。

```
{
  "Version": "2012-10-17",
  "Statement": [{
```

```
      "Sid": "AllowDeleteLexicon",
      "Effect": "Allow",
      "Action": [
         "polly:DeleteLexicon"],
      "Resource": "*"
      }
   ]
}
```

例 4: 指定されたリージョンでレキシコンの削除を許可する

以下のアクセス許可ポリシーは、1 つのリージョン (この場合は us-east-2) 内で所有するすべてのプロジェクトのレキシコンを削除するアクセス許可をユーザーに付与します。

```
{
  "Version": "2012-10-17",
  "Statement": [{
      "Sid": "AllowDeleteSpecifiedRegion",
      "Effect": "Allow",
      "Action": [
         "polly:DeleteLexicon"],
      "Resource": "arn:aws:polly:us-east-2:123456789012:lexicon/*"
      }
   ]
}
```

例 5: 指定されたレキシコン DeleteLexicon を許可する

以下のアクセス許可ポリシーは、特定のリージョン (この場合は us-east-2) 内で所有する特定のレキシコン (この場合は myLexicon) を削除するアクセス許可をユーザーに付与します。

```
{
  "Version": "2012-10-17",
  "Statement": [{
      "Sid": "AllowDeleteForSpecifiedLexicon",
      "Effect": "Allow",
      "Action": [
         "polly:DeleteLexicon"],
      "Resource": "arn:aws:polly:us-east-2:123456789012:lexicon/myLexicon"
      }
   ]
}
```

Amazon Polly API アクセス許可: アクション、アクセス許可、リソースの参照

IAM アイデンティティ (アイデンティティベースのポリシー) にアタッチできるアクセス許可ポリシーを設定する場合、次のリストを参照として使用できます。には、各 Amazon Polly API オペレーション、アクションを実行するためのアクセス許可を付与できる対応するアクション、およびアクセス許可を付与できる AWS リソースが含まれます。ポリシーの `Action` フィールドでアクションを指定し、ポリシーの `Resource` フィールドでリソースの値を指定します。

Amazon Polly ポリシーで AWS全体の条件キーを使用して、条件を表現できます。全体のキーの完全なリストについては、「IAM AWSユーザーガイド」の「利用可能なキー」を参照してください。

> **Note**
>
> アクションを指定するには、API オペレーション名 (`polly:GetLexicon` など) の前に `polly` プレフィックスを使用します。

Amazon Polly は、アクションのアイデンティティベースのポリシーをリソースレベルでサポートしています。このため、`Resource` 値は ARN により示されます。たとえば、`Resource` 値としての `arn:aws:polly:*us-east-2*:*account-id*:lexicon/*` は、`us-east-2` リージョン内の所有するすべてのレキシコンに対するアクセス許可を指定します。

Amazon Polly API は、リソースレベルのアクションのアクセス許可をサポートしていないため、ほとんどのポリシーは `Resource` 値としてワイルドカード文字 (*) を指定します。ただし、特定のリージョンへのアクセス許可を制限する必要がある場合、このワイルドカード文字は適切な ARN に置き換えられます。`arn:aws:polly:*region*:*account-id*:lexicon/*`.

Amazon Polly API およびアクションで必要なアクセス許可

API オペレーション: DeleteLexicon

必要なアクセス許可 (API アクション): `polly:DeleteLexicon`

リソース: `arn:aws:polly:*region*:*account-id*:lexicon/*LexiconName*`

API オペレーション: DescribeVoices

必要なアクセス許可 (API アクション): `polly:DescribeVoices`

リソース: arn:aws:polly:*region*:*account-id*:lexicon/*voice-name*

API オペレーション: GetLexicon

必要なアクセス許可 (API アクション): polly:GetLexicon

リソース: arn:aws:polly:*region*:*account-id*:lexicon/*voice-name*

API オペレーション: ListLexicons

必要なアクセス許可 (API アクション): polly:ListLexicons

リソース: arn:aws:polly:*region*:*account-id*:lexicon/*

API オペレーション: PutLexicon

必要なアクセス許可 (API アクション): polly:ListLexicons

リソース: *

API オペレーション: SynthesizeSpeech

必要なアクセス許可 (API アクション): polly:SynthesizeSpeech

リソース: *

Amazon Polly の ID とアクセスのトラブルシューティング

以下の情報を使用して、Amazon Polly と IAM の使用時に発生する可能性がある一般的な問題の診断と修正に役立てます。

トピック

- Amazon Polly でアクションを実行する権限がない
- iam を実行する権限がありません。PassRole
- 自分の 以外のユーザーに Amazon Polly リソース AWS アカウント へのアクセスを許可したい

Amazon Polly でアクションを実行する権限がない

「I am not authorized to perform an action in Amazon Bedrock」というエラーが表示された場合、そのアクションを実行できるようにポリシーを更新する必要があります。

次のエラー例は、mateojackson IAM ユーザーがコンソールを使用して、ある *my-example-widget* リソースに関する詳細情報を表示しようとしたことを想定して、その際に必要なpolly:*GetWidget* アクセス許可を持っていない場合に発生するものです。

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 polly:GetWidget on resource: my-example-widget
```

この場合、polly:*GetWidget* アクションを使用して *my-example-widget*リソースへのアクセスを許可するように、mateojackson ユーザーのポリシーを更新する必要があります。

サポートが必要な場合は、 AWS 管理者にお問い合わせください。サインイン資格情報を提供した担当者が管理者です。

iam を実行する権限がありません。PassRole

iam:PassRole アクションを実行する権限がないというエラーが表示された場合は、ポリシーを更新して Amazon Polly にロールを渡せるようにする必要があります。

一部の AWS のサービス では、新しいサービスロールまたはサービスにリンクされたロールを作成する代わりに、そのサービスに既存のロールを渡すことができます。そのためには、サービスにロールを渡す権限が必要です。

次の例のエラーは、marymajor という IAM ユーザーがコンソールを使用して Amazon Polly でアクションを実行しようとする場合に発生します。ただし、このアクションをサービスが実行するには、サービスロールから付与された権限が必要です。Mary には、ロールをサービスに渡す権限がありません。

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

この場合、Mary のポリシーを更新してメアリーに iam:PassRole アクションの実行を許可する必要があります。

サポートが必要な場合は、 AWS 管理者にお問い合わせください。サインイン資格情報を提供した担当者が管理者です。

自分の 以外のユーザーに Amazon Polly リソース AWS アカウント へのアクセスを許可したい

他のアカウントのユーザーや組織外の人が、リソースにアクセスするために使用できるロールを作成できます。ロールの引き受けを委託するユーザーを指定できます。リソースベースのポリシーまた

はアクセスコントロールリスト (ACL) をサポートするサービスの場合、それらのポリシーを使用して、リソースへのアクセスを付与できます。

詳細については、以下を参照してください:

- Amazon Polly がこれらの機能をサポートしているかどうかを確認するには、「Amazon Polly で IAM を使用する方法」を参照してください。
- 所有 AWS アカウント している のリソースへのアクセスを提供する方法については、IAM ユーザーガイドの「所有 AWS アカウント している別の の IAM ユーザーへのアクセスを提供する」を参照してください。
- リソースへのアクセスをサードパーティー に提供する方法については AWS アカウント、IAM ユーザーガイドの「サードパーティー AWS アカウント が所有する へのアクセスを提供する」を参照してください。
- ID フェデレーションを介してアクセスを提供する方法については、『IAM ユーザーガイド』の「外部で認証されたユーザー (ID フェデレーション) へのアクセス権限」を参照してください。
- クロスアカウントアクセスでのロールとリソースベースのポリシーの使用の違いの詳細については、『IAM ユーザーガイド』の「IAM ロールとリソースベースのポリシーとの相違点」を参照してください。

Amazon Polly でのログ記録とモニタリング

モニタリングは、Amazon Polly アプリケーションの信頼性、可用性、パフォーマンスを維持する上で重要な部分です。Amazon Polly API コールをモニタリングするには、 を使用できます AWS CloudTrail。ジョブのステータスをモニタリングするには、Amazon CloudWatch Logs を使用します。

- Amazon CloudWatch アラーム – CloudWatch アラームを使用すると、指定した期間にわたって 1 つのメトリクスを監視できます。メトリクスが特定のしきい値を超えると、Amazon Simple Notification Service トピックまたは AWS Auto Scaling ポリシーに通知が送信されます。CloudWatch アラームは、メトリクスが特定の状態にあるときにアクションを呼び出しません。状態が変わり、それが指定した期間だけ維持される必要があります。詳細については、「Amazon Polly CloudWatch との統合」を参照してください。
- CloudTrail logs – Amazon Polly のユーザー、ロール、または AWS サービスによって実行されたアクションの記録 CloudTrail を提供します。によって収集された情報を使用して CloudTrail、Amazon Polly に対して行われたリクエストを判断できます。リクエストの実行元 IP

アドレス、実行者、実行日時、および追加の詳細を判断することもできます。詳細については、「を使用した Amazon Polly API コールのログ記録 AWS CloudTrail」を参照してください。

Amazon Polly のコンプライアンス検証

サードパーティーの監査者は、複数の コンプライアンスプログラムの一環として Amazon Polly のセキュリティと AWS コンプライアンスを評価します。これらのプログラムには、SOC、PCI、FedRAMP、HIPAA などがあります。

特定のコンプライアンスプログラムの対象となる AWS サービスのリストについては、「コンプライアンスプログラムAWS による対象範囲内のサービスコンプライアンスプログラム」を参照してください。一般的な情報については、AWS 「コンプライアンスプログラム」を参照してください。

を使用して、サードパーティーの監査レポートをダウンロードできます AWS Artifact。詳細については、「Downloading Reports in AWS Artifact」を参照してください。

Amazon Polly を使用する際のお客様のコンプライアンス責任は、お客様のデータの機密性、貴社のコンプライアンス目的、適用される法律および規制によって決まります。 では、コンプライアンスに役立つ以下のリソース AWS を提供しています。

- 「セキュリティ＆コンプライアンスクイックリファレンスガイド」 – これらのデプロイガイドには、アーキテクチャ上の考慮事項の説明と、 AWSでセキュリティとコンプライアンスに重点を置いたベースライン環境をデプロイするための手順が記載されています。
- 「HIPAA セキュリティとコンプライアンスの設計」ホワイトペーパー — このホワイトペーパーでは、企業が AWS を使用して HIPAA 準拠のアプリケーションを作成する方法について説明します。
- AWS コンプライアンスリソース – このワークブックとガイドのコレクションは、お客様の業界や地域に適用される場合があります。
- 「 デベロッパーガイド」の「ルールによるリソースの評価」 – この AWS Config サービスは、リソース設定が社内プラクティス、業界ガイドライン、および規制にどの程度準拠しているかを評価します。 AWS Config
- AWS Security Hub – この AWS サービスは、 内のセキュリティ状態を包括的に把握 AWS し、セキュリティ業界標準とベストプラクティスへの準拠を確認するのに役立ちます。

Amazon Polly の耐障害性

AWS グローバルインフラストラクチャは、 AWS リージョンとアベイラビリティーゾーンを中心に構築されています。 AWS リージョンは、低レイテンシー、高スループット、および高度に冗長なネットワークで接続された、物理的に分離された複数のアベイラビリティーゾーンを提供します。アベイラビリティーゾーンでは、ゾーン間で中断することなく自動的にフェイルオーバーするアプリケーションとデータベースを設計および運用することができます。アベイラビリティーゾーンは、従来の単一または複数のデータセンターインフラストラクチャよりも可用性が高く、フォールトトレラントで、スケーラブルです。

AWS リージョンとアベイラビリティーゾーンの詳細については、AWS 「 グローバルインフラストラクチャ」を参照してください。

Amazon Polly のインフラストラクチャセキュリティ

マネージドサービスである Amazon Polly は、ホワイトペーパー「Amazon Web Services: セキュリティプロセスの概要」に記載されている AWS グローバルネットワークセキュリティの手順で保護されています。

が AWS 公開した API コールを使用して、ネットワーク経由で Amazon Polly にアクセスします。クライアントで Transport Layer Security (TLS) 1.0 以降がサポートされている必要があります。TLS 1.2 以降が推奨されています。また、Ephemeral Diffie-Hellman (DHE) や Elliptic Curve Ephemeral Diffie-Hellman (ECDHE) などの Perfect Forward Secrecy (PFS) を使用した暗号スイートもクライアントでサポートされている必要があります。これらのモードは、Java 7 以降など、最近のほとんどのシステムでサポートされています。

また、リクエストには、アクセスキー ID と、IAM プリンシパルに関連付けられているシークレットアクセスキーを使用して署名する必要があります。または、AWS Security Token Service (AWS STS) を使用して、一時的なセキュリティ認証情報を生成し、リクエストに署名することもできます。

Amazon Polly のセキュリティベストプラクティス

お客様の信頼、プライバシー、コンテンツのセキュリティが当社の最優先事項です。また、当社では、不正なアクセスやお客様のコンテンツが公開されることを防ぐように設計された、高度で信頼できる技術的および物理的な制御を行っています。さらに、当社がデータを使用する場合はお客様との契約を確実に遵守します。詳細については、AWS データプライバシーのよくある質問を参照してください。

Amazon Polly はテキスト提出物の内容を保持しません。

コンプライアンス、ペネトレーションテスト、速報、リソースなど、 AWS セキュリティに関する幅広い情報については、 AWS クラウドセキュリティのウェブサイトを参照してください。

インターフェイス VPC エンドポイントと Amazon Polly の使用

Amazon Virtual Private Cloud (Amazon VPC) を使用して AWS リソースをホストする場合、VPC と Amazon Polly の間にプライベート接続を確立できます。この接続を使用すると、公開インターネットを経由することなく、Amazon Polly を使用して音声を合成できます。

Amazon VPC は、定義した仮想ネットワークで AWS リソースを起動するために使用できる AWS サービスです。VPC を使用することで、IP アドレス範囲、サブネット、ルートテーブル、ネットワークゲートウェイなどのネットワーク設定を制御できます。VPC を Amazon Polly に接続するには、Amazon Polly のインターフェイス VPC エンドポイントを定義します。このタイプのエンドポイントにより、VPC を AWS のサービスに接続できるようになります。このエンドポイントは、インターネットゲートウェイ、ネットワークアドレス変換 (NAT) インスタンス、または VPN 接続を必要とせず、信頼性が高くスケーラブルな Amazon Polly への接続を提供します。詳細については、『Amazon VPC ユーザーガイド』https://docs.aws.amazon.com/vpc/latest/userguide/what-is-amazon-vpc.htmlの「Amazon VPC とは」を参照してください。

インターフェイス VPC エンドポイントは、 を利用しています。これは AWS PrivateLink、Elastic Network Interface とプライベート IP アドレスを使用して 間の AWS のサービス プライベート通信を可能にする AWS テクノロジーです。詳細については、「New -for AWS PrivateLinkAWS のサービス」を参照してください。

以下の手順は、Amazon VPC のユーザー向けです。詳細については、『Amazon VPC ユーザーガイド』の「開始方法」を参照してください。

可用性

VPC エンドポイントは、Amazon Polly がサポートされているすべての地域でサポートされています。 AWS リージョンとアベイラビリティーゾーンの詳細については、AWS 「 グローバルインフラストラクチャ」を参照してください。

Amazon Polly 用の VPC エンドポイントの作成

VPC で Amazon Polly の使用を開始するには、Amazon Polly のインターフェイス VPC エンドポイントを作成します。選択するサービスは、[com.amazonaws.***Region***.polly] です。Amazon Polly の設

定を変更する必要はありません。詳細については、『Amazon VPC ユーザーガイド』の「インターフェイスエンドポイントの作成」を参照してください。

VPC と Amazon Polly との間の接続をテストする

エンドポイントの作成が完了したら、接続をテストできます。

VPC と Amazon Polly のエンドポイント間の接続をテストするには

1. VPC にある Amazon EC2 インスタンスに接続します。接続の詳細については、Amazon EC2 ドキュメントの「Linux インスタンスへの接続」または「Windows インスタンスへの接続」を参照してください。
2. インスタンスから、 AWS CLI の `aws polly describe-voices` を使用して、使用可能な Amazon Polly 音声を一覧表示します。

コマンドに対する応答に使用可能な Amazon Polly 音声のリストが含まれていた場合、コマンドは成功しており VPC エンドポイントが機能しています。

Amazon Polly エンドポイントへのアクセスのコントロール

VPC エンドポイントポリシーは、エンドポイントの作成時または変更時にエンドポイントに加える国際機械技術者協会 (IAM) のリソースポリシーです。エンドポイントの作成時にポリシーをアタッチしない場合、サービスへのフルアクセスを許可するデフォルトのポリシーがアタッチされます。エンドポイントポリシーは、IAM ユーザーポリシーやサービス固有のポリシーを上書き、または置き換えません。これは、評価項目から指定されたサービスへのアクセスを制御するための別のポリシーです。

評価項目のポリシーは、JSON形式で記載する必要があります。

詳細については、Amazon VPC ユーザーガイド VPC エンドポイントによるサービスのアクセスコントロールを参照してください。

Amazon Polly のエンドポイントポリシーの例を次に示します。このポリシーでは、VPC を介して Amazon Polly に接続するユーザーは Amazon Polly で音声を説明したり、音声を合成したりできますが、他の Amazon Polly アクションを実行することはできません。

```
{
  "Statement": [
    {
```

```
      "Sid": "SynthesisAndDescribeVoicesOnly",
      "Principal": "*",
      "Action": [
        "polly:DescribeVoices",
        "polly:SynthesizeSpeech"
      ],
      "Effect": "Allow",
      "Resource": "*"
    }
  ]
}
```

Amazon Polly の VPC エンドポイントポリシーを変更するには

1. https://console.aws.amazon.com/vpc で Amazon VPC コンソールを開きます。
2. ナビゲーションペインで、[エンドポイント] を選択します。
3. Amazon Polly のエンドポイントをまだ作成していない場合は、[エンドポイントの作成] を選択します。次に、[com.amazonaws.***Region***.polly] を選択し、[エンドポイントの作成] を選択します。
4. [com.amazonaws.***Region***.polly] エンドポイントを選択し、画面の下部で [ポリシー] タブを選択します。
5. [ポリシーの編集] を選択してポリシーを変更します。

VPC コンテキストキーのサポート

Amazon Polly では、特定の VPC または特定の VPC エンドポイントへのアクセスを制限できる `aws:SourceVpc` および `aws:SourceVpce` コンテキストキーをサポートしています。これらのキーは、ユーザーが VPC エンドポイントを使用している場合にのみ使用できます。詳細については、『IAM ユーザーガイド』の「一部のサービスに使用可能なキー」を参照してください。

を使用した Amazon Polly API コールのログ記録 AWS CloudTrail

Amazon Polly は AWS CloudTrail、Amazon Polly のユーザー、ロール、または サービスによって実行されたアクションを記録する AWS サービスAmazon PollyAmazon Polly のすべての API コールをイベントとして CloudTrail キャプチャします。キャプチャされた呼び出しには、Amazon Polly コンソールからの呼び出しと、Amazon Polly API オペレーションへのコード呼び出しが含まれます。証跡を作成する場合は、Amazon Polly の CloudTrail イベントなど、Amazon S3 バケットへのイベントの継続的な配信を有効にすることができます。 Amazon S3 Amazon Polly 証跡を設定しない場合でも、 CloudTrail コンソールのイベント履歴 で最新のイベントを表示できます。で収集された情報を使用して CloudTrail、Amazon Polly に対するリクエスト、リクエスト元の IP アドレス、リクエスト者、リクエスト日時などの詳細を確認できます。

の設定と有効化の方法など CloudTrail、 の詳細については、AWS CloudTrail 「 ユーザーガイド」を参照してください。

の Amazon Polly 情報 CloudTrail

CloudTrail AWS アカウントを作成すると、 がアカウントで有効になります。Amazon Polly でサポートされているイベントアクティビティが発生すると、そのアクティビティは CloudTrail イベント履歴 の他の AWS サービスイベントとともにイベントに記録されます。 AWS アカウントで最近のイベントを表示、検索、ダウンロードできます。詳細については、「イベント履歴を使用した CloudTrail イベントの表示」を参照してください。

Amazon Polly のイベントなど、 AWS アカウント内のイベントの継続的な記録については、証跡を作成します。証跡により CloudTrail 、 はログファイルを Amazon S3 バケットに配信できます。デフォルトでは、コンソールで証跡を作成すると、証跡はすべての AWS リージョンに適用されます。証跡は、 AWS パーティション内のすべてのリージョンからのイベントをログに記録し、指定した Amazon S3 バケットにログファイルを配信します。さらに、 CloudTrail ログで収集されたイベントデータをより詳細に分析し、それに基づいて行動するように他の AWS サービスを設定できます。詳細については、次を参照してください:

- 証跡の作成のための概要
- CloudTrail サポートされているサービスと統合
- の Amazon SNS 通知の設定 CloudTrail

- 複数のリージョンからの CloudTrail ログファイルの受信と複数のアカウントからのログファイルの受信 CloudTrail

Amazon Polly では、以下のアクションをイベントとして CloudTrail ログファイルに記録できます。

- DeleteLexicon
- DescribeVoices
- GetLexicon
- GetSpeechSynthesisTask
- ListLexicons
- ListSpeechSynthesisTasks
- PutLexicon
- StartSpeechSynthesisTask
- SynthesizeSpeech

各イベントまたはログエントリには、誰がリクエストを生成したかという情報が含まれます。アイデンティティ情報は、以下を判別するのに役立ちます:

- リクエストがルートユーザーまたは AWS Identity and Access Management (IAM) ユーザーの認証情報を使用して行われたかどうか。
- リクエストがロールまたはフェデレーションユーザーのテンポラリなセキュリティ認証情報を使用して行われたかどうか。
- リクエストが別の AWS サービスによって行われたかどうか。

詳細については、CloudTrail userIdentity Element」を参照してください。

例: Amazon Polly ログファイルのエントリ

証跡は、指定した Amazon S3 バケットにイベントをログファイルとして配信できるようにする設定です。 CloudTrail ログファイルには 1 つ以上のログエントリが含まれます。イベントは任意のソースからの単一のリクエストを表し、リクエストされたアクション、アクションの日時、リクエストパラメータなどに関する情報が含まれます。 CloudTrail ログファイルはパブリック API コールの順序付けられたスタックトレースではないため、特定の順序では表示されません。

次の例は、 を示す CloudTrail ログエントリを示していますSynthesizeSpeech。

```
{
"Records": [
        {
"awsRegion": "us-east-2",
            "eventID": "19bd70f7-5e60-4cdc-9825-936c552278ae",
            "eventName": "SynthesizeSpeech",
            "eventSource": "polly.amazonaws.com",
            "eventTime": "2016-11-02T03:49:39Z",
            "eventType": "AwsApiCall",
            "eventVersion": "1.05",
            "recipientAccountId": "123456789012",
            "requestID": "414288c2-a1af-11e6-b17f-d7cfc06cb461",
            "requestParameters": {
"lexiconNames": [
                    "SampleLexicon"
                ],
                "engine": "neural",
                "outputFormat": "mp3",
                "sampleRate": "22050",
                "text": "**********",
                "textType": "text",
                "voiceId": "Kendra"
            },
            "responseElements": null,
            "sourceIPAddress": "1.2.3.4",
            "userAgent": "Amazon CLI/Polly 1.10 API 2016-06-10",
            "userIdentity": {
"accessKeyId": "EXAMPLE_KEY_ID",
                "accountId": "123456789012",
                "arn": "arn:aws:iam::123456789012:user/Alice",
                "principalId": "EX_PRINCIPAL_ID",
                "type": "IAMUser",
                "userName": "Alice"
            }
        }

    ]
}
```

Amazon Polly CloudWatch との統合

Amazon Polly を操作すると、次のメトリクスとディメンションが 1 分 CloudWatch ごとに に送信されます。Amazon Polly のメトリクスを表示するには、以下の手順を使用できます。

を使用して Amazon Polly をモニタリングできます。これにより CloudWatch、Amazon Polly から raw データを収集し、ほぼリアルタイムの読み取り可能なメトリクスに加工できます。これらの統計は 2 週間記録されるため、historical information にアクセスしてウェブアプリケーションまたはサービスの動作をより的確に把握することができます。デフォルトでは、Amazon Polly メトリクスデータは 1 CloudWatch 分間隔で に送信されます。詳細については、「Amazon ユーザーガイド CloudWatch」の「Amazon CloudWatch とは」を参照してください。

CloudWatch メトリクスの取得 (コンソール)

1. https://console.aws.amazon.com/cloudwatch/ で CloudWatch コンソールを開きます。
2. ナビゲーションペインで メトリクスを選択します。
3. CloudWatch カテゴリ別メトリクスペインの Amazon Polly のメトリクスカテゴリでメトリクスカテゴリを選択し、上部ペインで下にスクロールしてメトリクスの完全なリストを表示します。

での CloudWatch メトリクスの取得 AWS CLI

次のコードは、Amazon Polly の使用可能なメトリクスを表示します。

```
aws cloudwatch list-metrics --namespace "AWS/Polly"
```

上記のコマンドは、次のような Amazon Polly メトリクスのリストを返します。MetricName エレメントは、メトリクスの内容を識別します。

```
{
    "Metrics": [
        {
            "Namespace": "AWS/Polly",
            "Dimensions": [
                {
                    "Name": "Operation",
                    "Value": "SynthesizeSpeech"
                }
```

```
            ],
            "MetricName": "ResponseLatency"
        },
        {
            "Namespace": "AWS/Polly",
            "Dimensions": [
                {
                    "Name": "Operation",
                    "Value": "SynthesizeSpeech"
                }
            ],
            "MetricName": "RequestCharacters"
        }
```

詳細については、「Amazon CloudWatch API リファレンスGetMetricStatistics」の「」を参照してください。

Amazon Polly メトリクス

Amazon Polly は、リクエストごとに次のメトリクスを生成します。これらのメトリクスは集計され、1 分間隔で に送信され、利用可能な CloudWatch 場所に送信されます。

メトリクス	説明
RequestCharacters	リクエスト内の文字数。これは請求可能な文字のみで、SSML タグは含まれません。 有効なディメンション: オペレーション 有効な統計: 最小、最大、平均 SampleCount、合計 単位: 個
ResponseLatency	リクエストが行われてからストリーミングレスポンスの開始までのレイテンシー。 有効なディメンション: オペレーション 有効な統計: 最小、最大、平均、 SampleCount 単位: ミリ秒

メトリクス	説明
2XXCount	正常なレスポンスに対して返された HTTP 200 レベルのコード。 有効なディメンション: Operation 有効な統計: Average、 SampleCount、Sum 単位: 個
4XXCount	エラーに対して返された HTTP 400 レベルのエラーコード。正常なレスポンスが行われるたびに、ゼロ (0) が表示されます。 有効なディメンション: Operation 有効な統計: Average、 SampleCount、Sum 単位: 個
5XXCount	エラーに対して返された HTTP 500 レベルのエラーコード。正常なレスポンスが行われるたびに、ゼロ (0) が表示されます。 有効なディメンション: Operation 有効な統計: Average、 SampleCount、Sum 単位: 個

Amazon Polly メトリクスのディメンション

Amazon Polly メトリクスは AWS/Polly 名前空間を使用し、次のディメンションのメトリクスを提供します。

ディメンション	説明
Operation	メトリクスは、それが参照する API メソッドによってグループ化されます。可能な値は SynthesizeSpeech 、PutLexicon 、DescribeVoices などです。

Amazon Polly API リファレンス

このセクションには、Amazon Polly API リファレンスが含まれています。

> **Note**
>
> 認証済み API コールは、署名バージョン 4 の署名プロセスを使用して署名される必要があります。詳細については、「」のAWS 「 API リクエストの署名」を参照してくださいAmazon Web Services 全般のリファレンス。

トピック

アクション

以下のアクションがサポートされています:

DeleteLexicon

AWS リージョンに保存されている、指定された発音レキシコンを削除します。削除されたレキシコンは、音声合成には使用できません。また、GetLexicon または ListLexicon API を使用して取得することもできません。

詳細については、レキシコンの管理を参照してください。

リクエストの構文

```
DELETE /v1/lexicons/LexiconName HTTP/1.1
```

URI リクエストパラメータ

リクエストでは、次の URI パラメータを使用します。

LexiconName

削除するレキシコンの名前。リージョン内の既存のレキシコンである必要があります。

Pattern: `[0-9A-Za-z]{1,20}`

必須: はい

リクエストボディ

リクエストにリクエスト本文がありません。

レスポンスの構文

```
HTTP/1.1 200
```

レスポンス要素

アクションが成功した場合、サービスは空の HTTP 本文を持つ HTTP 200 レスポンスを返します。

エラー

LexiconNotFoundException

Amazon Polly は、指定されたレキシコンを見つけることができません。これは、レキシコンが見つからない、その名前のスペルが間違っている、または別のリージョンにあるレキシコンが指定されていることが原因である可能性があります。

レキシコンが存在すること、リージョンにあること (ListLexicons を参照)、名前のスペルが正しいことを確認します。その後、もう一度試してください。

HTTP ステータスコード: 404

ServiceFailureException

不明な状態が原因で、サービス障害が発生しました。

HTTP ステータスコード：500

その他の参照資料

言語固有の AWS SDK のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS Go バージョン 2 用 SDK
- AWS Java V2 用 SDK
- AWS V3 用 JavaScript SDK
- AWS PHP V3 用SDK
- AWS Python 用 SDK
- AWS ルビー V3 用 SDK

DescribeVoices

音声合成をリクエストするときに使用可能な音声のリストを返します。各音声は特定の言語で発声され、男性または女性のいずれかであり、音声名の ASCII バージョンである ID によって識別されます。

音声の合成 (SynthesizeSpeech) では、DescribeVoices で返された音声のリストから、必要な音声の音声 ID を指定します。

例えば、ニュースリーダーアプリケーションで、特定の言語でニュースを読ませたいが、ユーザーが音声を選択できるようにしたいとします。DescribeVoices オペレーションを使用すると、選択可能な音声のリストをユーザーに提供できます。

必要に応じて、使用可能な音声をフィルタリングする言語コードを指定できます。例えば、en-US を指定すると、使用可能なすべてのアメリカ英語の音声のリストが返されます。

このオペレーションには polly:DescribeVoices アクションを実行するアクセス許可が必要です。

リクエストの構文

```
GET /v1/voices?
Engine=Engine&IncludeAdditionalLanguageCodes=IncludeAdditionalLanguageCodes&LanguageCode=Langua
 HTTP/1.1
```

URI リクエストパラメータ

リクエストでは、次の URI パラメータを使用します。

Engine

音声合成の入力テキストを処理するときに Amazon standardPolly で使用されるエンジン (neural、、long-formまたは generative) を指定します。 Amazon Polly

有効な値：standard | neural | long-form | generative

IncludeAdditionalLanguageCodes

指定した言語を追加言語として使用するバイリンガルの音声を返すかどうかを示すブール値。例えば、米国英語 (es-US) を使用するすべての言語をリクエストした場合、イタリア語 (iT-IT) と米

国英語の両方を話すイタリア語の音声があれば、yes を指定した場合はその音声が含まれ、no を指定した場合は含まれないことになります。

LanguageCode

返される音声のリストをフィルタリングするための言語識別タグ (言語名の ISO 639 コード-ISO 3166 国コード) です。このオプションパラメータを指定しないと、使用可能な音声がすべて返されます。

有効な値 : arb | cmn-CN | cy-GB | da-DK | de-DE | en-AU | en-GB | en-GB-WLS | en-IN | en-US | es-ES | es-MX | es-US | fr-CA | fr-FR | is-IS | it-IT | ja-JP | hi-IN | ko-KR | nb-NO | nl-NL | pl-PL | pt-BR | pt-PT | ro-RO | ru-RU | sv-SE | tr-TR | en-NZ | en-ZA | ca-ES | de-AT | yue-CN | ar-AE | fi-FI | en-IE | nl-BE | fr-BE

NextToken

前の DescribeVoices オペレーションから返された不透明なページ分割トークン。存在する場合、これはリストを継続する場所を示します。

長さの制限: 最小長は 0 です。最大長は 4,096 です。

リクエスト本文

リクエストにリクエスト本文がありません。

レスポンスの構文

```
HTTP/1.1 200
Content-type: application/json

{
   "NextToken": "string",
   "Voices": [
      {
         "AdditionalLanguageCodes": [ "string" ],
         "Gender": "string",
         "Id": "string",
         "LanguageCode": "string",
         "LanguageName": "string",
         "Name": "string",
         "SupportedEngines": [ "string" ]
```

```
      }
   ]
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

NextToken

音声のリストを継続するために次のリクエストで使用するページ分割トークン。NextToken は、レスポンスが切り詰められた場合にのみ返されます。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 4,096 です。

Voices

音声とそのプロパティのリスト。

型: Voice オブジェクトの配列

エラー

InvalidNextTokenException

NextToken が無効です。スペルが正しいことを確認してから、もう一度試してください。

HTTP ステータスコード：400

ServiceFailureException

不明な状態が原因で、サービス障害が発生しました。

HTTP ステータスコード：500

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

GetLexicon

AWS リージョンに保存されている、指定された発音レキシコンのコンテンツを返します。詳細については、レキシコンの管理を参照してください。

リクエストの構文

```
GET /v1/lexicons/LexiconName HTTP/1.1
```

URI リクエストパラメータ

リクエストでは、次の URI パラメータを使用します。

LexiconName

レキシコンの名前。

Pattern: [0-9A-Za-z]{1,20}

必須: はい

リクエストボディ

リクエストにリクエスト本文がありません。

レスポンスの構文

```
HTTP/1.1 200
Content-type: application/json

{
   "Lexicon": {
      "Content": "string",
      "Name": "string"
   },
   "LexiconAttributes": {
      "Alphabet": "string",
      "LanguageCode": "string",
      "LastModified": number,
      "LexemesCount": number,
```

```
        "LexiconArn": "string",
        "Size": number
    }
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

Lexicon

レキシコンの名前と文字列の内容を提供するレキシコンオブジェクト。

タイプ：Lexicon オブジェクト

LexiconAttributes

レキシコンのメタデータ。使用される表音アルファベット、言語コード、レキシコン ARN、レキシコンで定義された語彙素の数、バイト単位のレキシコンのサイズなどが含まれる。

型: LexiconAttributes オブジェクト

エラー

LexiconNotFoundException

Amazon Polly は、指定されたレキシコンを見つけることができません。これは、レキシコンが見つからない、その名前のスペルが間違っている、または別のリージョンにあるレキシコンが指定されていることが原因である可能性があります。

レキシコンが存在すること、リージョンにあること (ListLexicons を参照)、名前のスペルが正しいことを確認します。その後、もう一度試してください。

HTTP ステータスコード: 404

ServiceFailureException

不明な状態が原因で、サービス障害が発生しました。

HTTP ステータスコード：500

その他の参照資料

言語固有の AWS SDK のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS Go バージョン 2 用 SDK
- AWS Java V2 用 SDK
- AWS V3 用 JavaScript SDK
- AWS PHP V3 用SDK
- AWS Python 用 SDK
- AWS ルビー V3 用 SDK

GetSpeechSynthesisTask

TaskId SpeechSynthesisTask に基づいて特定のオブジェクトを取得します。このオブジェクトには、タスクのステータスや、タスクの出力を含む S3 バケットへのリンクなど、指定された音声合成タスクに関する情報が含まれます。

リクエストの構文

```
GET /v1/synthesisTasks/TaskId HTTP/1.1
```

URI リクエストパラメータ

リクエストでは、次の URI パラメータを使用します。

TaskId

Amazon Polly で生成された音声合成タスクの識別子。

Pattern: `^[a-zA-Z0-9_-]{1,100}$`

必須: はい

リクエストボディ

リクエストにリクエスト本文がありません。

レスポンスの構文

```
HTTP/1.1 200
Content-type: application/json

{
   "SynthesisTask": {
      "CreationTime": number,
      "Engine": "string",
      "LanguageCode": "string",
      "LexiconNames": [ "string" ],
      "OutputFormat": "string",
      "OutputUri": "string",
      "RequestCharacters": number,
```

```
      "SampleRate": "string",
      "SnsTopicArn": "string",
      "SpeechMarkTypes": [ "string" ],
      "TaskId": "string",
      "TaskStatus": "string",
      "TaskStatusReason": "string",
      "TextType": "string",
      "VoiceId": "string"
   }
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

SynthesisTask

SynthesisTask 要求されたタスクから、出力形式、作成時間、タスクステータスなどの情報を提供するオブジェクト。

型: SynthesisTask オブジェクト

エラー

InvalidTaskIdException

入力されたタスク ID が無効です。有効なタスク ID を入力して、もう一度お試しください。

HTTP ステータスコード：400

ServiceFailureException

不明な状態が原因で、サービス障害が発生しました。

HTTP ステータスコード：500

SynthesisTaskNotFoundException

リクエストされたタスク ID を持つ音声合成タスクが見つかりません。

HTTP ステータスコード：400

その他の参照資料

この API を言語固有の AWS SDK で使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS Go バージョン 2 用 SDK
- AWS Java V2 用 SDK
- AWS V3 用 JavaScript SDK
- AWS PHP V3 用SDK
- AWS Python 用 SDK
- AWS ルビー V3 用 SDK

ListLexicons

AWS リージョンに保存されている発音レキシコンのリストを返します。詳細については、レキシコンの管理を参照してください。

リクエストの構文

```
GET /v1/lexicons?NextToken=NextToken HTTP/1.1
```

URI リクエストパラメータ

リクエストでは、次の URI パラメータを使用します。

NextToken

前の ListLexicons オペレーションから返された不透明なページ分割トークン。存在する場合、レキシコンのリストを継続する場所を示します。

長さの制限: 最小長は 0 です。最大長は 4,096 です。

リクエスト本文

リクエストにリクエスト本文がありません。

レスポンスの構文

```
HTTP/1.1 200
Content-type: application/json

{
   "Lexicons": [
      {
         "Attributes": {
            "Alphabet": "string",
            "LanguageCode": "string",
            "LastModified": number,
            "LexemesCount": number,
            "LexiconArn": "string",
            "Size": number
         },
         "Name": "string"
```

```
      }
   ],
   "NextToken": "string"
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

Lexicons

レキシコンの名前と属性のリスト。

型: LexiconDescription オブジェクトの配列

NextToken

レキシコンのリストを継続するために、次のリクエストで使用するページ分割トークン。NextToken は、レスポンスが切り詰められた場合にのみ返されます。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 4,096 です。

エラー

InvalidNextTokenException

NextToken は無効です。スペルが正しいことを確認してから、もう一度試してください。

HTTP ステータスコード：400

ServiceFailureException

不明な状態が原因で、サービス障害が発生しました。

HTTP ステータスコード：500

その他の参照資料

この API を言語固有の AWS SDK で使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS Go バージョン 2 用 SDK
- AWS Java V2 用 SDK
- AWS V3 用 JavaScript SDK
- AWS PHP V3 用SDK
- AWS Python 用 SDK
- AWS ルビー V3 用 SDK

ListSpeechSynthesisTasks

SpeechSynthesisTask 作成日順に並べられたオブジェクトのリストを返します。このオペレーションでは、タスクをそのステータスでフィルタリングできます。例えば、ユーザーは完了したタスクのみを一覧表示できます。

リクエストの構文

```
GET /v1/synthesisTasks?MaxResults=MaxResults&NextToken=NextToken&Status=Status HTTP/1.1
```

URI リクエストパラメータ

リクエストでは、次の URI パラメータを使用します。

MaxResults

List オペレーションで返される音声合成タスクの最大数。

有効範囲: 最小値 は 1 です。最大値は 100 です。

NextToken

音声合成タスクのリストを継続するために、次のリクエストで使用するページ分割トークン。

長さの制限: 最小長は 0 です。最大長は 4,096 です。

Status

List オペレーションで返される音声合成タスクのステータス

有効な値 : `scheduled | inProgress | completed | failed`

リクエスト本文

リクエストにリクエスト本文がありません。

レスポンスの構文

```
HTTP/1.1 200
Content-type: application/json
```

```
{
   "NextToken": "string",
   "SynthesisTasks": [
      {
         "CreationTime": number,
         "Engine": "string",
         "LanguageCode": "string",
         "LexiconNames": [ "string" ],
         "OutputFormat": "string",
         "OutputUri": "string",
         "RequestCharacters": number,
         "SampleRate": "string",
         "SnsTopicArn": "string",
         "SpeechMarkTypes": [ "string" ],
         "TaskId": "string",
         "TaskStatus": "string",
         "TaskStatusReason": "string",
         "TextType": "string",
         "VoiceId": "string"
      }
   ]
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

NextToken

このリクエストの前の List オペレーションから返された不透明なページ分割トークン。存在する場合、これはリストを継続する場所を示します。

型: 文字列

長さの制限: 最小長は 0 です。最大長は 4,096 です。

SynthesisTasks

リストリクエストで指定されたタスクからの情報 (出力形式、作成日時、タスクステータスなど) SynthesisTask を提供するオブジェクトのリスト。

型: SynthesisTask オブジェクトの配列

エラー

InvalidNextTokenException

NextToken は無効です。スペルが正しいことを確認してから、もう一度試してください。

HTTP ステータスコード：400

ServiceFailureException

不明な状態が原因で、サービス障害が発生しました。

HTTP ステータスコード：500

その他の参照資料

言語固有の AWS SDK の 1 つでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS Go バージョン 2 用 SDK
- AWS Java V2 用 SDK
- AWS V3 用 JavaScript SDK
- AWS PHP V3 用SDK
- AWS Python 用 SDK
- AWS ルビー V3 用 SDK

PutLexicon

発音レキシコンを AWS リージョンに保存します。同じ名前のレキシコンがそのリージョンに既に存在する場合は、新しいレキシコンで上書きされます。レキシコン操作には最終的に一貫性があるため、レキシコンが操作に使用できるようになるまでにはしばらく時間がかかる場合があります。SynthesizeSpeech

詳細については、レキシコンの管理を参照してください。

リクエストの構文

```
PUT /v1/lexicons/LexiconName HTTP/1.1
Content-type: application/json

{
   "Content": "string"
}
```

URI リクエストパラメータ

リクエストでは、次の URI パラメータを使用します。

LexiconName

レキシコンの名前。名前は、正規表現形式 [0-9A-Za-z]{1,20} に従う必要があります。つまり、名前は、大文字と小文字が区別される 20 文字までの英数字文字列です。

Pattern: `[0-9A-Za-z]{1,20}`

必須: はい

リクエストボディ

リクエストは以下の JSON 形式のデータを受け入れます。

Content

文字列データとしての PLS レキシコンの内容。

型: 文字列

必須: はい

レスポンスの構文

```
HTTP/1.1 200
```

レスポンス要素

アクションが成功した場合、サービスは空の HTTP 本文を持つ HTTP 200 レスポンスを返します。

エラー

InvalidLexiconException

Amazon Polly は、指定されたレキシコンを見つけることができません。レキシコンの名前のスペルが正しいことを確認してから、もう一度試してください。

HTTP ステータスコード：400

LexiconSizeExceededException

このオペレーションでは、指定されたレキシコンの最大サイズを超過します。

HTTP ステータスコード：400

MaxLexemeLengthExceededException

このオペレーションでは、レキシコンの最大サイズを超過します。

HTTP ステータスコード：400

MaxLexiconsNumberExceededException

このオペレーションでは、レキシコンの最大数を超過します。

HTTP ステータスコード：400

ServiceFailureException

不明な状態が原因で、サービス障害が発生しました。

HTTP ステータスコード：500

UnsupportedPlsAlphabetException

レキシコンで指定されたアルファベットは、サポートされているアルファベットではありません。有効な値は、`x-sampa` および `ipa` です。

HTTP ステータスコード：400

UnsupportedPlsLanguageException

レキシコンで指定された言語はサポートされていません。サポートされている言語のリストについては、レキシコン属性を参照してください。。

HTTP ステータスコード：400

その他の参照資料

この API を言語固有の AWS SDK で使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS Go バージョン 2 用 SDK
- AWS Java V2 用 SDK
- AWS V3 用 JavaScript SDK
- AWS PHP V3 用SDK
- AWS Python 用 SDK
- AWS ルビー V3 用 SDK

StartSpeechSynthesisTask

新しい SpeechSynthesisTask を開始することにより、非同期合成タスクの作成を可能にします。このオペレーションには、音声合成に必要なすべての標準情報と、合成タスクの出力を保存するサービスのための Amazon S3 バケットの名前と 2 つのオプションのパラメータ (OutputS3KeyPrefix とSnsTopicArn) が必要です。合成タスクが作成されると、このオペレーションは SpeechSynthesisTask オブジェクトを返します。このオブジェクトには、このタスクの ID と現在のステータスが含まれます。SpeechSynthesisTask オブジェクトは、非同期合成タスクの開始後 72 時間使用できます。

リクエストの構文

```
POST /v1/synthesisTasks HTTP/1.1
Content-type: application/json

{
   "Engine": "string",
   "LanguageCode": "string",
   "LexiconNames": [ "string" ],
   "OutputFormat": "string",
   "OutputS3BucketName": "string",
   "OutputS3KeyPrefix": "string",
   "SampleRate": "string",
   "SnsTopicArn": "string",
   "SpeechMarkTypes": [ "string" ],
   "Text": "string",
   "TextType": "string",
   "VoiceId": "string"
}
```

URI リクエストパラメータ

リクエストでは URI パラメータを使用しません。

リクエストボディ

リクエストは以下の JSON 形式のデータを受け入れます。

Engine

音声合成の入力テキストを処理するときに使用する Amazon standardPolly のエンジン (neural、、long-formまたは generative) を指定します。 Amazon Polly 選択したエンジンでサポートされていない音声を使用すると、エラーが発生します。

型: 文字列

有効な値 : standard | neural | long-form | generative

必須 : いいえ

LanguageCode

音声合成リクエストのオプション言語コード。これは、インド英語 (en-IN) とヒンディー語 (hi-IN) のどちらにも使用できる Aditi などのバイリンガル音声を使用する場合にのみ必要です。

バイリンガル音声の使用時に言語コードが指定されていない場合、Amazon Polly はバイリンガル音声のデフォルト言語を使用します。任意の音声のデフォルト言語は、 LanguageCodeパラメータの DescribeVoicesオペレーションによって返される言語です。例えば、言語コードが指定されていない場合、Aditi はヒンディー語ではなくインド英語を使用します。

型: 文字列

有効な値 : arb | cmn-CN | cy-GB | da-DK | de-DE | en-AU | en-GB | en-GB-WLS | en-IN | en-US | es-ES | es-MX | es-US | fr-CA | fr-FR | is-IS | it-IT | ja-JP | hi-IN | ko-KR | nb-NO | nl-NL | pl-PL | pt-BR | pt-PT | ro-RO | ru-RU | sv-SE | tr-TR | en-NZ | en-ZA | ca-ES | de-AT | yue-CN | ar-AE | fi-FI | en-IE | nl-BE | fr-BE

必須 : いいえ

LexiconNames

合成時にサービスが適用する 1 つ以上の発音レキシコン名のリスト。レキシコンは、レキシコンの言語が音声の言語と同じ場合にのみ適用されます。

タイプ : 文字列の配列

配列メンバー: 5 つの項目の最大数。

Pattern: [0-9A-Za-z]{1,20}

必須: いいえ

OutputFormat

返された出力がエンコードされる形式。オーディオストリームの場合、これは mp3、ogg_vorbis、または pcm になります。スピーチマークの場合、これは json になります。

型: 文字列

有効な値：`json | mp3 | ogg_vorbis | pcm`

必須: はい

OutputS3BucketName

出力ファイルの保存先となる Amazon S3 バケット名。

型: 文字列

Pattern: `^[a-z0-9][\.\-a-z0-9]{1,61}[a-z0-9]$`

必須：はい

OutputS3KeyPrefix

出力音声ファイルの Amazon S3 キープレフィックス。

型: 文字列

パターン：`^[0-9a-zA-Z\/\!\-_\.\*\'\(\):;\$@=+\,\?&]{0,800}$`

必須: いいえ

SampleRate

Hz で指定した音声周波数。

mp3 および ogg_vorbis の有効な値は「8000」、「16000」、「22050」、および「24000」です。標準音声のデフォルト値は「22050」です。ニューラル音声のデフォルト値は「24000」です。ロングフォーム音声のデフォルト値は「24000」です。生成音声のデフォルト値は「24000」です。

pcm の有効な値は「8000」と「16000」です。デフォルト値は「16000」です。

タイプ: 文字列

必須: いいえ

SnsTopicArn

音声合成タスクのステータス通知を提供するためにオプションで使用される SNS トピックの ARN。

型: 文字列

パターン：`^arn:aws(-(cn|iso(-b)?|us-gov))?:sns:[a-z0-9_-]{1,50}:\d{12}:[a-zA-Z0-9_-]{1,251}([a-zA-Z0-9_-]{0,5}|\.fifo)$`

必須: いいえ

SpeechMarkTypes

入力テキストに対して返されるスピーチマークのタイプ。

タイプ：文字列の配列

配列メンバー: 最大数は 4 項目です。

有効な値：`sentence | ssml | viseme | word`

必須：いいえ

Text

合成する入力テキスト。として ssml を指定する場合は TextType、入力テキストの SSML 形式に従います。

型: 文字列

必須: はい

TextType

入力テキストがプレーンテキストか SSML かを指定します。デフォルト値はプレーンテキストです。

型: 文字列

有効な値：`ssml | text`

必須：いいえ

VoiceId

合成に使用する音声 ID。

型: 文字列

有効な値 : Aditi | Amy | Astrid | Bianca | Brian | Camila | Carla | Carmen | Celine | Chantal | Conchita | Cristiano | Dora | Emma | Enrique | Ewa | Filiz | Gabrielle | Geraint | Giorgio | Gwyneth | Hans | Ines | Ivy | Jacek | Jan | Joanna | Joey | Justin | Karl | Kendra | Kevin | Kimberly | Lea | Liv | Lotte | Lucia | Lupe | Mads | Maja | Marlene | Mathieu | Matthew | Maxim | Mia | Miguel | Mizuki | Naja | Nicole | Olivia | Penelope | Raveena | Ricardo | Ruben | Russell | Salli | Seoyeon | Takumi | Tatyana | Vicki | Vitoria | Zeina | Zhiyu | Aria | Ayanda | Arlet | Hannah | Arthur | Daniel | Liam | Pedro | Kajal | Hiujin | Laura | Elin | Ida | Suvi | Ola | Hala | Andres | Sergio | Remi | Adriano | Thiago | Ruth | Stephen | Kazuha | Tomoko | Niamh | Sofie | Lisa | Isabelle | Zayd | Danielle | Gregory | Burcu

必須: はい

レスポンスの構文

```
HTTP/1.1 200
Content-type: application/json

{
   "SynthesisTask": {
      "CreationTime": number,
      "Engine": "string",
      "LanguageCode": "string",
      "LexiconNames": [ "string" ],
      "OutputFormat": "string",
      "OutputUri": "string",
      "RequestCharacters": number,
      "SampleRate": "string",
      "SnsTopicArn": "string",
      "SpeechMarkTypes": [ "string" ],
      "TaskId": "string",
      "TaskStatus": "string",
      "TaskStatusReason": "string",
```

```
      "TextType": "string",
      "VoiceId": "string"
   }
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

サービスから以下のデータが JSON 形式で返されます。

SynthesisTask

SynthesisTask 新しく送信された音声合成タスクに関する情報と属性を提供する オブジェクト。

型: SynthesisTask オブジェクト

エラー

EngineNotSupportedException

このエンジンは、指定された音声に対応していません。エンジンが対応する新しい音声を選択するか、エンジンを変更してオペレーションを再開してください。

HTTP ステータスコード：400

InvalidS3BucketException

提供された Amazon S3 バケット名が無効です。S3 バケットの命名要件で入力を確認し、もう一度試してください。

HTTP ステータスコード：400

InvalidS3KeyException

提供された Amazon S3 のキープレフィックスが無効です。有効な S3 オブジェクトキー名を指定してください。

HTTP ステータスコード：400

InvalidSampleRateException

指定されたサンプルレートが無効です。

HTTP ステータスコード：400

InvalidSnsTopicArnException

提供された SNS トピック ARN が無効です。有効な SNS トピック ARN を指定して、もう一度試してください。

HTTP ステータスコード：400

InvalidSsmlException

提供された SSML が無効です。SSML 構文、タグと値のスペルを確認し、もう一度試してください。

HTTP ステータスコード：400

LanguageNotSupportedException

指定された言語は、現在 Amazon Polly がこの容量でサポートしていません。

HTTP ステータスコード：400

LexiconNotFoundException

Amazon Polly は、指定されたレキシコンを見つけることができません。これは、レキシコンが見つからない、その名前のスペルが間違っている、または別のリージョンにあるレキシコンが指定されていることが原因である可能性があります。

レキシコンが存在すること、リージョンにあること (ListLexicons を参照)、名前のスペルが正しいことを確認します。その後、もう一度試してください。

HTTP ステータスコード: 404

MarksNotSupportedForFormatException

選択された `OutputFormat` では、スピーチマークはサポートされていません。スピーチマークは、`json` 形式のコンテンツに対してのみ使用できます。

HTTP ステータスコード：400

ServiceFailureException

不明な状態が原因で、サービス障害が発生しました。

HTTP ステータスコード：500

SsmlMarksNotSupportedForTextTypeException

SSML スピーチマークは、プレーンテキストタイプの入力ではサポートされていません。

HTTP ステータスコード：400

TextLengthExceededException

「テキスト」パラメータの値が、許容される上限より長くなっています。SynthesizeSpeech API の場合、入力テキストの制限は合計で最大 6,000 文字で、そのうち請求対象文字は 3,000 文字までに制限されています。StartSpeechSynthesisTask API の場合、最大 200,000 文字で、そのうち請求対象文字は 100,000 文字までに制限されています。SSML タグは、課金対象文字としてカウントされません。

HTTP ステータスコード：400

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

SynthesizeSpeech

UTF-8 入力 (プレーンテキストまたは SSML) をバイトストリームに合成します。SSML 入力は、有効で正しい形式の SSML でなければなりません。音素マッピングを使用しない場合、一部のアルファベットはすべての音声で使用できない可能性があります (例えば、キリル文字は英語の音声ではまったく読み取れない場合があります)。詳細については、仕組みを参照してください。

リクエストの構文

```
POST /v1/speech HTTP/1.1
Content-type: application/json

{
   "Engine": "string",
   "LanguageCode": "string",
   "LexiconNames": [ "string" ],
   "OutputFormat": "string",
   "SampleRate": "string",
   "SpeechMarkTypes": [ "string" ],
   "Text": "string",
   "TextType": "string",
   "VoiceId": "string"
}
```

URI リクエストパラメータ

リクエストでは URI パラメータを使用しません。

リクエストボディ

リクエストは以下の JSON 形式のデータを受け入れます。

Engine

音声合成の入力テキストを処理するときに使用する Amazonstandard neuralPolly のエンジン (long-form、、、または generative) を指定します。 Amazon Polly 選択した音声でサポートされているエンジンを指定します。エンジンを指定しない場合、標準エンジンはデフォルトで選択されます。選択した音声が標準エンジンでサポートされていない場合、エラーが発生します。Amazon Polly の音声と、どの音声が各エンジンで使用可能かについては、「使用可能な音声」を参照して下さい。

タイプ: 文字列

有効な値: `standard | neural | long-form | generative`

必須: はい

型: 文字列

有効な値 : `standard | neural | long-form | generative`

必須 : いいえ

LanguageCode

音声合成リクエストのオプション言語コード。これは、インド英語 (en-IN) とヒンディー語 (hi-IN) のどちらにも使用できる Aditi などのバイリンガル音声を使用する場合にのみ必要です。

バイリンガル音声の使用時に言語コードが指定されていない場合、Amazon Polly はバイリンガル音声のデフォルト言語を使用します。任意の音声のデフォルト言語は、 LanguageCodeパラメータの DescribeVoicesオペレーションによって返される言語です。例えば、言語コードが指定されていない場合、Aditi はヒンディー語ではなくインド英語を使用します。

型: 文字列

有効な値 : `arb | cmn-CN | cy-GB | da-DK | de-DE | en-AU | en-GB | en-GB-WLS | en-IN | en-US | es-ES | es-MX | es-US | fr-CA | fr-FR | is-IS | it-IT | ja-JP | hi-IN | ko-KR | nb-NO | nl-NL | pl-PL | pt-BR | pt-PT | ro-RO | ru-RU | sv-SE | tr-TR | en-NZ | en-ZA | ca-ES | de-AT | yue-CN | ar-AE | fi-FI | en-IE | nl-BE | fr-BE`

必須 : いいえ

LexiconNames

合成時にサービスが適用する 1 つ以上の発音レキシコン名のリスト。レキシコンは、レキシコンの言語が音声の言語と同じ場合にのみ適用されます。レキシコンの保存については、「」を参照してくださいPutLexicon。

タイプ : 文字列の配列

配列メンバー: 5 つの項目の最大数。

Pattern: `[0-9A-Za-z]{1,20}`

必須: いいえ

OutputFormat

返された出力がエンコードされる形式。オーディオストリームの場合、これは mp3、ogg_vorbis、または pcm になります。スピーチマークの場合、これは json になります。

pcm を使用する場合に返されるコンテンツは、符号付きの 16 ビット、1 つのチャネル (モノ)、リトルエンディアン形式の audio/pcm です。

型: 文字列

有効な値 : `json | mp3 | ogg_vorbis | pcm`

必須: はい

SampleRate

Hz で指定したオーディオ周波数。

mp3 および ogg_vorbis の有効な値は「8000」、「16000」、「22050」、および「24000」です。標準音声のデフォルト値は「22050」です。ニューラル音声のデフォルト値は「24000」です。ロングフォーム音声のデフォルト値は「24000」です。生成音声のデフォルト値は「24000」です。

pcm の有効な値は「8000」と「16000」です。デフォルト値は「16000」です。

タイプ: 文字列

必須: いいえ

SpeechMarkTypes

入力テキストに対して返されるスピーチマークのタイプ。

タイプ : 文字列の配列

配列メンバー: 最大数は 4 項目です。

有効な値 : `sentence | ssml | viseme | word`

必須 : いいえ

Text

合成する入力テキスト。`ssml` を `TextType` として指定した場合は、入力テキストの SSML 形式に従います。

型: 文字列

必須: はい

TextType

入力テキストがプレーンテキストか SSML かを指定します。デフォルト値はプレーンテキストです。詳細については、「SSML の使用」を参照してください。

型: 文字列

有効な値 : `ssml | text`

必須 : いいえ

VoiceId

合成に使用する音声 ID。DescribeVoices オペレーションを呼び出すことで、使用可能な音声 IDs のリストを取得できます。

型: 文字列

有効な値 : `Aditi | Amy | Astrid | Bianca | Brian | Camila | Carla | Carmen | Celine | Chantal | Conchita | Cristiano | Dora | Emma | Enrique | Ewa | Filiz | Gabrielle | Geraint | Giorgio | Gwyneth | Hans | Ines | Ivy | Jacek | Jan | Joanna | Joey | Justin | Karl | Kendra | Kevin | Kimberly | Lea | Liv | Lotte | Lucia | Lupe | Mads | Maja | Marlene | Mathieu | Matthew | Maxim | Mia | Miguel | Mizuki | Naja | Nicole | Olivia | Penelope | Raveena | Ricardo | Ruben | Russell | Salli | Seoyeon | Takumi | Tatyana | Vicki | Vitoria | Zeina | Zhiyu | Aria | Ayanda | Arlet | Hannah | Arthur | Daniel | Liam | Pedro | Kajal | Hiujin | Laura | Elin | Ida | Suvi | Ola | Hala | Andres | Sergio | Remi | Adriano | Thiago | Ruth | Stephen | Kazuha | Tomoko | Niamh | Sofie | Lisa | Isabelle | Zayd | Danielle | Gregory | Burcu`

必須: はい

レスポンスの構文

```
HTTP/1.1 200
Content-Type: ContentType
```

```
x-amzn-RequestCharacters: RequestCharacters

AudioStream
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

レスポンスでは、以下の HTTP ヘッダーが返されます。

ContentType

オーディオストリームのタイプを指定します。これは、リクエストの OutputFormat パラメータを反映したものでなければなりません。

- mp3 を OutputFormat としてリクエストした場合、返される ContentType は audio/mpeg です。
- ogg_vorbis を OutputFormat としてリクエストした場合、返される ContentType は audio/ogg です。
- pcm を OutputFormat としてリクエストした場合、返される ContentType は、符号付きの 16 ビット、1 つのチャネル (モノ)、リトルエンディアン形式の audio/pcm です。
- を jsonとしてリクエストした場合OutputFormat、ContentType返される は application/ ですx-json-stream。

RequestCharacters

合成された文字数。

レスポンスは、HTTP 本文として以下を返します。

AudioStream

合成された音声を含むストリーム。

エラー

EngineNotSupportedException

このエンジンは、指定された音声に対応していません。エンジンが対応する新しい音声を選択するか、エンジンを変更してオペレーションを再開してください。

HTTP ステータスコード：400

InvalidSampleRateException

指定されたサンプルレートが無効です。

HTTP ステータスコード：400

InvalidSsmlException

提供された SSML が無効です。SSML 構文、タグと値のスペルを確認し、もう一度試してください。

HTTP ステータスコード：400

LanguageNotSupportedException

指定された言語は、現在 Amazon Polly がこの容量でサポートしていません。

HTTP ステータスコード：400

LexiconNotFoundException

Amazon Polly は、指定されたレキシコンを見つけることができません。これは、レキシコンが見つからない、その名前のスペルが間違っている、または別のリージョンにあるレキシコンが指定されていることが原因である可能性があります。

レキシコンが存在すること、リージョンにあること (ListLexicons を参照)、名前のスペルが正しいことを確認します。その後、もう一度試してください。

HTTP ステータスコード: 404

MarksNotSupportedForFormatException

選択された `OutputFormat` では、スピーチマークはサポートされていません。スピーチマークは、`json` 形式のコンテンツに対してのみ使用できます。

HTTP ステータスコード：400

ServiceFailureException

不明な状態が原因で、サービス障害が発生しました。

HTTP ステータスコード：500

SsmlMarksNotSupportedForTextTypeException

SSML スピーチマークは、プレーンテキストタイプの入力ではサポートされていません。

HTTP ステータスコード：400

TextLengthExceededException

「テキスト」パラメータの値が、許容される上限より長くなっています。SynthesizeSpeech API の場合、入力テキストの制限は合計で最大 6,000 文字で、そのうち請求対象文字は 3,000 文字までに制限されています。StartSpeechSynthesisTask API の場合、最大 200,000 文字で、そのうち請求対象文字は 100,000 文字までに制限されています。SSML タグは、課金対象文字としてカウントされません。

HTTP ステータスコード：400

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS コマンドラインインターフェイス
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK for JavaScript V3
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V3

データ型

以下のデータ型 (タイプ) がサポートされています。

- Lexicon
- LexiconAttributes
- LexiconDescription
- SynthesisTask
- Voice

Lexicon

レキシコン名とレキシコンコンテンツを文字列形式で提供します。詳細については、Pronunciation Lexicon Specification (PLS) バージョン 1.0 を参照してください。

コンテンツ

Content

文字列形式のレキシコンコンテンツ。レキシコンのコンテンツは PLS 形式である必要があります。

タイプ: 文字列

必須: いいえ

Name

レキシコンの名前。

型: 文字列

パターン：`[0-9A-Za-z]{1,20}`

必須: いいえ

その他の参照資料

言語固有の AWS SDK でこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++
- AWS Java V2 用 SDK
- AWS ルビー V3 用 SDK

LexiconAttributes

語彙素の数や言語コードなど、レキシコンを記述するメタデータが含まれています。詳細については、レキシコンの管理を参照してください。

コンテンツ

Alphabet

レキシコンで使用される音声記号。有効な値は、ipa および x-sampa です。

タイプ: 文字列

必須: いいえ

LanguageCode

レキシコンが適用される言語コード。「en」などの言語コードを持つレキシコンは、すべての英語の言語 (en-GB、en-US、en-AUS、en-WLS など) に適用されます。

型: 文字列

有効な値 : `arb | cmn-CN | cy-GB | da-DK | de-DE | en-AU | en-GB | en-GB-WLS | en-IN | en-US | es-ES | es-MX | es-US | fr-CA | fr-FR | is-IS | it-IT | ja-JP | hi-IN | ko-KR | nb-NO | nl-NL | pl-PL | pt-BR | pt-PT | ro-RO | ru-RU | sv-SE | tr-TR | en-NZ | en-ZA | ca-ES | de-AT | yue-CN | ar-AE | fi-FI | en-IE | nl-BE | fr-BE`

必須 : いいえ

LastModified

レキシコンが最後に変更された日付 (タイムスタンプ値)。

型: タイムスタンプ

必須: いいえ

LexemesCount

レキシコン内の語彙素の数。

タイプ: 整数

必須: いいえ

LexiconArn

レキシコンの Amazon リソースネーム (ARN)。

タイプ: 文字列

必須: いいえ

Size

レキシコンの合計サイズ (文字単位)。

タイプ: 整数

必須 : いいえ

その他の参照資料

言語固有の AWS SDK でこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++
- AWS Java V2 用 SDK
- AWS ルビー V3 用 SDK

LexiconDescription

レキシコンのコンテンツについて説明します。

コンテンツ

Attributes

レキシコンメタデータを提供します。

タイプ：LexiconAttributes オブジェクト

必須: いいえ

Name

レキシコンの名前。

型: 文字列

パターン：`[0-9A-Za-z]{1,20}`

必須: いいえ

その他の参照資料

言語固有の AWS SDK でこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++
- AWS Java V2 用 SDK
- AWS ルビー V3 用 SDK

SynthesisTask

SynthesisTask 音声合成タスクに関する情報を提供する オブジェクト。

内容

CreationTime

合成タスクが開始された時刻のタイムスタンプ。

型: タイムスタンプ

必須: いいえ

Engine

音声合成の入力テキストを処理するときに使用する Amazon standardPolly のエンジン (neural、、long-formまたは generative) を指定します。 Amazon Polly 選択したエンジンでサポートされていない音声を使用すると、エラーが発生します。

型: 文字列

有効な値 : standard | neural | long-form | generative

必須 : いいえ

LanguageCode

合成タスクのオプションの言語コード。これは、インド英語 (en-IN) とヒンディー語 (hi-IN) のどちらにも使用できる Aditi などのバイリンガル音声を使用する場合にのみ必要です。

バイリンガル音声の使用時に言語コードが指定されていない場合、Amazon Polly はバイリンガル音声のデフォルト言語を使用します。音声のデフォルト言語は、 LanguageCodeパラメータの DescribeVoicesオペレーションによって返される言語です。例えば、言語コードが指定されていない場合、Aditi はヒンディー語ではなくインド英語を使用します。

型: 文字列

有効な値 : arb | cmn-CN | cy-GB | da-DK | de-DE | en-AU | en-GB | en-GB-WLS | en-IN | en-US | es-ES | es-MX | es-US | fr-CA | fr-FR | is-IS | it-IT | ja-JP | hi-IN | ko-KR | nb-NO | nl-NL | pl-PL | pt-BR | pt-PT | ro-RO | ru-RU | sv-SE | tr-TR | en-NZ | en-ZA | ca-ES | de-AT | yue-CN | ar-AE | fi-FI | en-IE | nl-BE | fr-BE

必須：いいえ

LexiconNames

合成時にサービスが適用する 1 つ以上の発音レキシコン名のリスト。レキシコンは、レキシコンの言語が音声の言語と同じ場合にのみ適用されます。

タイプ：文字列の配列

配列メンバー: 5 つの項目の最大数。

Pattern: `[0-9A-Za-z]{1,20}`

必須: いいえ

OutputFormat

返された出力がエンコードされる形式。オーディオストリームの場合、これは mp3、ogg_vorbis、または pcm になります。スピーチマークの場合、これは json になります。

型: 文字列

有効な値：`json | mp3 | ogg_vorbis | pcm`

必須：いいえ

OutputUri

出力音声ファイルのパスウェイ。

タイプ: 文字列

必須: いいえ

RequestCharacters

合成された請求可能な文字の数。

タイプ: 整数

必須: いいえ

SampleRate

Hz で指定した音声周波数。

mp3 および ogg_vorbis の有効な値は「8000」、「16000」、「22050」、および「24000」です。標準音声のデフォルト値は「22050」です。ニューラル音声のデフォルト値は「24000」です。ロングフォーム音声のデフォルト値は「24000」です。生成音声のデフォルト値は「24000」です。

pcm の有効な値は「8000」と「16000」です。デフォルト値は「16000」です。

タイプ: 文字列

必須: いいえ

SnsTopicArn

音声合成タスクのステータス通知を提供するためにオプションで使用される SNS トピックの ARN。

型: 文字列

パターン：`^arn:aws(-(cn|iso(-b)?|us-gov))?:sns:[a-z0-9_-]{1,50}:\d{12}:[a-zA-Z0-9_-]{1,251}([a-zA-Z0-9_-]{0,5}|\.fifo)$`

必須: いいえ

SpeechMarkTypes

入力テキストに対して返されるスピーチマークのタイプ。

タイプ：文字列の配列

配列メンバー: 最大数は 4 項目です。

有効な値：`sentence | ssml | viseme | word`

必須：いいえ

TaskId

Amazon Polly で生成された音声合成タスクの識別子。

型: 文字列

パターン：`^[a-zA-Z0-9_-]{1,100}$`

必須: いいえ

TaskStatus

個々の音声合成タスクの現在のステータス。

型: 文字列

有効な値 : scheduled | inProgress | completed | failed

必須 : いいえ

TaskStatusReason

特定の音声合成タスクの現在のステータスの理由 (タスクが失敗した場合のエラーを含む)。

タイプ: 文字列

必須: いいえ

TextType

入力テキストがプレーンテキストか SSML かを指定します。デフォルト値はプレーンテキストです。

型: 文字列

有効な値 : ssml | text

必須 : いいえ

VoiceId

合成に使用する音声 ID。

型: 文字列

有効な値 : Aditi | Amy | Astrid | Bianca | Brian | Camila | Carla | Carmen | Celine | Chantal | Conchita | Cristiano | Dora | Emma | Enrique | Ewa | Filiz | Gabrielle | Geraint | Giorgio | Gwyneth | Hans | Ines | Ivy | Jacek | Jan | Joanna | Joey | Justin | Karl | Kendra | Kevin | Kimberly | Lea | Liv | Lotte | Lucia | Lupe | Mads | Maja | Marlene | Mathieu | Matthew | Maxim | Mia | Miguel | Mizuki | Naja | Nicole | Olivia | Penelope | Raveena | Ricardo | Ruben | Russell | Salli | Seoyeon | Takumi | Tatyana | Vicki | Vitoria | Zeina | Zhiyu | Aria | Ayanda | Arlet | Hannah | Arthur | Daniel | Liam | Pedro | Kajal | Hiujin | Laura

```
| Elin | Ida | Suvi | Ola | Hala | Andres | Sergio | Remi | Adriano
| Thiago | Ruth | Stephen | Kazuha | Tomoko | Niamh | Sofie | Lisa |
Isabelle | Zayd | Danielle | Gregory | Burcu
```

必須：いいえ

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

Voice

音声の説明。

内容

AdditionalLanguageCodes

デフォルトの言語に加えて、指定された音声で使用可能な言語の追加コード。

例えば、Aditi のデフォルト言語は、その言語に最初に使用されたインド英語 (en-IN) です。Aditi はバイリンガルで、インド英語とヒンディー語の両方を流暢に話すため、このパラメータには hi-IN コードが表示されます。

タイプ：文字列の配列

有効な値: `arb | cmn-CN | cy-GB | da-DK | de-DE | en-AU | en-GB | en-GB-WLS | en-IN | en-US | es-ES | es-MX | es-US | fr-CA | fr-FR | is-IS | it-IT | ja-JP | hi-IN | ko-KR | nb-NO | nl-NL | pl-PL | pt-BR | pt-PT | ro-RO | ru-RU | sv-SE | tr-TR | en-NZ | en-ZA | ca-ES | de-AT | yue-CN | ar-AE | fi-FI | en-IE | nl-BE | fr-BE`

必須：いいえ

Gender

音声の性別。

型: 文字列

有効な値：`Female | Male`

必須：いいえ

Id

Amazon Polly が割り当てた音声 ID。これは、SynthesizeSpeech オペレーションを呼び出しているときに指定する ID です。

型: 文字列

有効な値：`Aditi | Amy | Astrid | Bianca | Brian | Camila | Carla | Carmen | Celine | Chantal | Conchita | Cristiano | Dora | Emma | Enrique | Ewa`

| Filiz | Gabrielle | Geraint | Giorgio | Gwyneth | Hans | Ines | Ivy | Jacek | Jan | Joanna | Joey | Justin | Karl | Kendra | Kevin | Kimberly | Lea | Liv | Lotte | Lucia | Lupe | Mads | Maja | Marlene | Mathieu | Matthew | Maxim | Mia | Miguel | Mizuki | Naja | Nicole | Olivia | Penelope | Raveena | Ricardo | Ruben | Russell | Salli | Seoyeon | Takumi | Tatyana | Vicki | Vitoria | Zeina | Zhiyu | Aria | Ayanda | Arlet | Hannah | Arthur | Daniel | Liam | Pedro | Kajal | Hiujin | Laura | Elin | Ida | Suvi | Ola | Hala | Andres | Sergio | Remi | Adriano | Thiago | Ruth | Stephen | Kazuha | Tomoko | Niamh | Sofie | Lisa | Isabelle | Zayd | Danielle | Gregory | Burcu

必須：いいえ

LanguageCode

音声の言語コード。

型: 文字列

有効な値：arb | cmn-CN | cy-GB | da-DK | de-DE | en-AU | en-GB | en-GB-WLS | en-IN | en-US | es-ES | es-MX | es-US | fr-CA | fr-FR | is-IS | it-IT | ja-JP | hi-IN | ko-KR | nb-NO | nl-NL | pl-PL | pt-BR | pt-PT | ro-RO | ru-RU | sv-SE | tr-TR | en-NZ | en-ZA | ca-ES | de-AT | yue-CN | ar-AE | fi-FI | en-IE | nl-BE | fr-BE

必須：いいえ

LanguageName

人が読み取り可能な英語の言語の名前。

タイプ: 文字列

必須: いいえ

Name

音声の名前 (Salli、Kendra など)。これは、アプリケーションに表示される、人が読み取り可能な音声名を提供します。

タイプ: 文字列

必須: いいえ

SupportedEngines

特定の音声でサポートされているエンジン (`standard`、`neural`、`long-form`または `generative`) を指定します。

タイプ：文字列の配列

有効な値: `standard | neural | long-form | generative`

必須：いいえ

その他の参照資料

言語固有の AWS SDKs のいずれかでこの API を使用する方法の詳細については、以下を参照してください。

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

Amazon Polly のドキュメント履歴

次の表に、Amazon Polly 開発者ガイドの各リリースの重要な変更点を示します。このドキュメントの更新に関する通知を受け取るには、RSS フィードにサブスクライブできます。

- ドキュメントの最終更新日: 2024 年 3 月 28 日

変更	説明	日付
新しい生成音声エンジンが追加されました	Amazon Polly は、生成バリアント Amy、Matthew、Ruth の 3 つの英語音声を使用して、より長いコンテンツ用に設計された生成音声エンジンを提供するようになりました。詳細については、「生成音声」を参照してください。	2024 年 3 月 28 日
NTTS 用の新しい音声の追加	Amazon Polly が NTTS トルコ語音声 Burcu を提供するようになりました。NTTS 音声のリストについては、ニューラル音声を参照してください。	2024 年 2 月 14 日
新しいロングフォーム音声エンジンが追加されました	Amazon Polly では、長いコンテンツ用に設計されたロングフォーム音声エンジンの提供が開始され、Danielle、Gregory、Ruth の 3 人の en-US 音声が利用できます。詳細については、「ロングフォーム音声」を参照してください。	2023 年 11 月 16 日
NTTS 用の新しい音声の追加	Amazon Polly では、2 つの新しい NTTS 米国英語音声の Danielle と Gregory が新たに提供されるようになりまし	2023 年 10 月 5 日

	た。NTTS 音声のリストについては、ニューラル音声を参照してください。	
Amazon Polly for Windows	Amazon Polly Windows Speech Application Programming Interface (SAPI) プラグインはサポートされなくなります。	2023 年 9 月 26 日
Amazon Polly のクォータガイダンスを更新	Amazon Polly のクォータガイドを更新しました。例と用語の明確な説明が追加されました。最新情報については、「Amazon Polly のクォータ」を参照してください。	2023 年 8 月 17 日
NTTS 用の新しい音声の追加	Amazon Polly では、アラビア語湾岸方言 NTTS 音声の Zayd が新たに提供されるようになりました。NTTS 音声のリストについては、ニューラル音声を参照してください。	2023 年 8 月 16 日
NTTS 用の新しい音声の追加	Amazon Polly では、ベルギーフランス語 NTTS 音声の Isabelle が新たに提供されるようになりました。NTTS 音声のリストについては、ニューラル音声を参照してください。	2023 年 8 月 1 日

NTTS 用の新しい音声の追加	Amazon Polly では、ベルギーオランダ語 (Flemish) NTTS 音声の Lisa が新たに提供されるようになりました。NTTS 音声のリストについては、ニューラル音声を参照してください。	2023 年 6 月 7 日
NTTS 用の新しい音声の追加	Amazon Polly では、アイルランド英語 (Niamh) とデンマーク語 (Sofie) の 2 つの新しい NTTS 音声が新たに提供されるようになりました。NTTS 音声のリストについては、ニューラル音声を参照してください。	2023 年 5 月 30 日
Amazon Polly の IAM ガイダンスを更新	IAM ベストプラクティスに沿ってガイドを更新しました。詳細については、「IAM のセキュリティのベストプラクティス」を参照してください。	2023 年 4 月 19 日
WordPress 更新	Amazon Polly WordPress プラグインはサポートされなくなります。	2023 年 4 月 6 日

新しいリージョンの追加	Amazon Polly がアジアパシフィック (大阪) AWS リージョンで利用可能になりました。このリージョンはニューラル TTS (NTTS) をサポートしています。NTTS をサポートするリージョンのリストの詳細については、機能とリージョンの互換性を参照してください。	2023 年 4 月 5 日
NTTS 用の新しい音声の追加	Amazon Polly では、2 つの新しい日本語 NTTS 音声の Kazuha と Tomoko が新たに提供されるようになりました。NTTS 音声のリストについては、ニューラル音声を参照してください。	2023 年 2 月 7 日
NTTS 用の新しい音声の追加	Amazon Polly では、2 つの新しい米国英語 NTTS 音声の Stephen と Ruth が新たに提供されるようになりました。NTTS 音声のリストについては、ニューラル音声を参照してください。	2023 年 1 月 31 日

NTTS 用の新しい音声の追加	Amazon Polly では、ブラジルポルトガル語 (Thiago)、カスティリヤスペイン語 (Sergio)、フランス語 (Rémi)、イタリア語 (Adriano)、メキシコスペイン語 (Andrés) の新しい NTTS 音声が新たに提供されるようになりました。NTTS 音声のリストについては、ニューラル音声を参照してください。	2023 年 1 月 24 日
NTTS 用の新しい音声の追加	Amazon Polly では、アラビア語 (Hala) とポーランド語 (Ola) の NTTS 音声が新たに提供されるようになりました。NTTS 音声のリストについては、ニューラル音声を参照してください。	2022 年 11 月 17 日
リリース AWS PrivateLink サポート	Amazon Polly が AWS PrivateLink サポートを開始しました。詳細については、「Using Amazon Polly with VPC endpoints」を参照してください。	2022 年 11 月 9 日
NTTS 用の新しい音声と言語の追加	Amazon Polly では、フィンランド語 (Suvi)、ノルウェー語 (Ida)、スウェーデン語 (Elin) の NTTS 音声が新たに提供されるようになりました。NTTS 音声のリストについては、ニューラル音声を参照してください。	2022 年 11 月 8 日

NTTS 用の新しい音声の追加	Amazon Polly では、ドイツ語 NTTS 音声の Laura が新たに提供されるようになりました。NTTS 音声のリストについては、ニューラル音声を参照してください。	2022 年 11 月 2 日
新しいリージョンの追加	Amazon Polly が欧州 (パリ) AWS リージョンで利用可能になりました。このリージョンはニューラル TTS (NTTS) をサポートしています。NTTS をサポートするリージョンのリストの詳細については、機能とリージョンの互換性を参照してください。	2022 年 9 月 22 日
NTTS 用の新しい音声と言語の追加	Amazon Polly では、広東語 NTTS 音声の Hiujin が新たに提供されるようになりました。NTTS 音声のリストについては、ニューラル音声を参照してください。	2022 年 9 月 20 日
新しいリージョンの追加	Amazon Polly が、アジアパシフィック (ムンバイ) AWS リージョンで使用できるようになりました。このリージョンはニューラル TTS (NTTS) をサポートしています。NTTS をサポートするリージョンのリストの詳細については、機能とリージョンの互換性を参照してください。	2022 年 9 月 1 日

NTTS 用の新しい音声の追加	Amazon Polly では、標準中国語音声の Zhiyu が NTTS 音声として提供されるようになりました。NTTS 音声のリストについては、ニューラル音声を参照してください。	2022 年 8 月 23 日
NTTS 用の新しい音声の追加	Amazon Polly では、ヒンディー語 NTTS 音声の Kajal が新たに提供されるようになりました。NTTS 音声のリストについては、ニューラル音声を参照してください。	2022 年 7 月 27 日
NTTS 用の新しい音声の追加	Amazon Polly では、米国スペイン語 (Pedro)、ドイツ語 (Daniel)、カナダフランス語 (Liam)、および英国英語 (Arthur) の NTTS 音声が新たに提供されるようになりました。NTTS 音声のリストについては、ニューラル音声を参照してください。	2022 年 6 月 28 日
NTTS 用の新しい音声の追加	Amazon Polly では、ポルトガル語 (ブラジル) 音声の Vitória が NTTS 音声として提供されるようになりました。NTTS 音声のリストについては、ニューラル音声を参照してください。	2022 年 4 月 27 日

NTTS 用の新しい音声の追加	Amazon Polly では、ポルトガル語 (欧州) 音声の Inês が NTTS 音声として提供されるようになりました。NTTS 音声のリストについては、ニューラル音声を参照してください。	2022 年 4 月 26 日
NTTS 用の新しい音声と言語の追加	Amazon Polly では、ドイツ語 (オーストリア) 言語と NTTS 音声の Hannah が新たに提供されるようになりました。NTTS 音声のリストについては、ニューラル音声を参照してください。	2022 年 4 月 19 日
NTTS 用の新しい音声と言語の追加	Amazon Polly では、スペイン語 (メキシコ) 音声の Mia が NTTS 音声として提供されるようになりました。新しい言語として、カタロニア語が NTTS 音声 Arlet と共に追加されました。NTTS 音声のリストについては、ニューラル音声を参照してください。	2022 年 3 月 22 日
NTTS 用の新しい音声の追加	Amazon Polly では、日本語音声の Takumi が NTTS 音声として提供されるようになりました。NTTS 音声のリストについては、ニューラル音声を参照してください。	2021 年 12 月 6 日

NTTS 用の新しい音声の追加	Amazon Polly では、フランス語音声の Léa が NTTS 音声として提供されるようになりました。NTTS 音声のリストについては、ニューラル音声を参照してください。	2021 年 11 月 18 日
NTTS 用の新しい音声の追加	Amazon Polly では、イタリア語の音声 Bianca とスペイン語 (欧州) 音声 Lucia が NTTS 音声として提供されるようになりました。NTTS 音声のリストについては、ニューラル音声を参照してください。	2021 年 11 月 8 日
NTTS 用の新しい音声の追加	Amazon Polly では、南アフリカ英語音声 Ayanda が新たに提供されるようになりました。この音声は NTTS 音声としてのみ使用できます。NTTS 音声のリストについては、ニューラル音声を参照してください。	2021 年 9 月 1 日
新しいリージョンの追加	Amazon Polly が アフリカ (ケープタウン) の AWS リージョンで利用可能になりました。このリージョンはニューラル TTS (NTTS) をサポートしています。NTTS をサポートするリージョンのリストの詳細については、機能とリージョンの互換性を参照してください。	2021 年 9 月 1 日

新しい言語と音声の追加	Amazon Polly ではニュージーランド英語 (en-NZ) がサポートされるようになりました。新しい NTTS 音声である Aria は、ニュージーランド英語とマオリ語の一部を話します。	2021 年 8 月 24 日
新機能	Amazon Polly では、ニューラル音声である Matthew と Joanna については、会話型の話し方がデフォルトバージョンになります。会話型の話し方の参照が削除されました。	2021 年 6 月 28 日
NTTS 用の新しい音声の追加	Amazon Polly では、ドイツ語音声の Vicki が NTTS 音声として提供されるようになりました。	2021 年 6 月 15 日
新しい音声の追加	フランス語 (カナダ) (fr-CA) のロケールに、新しい女性の音声、Gabrielle が追加されました。この音声は高品質で、NTTS 音声としてのみ利用可能です。すべてのニューラル音声と同様に、特定のリージョンでのみ利用可能です。リージョンのリストについては、機能とリージョンの互換性を参照してください。	2021 年 6 月 1 日
NTTS 用の新しい音声の追加	Amazon Polly では、韓国語音声の Seoyeon が NTTS 音声として提供されるようになりました。	2021 年 5 月 11 日

NTTS 用の新しいリージョンの追加	Amazon Polly は、カナダ (中部) AWS リージョンでニューラル TTS (NTTS) をサポートするようになりました。NTTS の詳細については、機能とリージョンの互換性を参照してください。	2021 年 3 月 17 日
新しい音声をニュースキャスタースタイルに利用可能に	ニュースキャスターの話し方の Matthew、Joanna、および Lupe の音声に加えて、Amazon Polly では、この話し方の追加のオプションが提供されるようになりました。ニューラルエンジンを使用して、イギリス英語 の Amy の音声をニュースキャスタースタイルで利用できます。詳細については、「NTTS の話し方」を参照してください。	2020 年 11 月 10 日
NTTS 用の新しいリージョンの追加	NTTS の既存のリージョン (us-east-1、us-west-2、eu-west-1、ap-southeast-2) に加えて、ニューラル音声が (ap-northeast-1 (東京)、ap-southeast-1 (シンガポール)、eu-central-1 (フランクフルト)、および eu-west-2 (ロンドン) のリージョンでサポートされるようになりました。NTTS の詳細については、機能とリージョンの互換性を参照してください。	2020 年 9 月 3 日

新しい音声の追加	米国英語 (en-US) で、子供の音声の Ivy と Justin に加えて、新たに男の子の音声の Kevin が追加されました。この新しい音声は非常に高品質で、NTTS 音声としてのみ利用可能です。この音声はすべてのニューラル音声と同様に、us-east-1 (バージニア北部)、us-west-2 (オレゴン)、eu-west-1 (アイルランド)、および ap-southeast-2 (シドニー) のリージョンでのみサポートされます。詳細については、「NTTS の音声」を参照してください。	2020 年 6 月 16 日
新しい音声をニュースキャスタースタイルに利用可能に	ニュースキャスターの話し方の Matthew と Joanna の音声に加えて、Amazon Polly では、この話し方の追加のオプションが提供されるようになりました。ニューラルエンジンを使用して、スペイン語 (米国) の Lupe の音声をニュースキャスタースタイルに利用できます。詳細については、「NTTS の話し方」を参照してください。	2020 年 4 月 16 日

新機能	ニュースキャスターの話し方に加えて、Amazon Polly では、2 つ目の話し方として NTTS が提供され、テキストから音節へのさらに良質な合成に役立ちます。会話型の話し方では、ニューラルシステムを使用して、より親しみのある表現力豊かな会話型の話し方で音声が生成され、多くのユースケースに利用できます。詳細については、「NTTS の話し方」を参照してください。	2019 年 11 月 25 日
新しい音声の追加	Camila (女性、ポルトガル語-ブラジル) と Lupe (女性、スペイン語-米国) の 2 つの新しい音声が追加されました。	2019 年 10 月 23 日
新機能の追加	Amazon Polly for Windows プラグインの追加により、すべての Amazon Polly 音声を Windows SAPI 準拠のアプリケーションに組み込むことができます。	2019 年 9 月 26 日

主な新機能	Amazon Polly は、リリース以降に Amazon Polly でサポートされている標準 text-to-speech (TTS) 音声に加えて、さらに高品質の音声を提供できる、改良されたニューラル TTS (NTTS) システムを提供するようになりました。これにより、可能な限り最も自然で人間のような text-to-speech 音声が提供されます。詳細については、「ニューラルテキスト読み上げ」を参照してください。	2019 年 7 月 30 日
新しい音声の追加	新しい音声: Lucia (女性、スペイン語)、Bianca (女性、イタリア語) が追加されました。	2018 年 8 月 2 日
新しい言語の追加	新しい言語: メキシコスペイン語 (es-MX) が追加されました。この言語では、Mia の女性の音声を使用します。	2018 年 8 月 2 日
新しい言語の追加	新しい言語が追加されました。ヒンディー語 (hi-IN)。この音声では、Aditi の女性の声が使用されます。この声はインド英語にも使用されるため、Aditi は Amazon Polly で最初のバイリンガル音声となります。	2018 年 8 月 2 日
新機能の追加	長いテキストの音声合成 (最大 100,000 課金対象文字) を追加しました。	2018 年 7 月 17 日

新しい SSML 機能の追加	合成音声の最大時間が追加されました。	2018 年 7 月 17 日
新しい音声の追加	新しい音声を追加しました。Léa (女性、フランス語)。	2018 年 6 月 5 日
リージョンの拡張	すべての商業地域への Amazon Polly サービスの拡張。	2018 年 6 月 4 日
新しい言語の追加	新しい言語が追加されました。韓国語 (ko-KR)。	2018 年 6 月 4 日
拡張された機能	Amazon Translate 機能の追加を含む Amazon Polly WordPress プラグイン機能。Amazon Translate	2018 年 6 月 4 日
新しい音声の追加	2 つの新しい音声が追加されました。Aditi (女性、インド英語) および Seoyeon (女性、韓国語)。	2017 年 11 月 15 日
新機能	SSML 機能の拡張と同様に、スピーチマーク機能の追加。	2017 年 4 月 19 日
新しいガイド	これは Amazon Polly 開発者ガイドの初回リリースです。	2016 年 11 月 30 日

AWS 用語集

AWS の最新の用語については、「AWS の用語集リファレンス」の「AWS 用語集」を参照してください。

翻訳は機械翻訳により提供されています。提供された翻訳内容と英語版の間で齟齬、不一致または矛盾がある場合、英語版が優先します。