aws

Entwicklerhandbuch

Amazon Polly

Amazon Polly: Entwicklerhandbuch

Table of Contents

Was ist Amazon Polly?

Amazon Polly ist ein Cloud-Service, der Text in naturgetreue Sprache umwandelt. Sie können Amazon Polly verwenden, um Anwendungen zu entwickeln, die das Engagement und die Barrierefreiheit erhöhen. Amazon Polly unterstützt mehrere Sprachen und beinhaltet eine Vielzahl lebensechter Stimmen. Mit Amazon Polly können Sie sprachgesteuerte Anwendungen entwickeln, die an mehreren Standorten funktionieren und die ideale Stimme für Ihre Kunden verwenden. Außerdem zahlen Sie nur für den Text, den Sie synthetisieren. Sie können die von Amazon Polly generierte Sprache auch zwischenspeichern und ohne zusätzliche Kosten wiedergeben.

Amazon Polly bietet viele Sprachoptionen, darunter generative, langformatige, neuronale und Standardoptionen text-to-speech (TTS). Diese Stimmen sorgen mithilfe neuer Technologien für maschinelles Lernen für bahnbrechende Verbesserungen der Sprachqualität, um möglichst natürliche und menschenähnliche Stimmen zu bieten. text-to-speech Die neuronale TTS-Technologie unterstützt auch einen Sprechstil von Nachrichtensendern, der auf Anwendungsfälle beim Erzählen von Nachrichten zugeschnitten ist.

Zu den häufigsten Anwendungsfällen für Amazon Polly gehören unter anderem: mobile Anwendungen wie Newsreader, Spiele, E-Learning-Plattformen, Barrierefreiheitsanwendungen für sehbehinderte Menschen und das schnell wachsende Segment Internet der Dinge (IoT).

Amazon Polly ist für die Verwendung mit regulierten Workloads nach HIPAA (Health Insurance Portability and Accountability Act von 1996) und dem Payment Card Industry Data Security Standard (PCI DSS) zertifiziert.

So funktioniert Amazon Polly

Amazon Polly wandelt den eingegebenen Text in lebensechte Sprache um. Um eine Amazon Polly Polly-Stimme zu verwenden, wählen Sie eine Sprachengine aus, rufen Sie eine Sprachsynthesemethode auf, geben Sie den Text an, den Sie synthetisieren möchten, und geben Sie dann ein Audioausgabeformat an. Amazon Polly synthetisiert dann den bereitgestellten Text zu einem hochwertigen Sprach-Audiostream.

- Text eingeben — Geben Sie den Text ein, den Sie synthetisieren möchten, und Amazon Polly gibt einen Audiostream zurück. Sie können die Eingabe als Klartext oder im SSML-Format (Speech Synthesis Markup Language) bereitstellen. Bei SSML können Sie verschiedene Sprachaspekte wie Aussprache, Lautstärke, Tonlage und Sprechgeschwindigkeit steuern. Weitere Informationen finden Sie unter Sprache aus SSML-Dokumenten generieren.

- Verfügbare Stimmen — Amazon Polly bietet ein Portfolio an Sprachen und eine Vielzahl von Stimmen, darunter eine zweisprachige Stimme (sowohl für Englisch als auch für Hindi). Bei den meisten Sprachen können Sie unter mehreren männlichen und weiblichen Stimmen wählen. Wenn Sie eine Sprachsynthese-Aufgabe starten, geben Sie die Sprach-ID an, und Amazon Polly verwendet diese Stimme, um den Text in Sprache umzuwandeln. Amazon Polly ist kein Übersetzungsdienst — die synthetisierte Sprache ist in derselben Sprache wie der Text. Zahlen, die als Ziffern dargestellt werden (z. B. 53, nicht dreiundfünfzig), werden in der Sprache der Stimme und nicht in der Sprache des Textes synthetisiert. Weitere Informationen finden Sie unter Stimmen in Amazon Polly.
- Ausgabeformat — Amazon Polly kann die synthetisierte Sprache in verschiedenen Formaten bereitstellen. Wählen Sie das Audioformat, das Ihren Anforderungen am besten entspricht. Beispielsweise können Sie die Sprache im Format MP3 oder im Ogg Vorbis-Format für die Wiedergabe durch Web- und Mobilanwendungen anfordern. Oder Sie könnten das PCM-Ausgabeformat für die Nutzung durch AWS IoT Geräte und Telefonielösungen anfordern. Für Telefonieanwendungen können Sie die Formate MU-Law oder A-Law verwenden.

> **Note**
>
> Beispiele für Amazon Polly Polly-Stimmen in Ihrem Browser finden Sie in der Amazon Polly Polly-Produktübersicht.

Vorteile

Zu den Vorteilen der Nutzung von Amazon Polly gehören:

- Hohe Qualität — Amazon Polly bietet leistungsstarke generative, langformige, neuronale und hochwertige text-to-speech (TTS) Stimmen. Diese Technologien synthetisieren natürliche Sprache mit hoher Aussprachegenauigkeit (einschließlich Abkürzungen, Akronymerweiterungen, Datums-/Uhrzeitinterpretationen und homographischer Disambiguierung).
- Niedrige Latenz — Amazon Polly erzielt schnelle Antworten, was es zu einer praktikablen Option für Anwendungsfälle mit niedriger Latenz wie Dialogsysteme macht.
- Support für ein großes Portfolio an Sprachen und Stimmen — Amazon Polly unterstützt Dutzende von Stimmen und Sprachen und bietet männliche und weibliche Sprachoptionen für die meisten Sprachen. Diese Zahl wird sich weiter erhöhen, da wir mehr neuronale Stimmen online bringen.

Die Stimmen Matthew und Joanna können auch den Sprechstil "Neural Newscaster (Neuronaler Newscaster)" verwenden, ähnlich dem, was man von einem professionellen Nachrichtenanker hören könnte.

- Kostengünstig — Das pay-per-use Modell von Amazon Polly bedeutet, dass keine Einrichtungskosten anfallen. Fangen Sie klein an und skalieren Sie sie, wenn Ihre Anwendung wächst.
- Cloud-basierte Lösung — TTS-Lösungen auf dem Gerät erfordern erhebliche Rechenressourcen, insbesondere CPU-Leistung, RAM und Festplattenspeicher. Dies kann zu höheren Entwicklungskosten und einem höheren Stromverbrauch auf Geräten wie Tablets, Smartphones usw. führen. Im Gegensatz dazu reduziert die in der durchgeführte TTS-Konvertierung die lokalen Ressourcenanforderungen AWS Cloud drastisch. Dies ermöglicht die Unterstützung aller verfügbaren Sprachen und Stimmen mit hervorragender Qualität. Darüber hinaus sind Sprachverbesserungen sofort für alle Endbenutzer verfügbar und erfordern keine zusätzlichen Updates für Geräte.

> **Note**
>
> Beispiele für Amazon Polly Polly-Stimmen in Ihrem Browser finden Sie in der Amazon Polly Polly-Produktübersicht.

Verwenden Sie zum ersten Mal?

Wenn Sie Amazon Polly zum ersten Mal verwenden, empfehlen wir Ihnen, die folgenden Abschnitte in der angegebenen Reihenfolge zu lesen:

1. So funktioniert Amazon Polly— In diesem Abschnitt werden verschiedene Amazon Polly Polly-Eingaben und -Optionen vorgestellt, mit denen Sie arbeiten können, um ein einfaches Erlebnis zu schaffen.
2. Erste Schritte mit Amazon Polly— In diesem Abschnitt richten Sie Ihr Konto ein und testen die Sprachsynthese von Amazon Polly.
3. Beispielcode und Anwendungen für Amazon Polly— Dieser Abschnitt enthält zusätzliche Beispiele, anhand derer Sie Amazon Polly erkunden können.

Amazon Polly mit einem AWS SDK verwenden

AWS Software Development Kits (SDKs) sind für viele beliebte Programmiersprachen verfügbar. Jedes SDK bietet eine API, Codebeispiele und Dokumentation, die es Entwicklern erleichtern, Anwendungen in ihrer bevorzugten Sprache zu erstellen.

SDK-Dokumentation	Codebeispiele
AWS SDK für C++	AWS SDK für C++ Codebeispiele
AWS CLI	AWS CLI Codebeispiele
AWS SDK für Go	AWS SDK für Go Codebeispiele
AWS SDK für Java	AWS SDK für Java Codebeispiele
AWS SDK für JavaScript	AWS SDK für JavaScript Codebeispiele
AWS SDK für Kotlin	AWS SDK für Kotlin Codebeispiele
AWS SDK für .NET	AWS SDK für .NET Codebeispiele
AWS SDK für PHP	AWS SDK für PHP Codebeispiele
AWS -Tools für PowerShell	AWS -Tools für PowerShell Codebeispiele
AWS SDK für Python (Boto3)	AWS SDK für Python (Boto3) Codebeispiele
AWS SDK für Ruby	AWS SDK für Ruby Codebeispiele
AWS SDK für Rust	AWS SDK für Rust Codebeispiele
AWS SDK für SAP ABAP	AWS SDK für SAP ABAP Codebeispiele
AWS SDK für Swift	AWS SDK für Swift Codebeispiele

Beispiel für die Verfügbarkeit

Sie können nicht finden, was Sie brauchen? Fordern Sie ein Codebeispiel an, indem Sie unten den Link Provide feedback (Feedback geben) auswählen.

Erste Schritte mit Amazon Polly

Amazon Polly bietet mehrere API-Operationen, die Sie problemlos in Ihre vorhandenen Anwendungen integrieren können. Eine Liste der unterstützten Operationen finden Sie unter Aktionen.

Sie können fast alle der gleichen Operationen auf der Amazon Polly Polly-Konsole und dem AWS CLI ausführen. Sie können jedoch keine synthetisierte Sprache auf dem hören. AWS CLI Um mit Audio auf dem zu arbeiten AWS CLI, speichern Sie Ihren Text in einer Datei. Öffnen Sie die Datei anschließend in einer Audioanwendung Ihrer Wahl.

Sie haben die Wahl zwischen den folgenden Optionen:

- AWS SDKs — Wenn Sie die SDKs verwenden, werden Ihre Anfragen an Amazon Polly automatisch mit den von Ihnen angegebenen Anmeldeinformationen signiert und authentifiziert. Diese Variante ist die empfohlene Option für die Anwendungsprogrammierung.
- AWS CLI — Sie können Amazon Polly verwenden AWS CLI , ohne Code schreiben zu müssen.

Bevor Sie Amazon Polly zum ersten Mal verwenden, müssen Sie sich für AWS registrieren. Wenn Sie sich für Amazon Web Services (AWS) registrieren, wird Ihr AWS Konto automatisch für alle Dienste angemeldet AWS, einschließlich Amazon Polly. Ihnen werden nur die Dienste und Ressourcen in Rechnung gestellt, die Sie nutzen. Wenn Sie ein neuer AWS Kunde sind, können Sie kostenlos mit Amazon Polly beginnen. Weitere Informationen finden Sie unter AWS – kostenloses Nutzungskontingent.

In den folgenden Abschnitten werden die ersten Schritte mit Amazon Polly beschrieben.

Themen

Melden Sie sich an für AWS

Bevor Sie einen AWS Dienst, einschließlich Amazon Polly, nutzen können, müssen Sie sich für AWS registrieren.

Melden Sie sich an für ein AWS-Konto

Um loszulegen AWS, benötigen Sie eine AWS-Konto. Informationen zum Erstellen eines AWS-Konto finden Sie unter Erste Schritte mit einem AWS-Konto im AWS -Kontenverwaltung Referenzhandbuch.

Weitere Informationen zu IAM finden Sie unter:

- AWS Identity and Access Management (IAM)
- Erste Schritte mit IAM
- IAM Benutzerhandbuch

> **Note**
>
> Notieren Sie sich Ihre AWS Konto-ID. Sie benötigen sie in den nächsten Schritten.

Einrichtung des AWS CLI

Gehen Sie wie folgt vor, um das herunterzuladen und AWS CLI für die Verwendung mit Amazon Polly zu konfigurieren.

Um das einzurichten AWS Command Line Interface

1. Herunterladen und Konfigurieren von AWS CLI. Eine Anleitung finden Sie unter den folgenden Themen im AWS Command Line Interface -Benutzerhandbuch:
   - Erste Schritte mit dem AWS Command Line Interface
   - Konfiguration des AWS Command Line Interface
2. Fügen Sie in der AWS CLI AWS Konfigurationsdatei ein benanntes Profil für den Administratorbenutzer hinzu. Sie können dieses Profil verwenden, wenn Sie die AWS CLI Befehle ausführen. Weitere Informationen zu benannten Profilen finden Sie unter Benannte Profile im AWS Command Line Interface Benutzerhandbuch.

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

Eine Liste der verfügbaren und von Amazon Polly unterstützten AWS Regionen finden Sie unter Regionen und Endpunkte in der. Allgemeine Amazon Web Services-Referenz

> **Note**
>
> Wenn Sie eine von Amazon Polly unterstützte Region verwenden, die Sie bei der Konfiguration von angegeben haben AWS CLI, lassen Sie die folgende Zeile in den AWS CLI Codebeispielen weg.
>
> ```
> --region aws-region
> ```

3. Geben Sie den folgenden Hilfebefehl in die Eingabeaufforderung ein, um die Einrichtung zu überprüfen.

   ```
   aws help
   ```

   Im Fenster sollte eine Liste gültiger AWS Befehle angezeigt werden. AWS CLI

Neukonfiguration des AWS CLI

Wenn Sie das bereits heruntergeladen und konfiguriert haben AWS CLI, ist Amazon Polly möglicherweise nicht verfügbar, es sei denn, Sie konfigurieren das neu. AWS CLI Mit dem folgenden Verfahren wird geprüft, ob dies erforderlich ist.

Um Amazon Polly zu reaktivieren von AWS CLI

1. Überprüfen Sie die Verfügbarkeit von Amazon Polly, indem Sie an der Eingabeaufforderung den folgenden AWS CLI Hilfebefehl eingeben.

   ```
   aws polly help
   ```

   Wenn Sie eine Beschreibung von Amazon Polly sehen und eine Liste gültiger Befehle im AWS CLI Fenster angezeigt wird, können Sie Amazon Polly sofort verwenden. AWS CLI In diesem Fall können Sie die übrigen Schritte dieser Anleitung überspringen. Fahren Sie fort mit Schritt 2, falls keine entsprechende Ausgabe angezeigt wird.

2. Aktivieren Sie Amazon Polly mit einer der beiden folgenden Optionen:

a. Deinstallieren Sie den und installieren Sie ihn erneut. AWS CLI

   Anweisungen finden Sie unter Installation von AWS Command Line Interface im AWS Command Line Interface Benutzerhandbuch.

   oder

b. Laden Sie die Datei service-2.json herunter.

   Führen Sie an der Eingabeaufforderung den folgenden Befehl aus.

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

3. Überprüfen Sie erneut die Verfügbarkeit von Amazon Polly.

```
aws polly help
```

   Die Beschreibung von Amazon Polly sollte sichtbar sein.

Beispiel für Sprachsynthese mit Amazon Polly

Diese Seite enthält ein kurzes Beispiel für die Sprachsynthese, das in der Konsole AWS CLI, und mit Python ausgeführt wurde. In diesem Beispiel wird die Sprachsynthese aus reinem Text durchgeführt, nicht aus SSML.

Console

Synthetisieren Sie Sprache auf der Konsole

1. Melden Sie sich bei der an AWS-Managementkonsole und öffnen Sie die Amazon Polly Polly-Konsole unter https://console.aws.amazon.com/polly/.
2. Wählen Sie die Registerkarte Text-to-Speech. Das Textfeld wird mit Beispieltext geladen, sodass Sie Amazon Polly schnell ausprobieren können.
3. Schalten Sie SSL aus.
4. Geben Sie den folgenden Text in das Eingabefeld ein oder kopieren Sie ihn in das Feld:

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

5. Wählen Sie unter Engine die Optionen Generativ, Long Form, Neural oder Standard aus.
6. Wählen Sie eine Sprache und AWS Region und dann eine Stimme aus. (Wenn Sie Neural für Engine auswählen, sind nur die Sprachen und Stimmen verfügbar, die NTTS unterstützen. Alle Standard- und Langform-Stimmen sind deaktiviert.)
7. Um die Rede sofort anzuhören, wählen Sie „Zuhören“.
8. Speichern Sie die Sprachausgabe auf einem der beiden folgenden Wege in einer Datei:

   a. Wählen Sie Herunterladen aus.

   b. Um zu einem anderen Dateiformat zu wechseln, erweitern Sie Zusätzliche Einstellungen, aktivieren Sie Einstellungen für das Sprachdateiformat, wählen Sie das gewünschte Dateiformat aus (z. B. OGG MP3, PCM, MU-law oder A-law), und wählen Sie dann Herunterladen aus.

AWS CLI

In dieser Übung rufen Sie den `SynthesizeSpeech` Vorgang auf, indem Sie den Eingabetext übergeben. Die resultierende Audioausgabe können Sie als Datei speichern und abspielen.

1. Führen Sie den `synthesize-speech` AWS CLI Befehl aus, um den Beispieltext in einer Audiodatei (`hello.mp3`) zu synthetisieren.

   Das folgende AWS CLI Beispiel ist für Unix, Linux und macOS formatiert. Ersetzen Sie unter Windows den Unix-Fortsetzungszeichen mit umgekehrtem Schrägstrich (\) am Ende jeder Zeile durch ein Caret (^) und setzen Sie den Eingabetext in vollständige Anführungszeichen („) und einfache Anführungszeichen (') für interne Tags.

   ```
   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
   ```

   Im Call to geben Sie einen Beispieltext an`synthesize-speech`, der von einer Stimme Ihrer Wahl synthetisiert werden soll. Sie müssen eine Sprach-ID (im folgenden Schritt erklärt) und ein Ausgabeformat angeben. Der Befehl speichert die resultierende Audioausgabe in der Datei `hello.mp3`. Zusätzlich zur MP3 Datei sendet der Vorgang die folgende Ausgabe an die Konsole.

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

2. Geben Sie die Datei `hello.mp3` wieder, um die Sprachausgabe zu überprüfen.

Python

Sie benötigen das AWS SDK für Python (Boto), um den Python-Beispielcode testen zu können. Detaillierte Anweisungen finden Sie unter AWS SDK für Python (Boto3).

Der Python-Code in diesem Beispiel führt die folgenden Aktionen aus:

- Ruft das AWS SDK für Python (Boto) auf, um eine `SynthesizeSpeech` Anfrage an Amazon Polly zu senden (indem Text als Eingabe bereitgestellt wird).
- Er greift auf den resultierenden Audiostream in der Antwort zu und speichert die Audioausgabe in einer Datei (`speech.mp3`) auf der lokalen Festplatte.

- Er spielt die Audiodatei mit dem Standard-Audioplayer des lokalen Systems ab.

Speichern Sie den Code in einer Datei ("example.py") und führen Sie die Datei aus.

```
"""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])
```

Umfassendere Beispiele finden Sie in den folgenden Artikeln:

- SSML auf der Konsole verwenden
- Lexika anwenden (Sprachsynthese)
- Beispielcode und Anwendungen für Amazon Polly

Synthetisieren von Sprache mit bidirektionalem Streaming

Amazon Polly bietet einen `StartSpeechSynthesisStream` Vorgang, der eine HTTP/2 Verbindung mit bidirektionaler Kommunikation zwischen Ihrer Anwendung und dem Service herstellt. Text fließt von Ihrer Anwendung zu Amazon Polly, während synthetisiertes Audio zurückfließt. Sie senden Text, sobald er verfügbar ist, und Amazon Polly gibt Audio zurück, während es synthetisiert wird, ohne dass eine Seite darauf wartet, dass die andere Seite fertig ist.

Dies ist nützlich, wenn Text schrittweise und nicht auf einmal erzeugt wird. Beispielsweise generiert ein Kundenservice-Chatbot, der auf einem Foundation-Modell auf Amazon Bedrock basiert, seine Antwort Token für Token. Mit bidirektionalem Streaming kann Ihre Anwendung jeden Textblock an Amazon Polly weiterleiten, während das Modell ihn erzeugt, und mit der Audiowiedergabe für den Anrufer beginnen, während das Modell den Rest der Antwort generiert.

Für diesen Vorgang sind die Generative Engine und ein AWS SDK erforderlich, das Event-Streams unterstützt. HTTP/2 Das Audio wird als Folge von Blöcken empfangen, die Ihre Anwendung zu einer vollständigen Audioausgabe zusammenfasst. Sprachzeichen werden von diesem Vorgang nicht unterstützt.

> **Note**
>
> AWS CLI (v1 und v2), AWS Tools für PowerShell (v4 und v5), Python und .NET v3 werden nicht unterstützt. Sie können die bidirektionale Streaming-API mit den folgenden SDKs verwenden: AWS SDK for Java 2.x, JavaScript v3, .NET v4, C++, Go v2, Kotlin, PHP v3, Ruby v3, Rust und Swift.

Themen

SynthesizeSpeech und verglichen StartSpeechSynthesisStream

SynthesizeSpeechist eine Anfrage-Antwort-Operation. Sie stellen den vollständigen Text in einer einzigen Anfrage bereit und erhalten das vollständige synthetisierte Audio in einer einzigen

Antwort. Es unterstützt alle Engines (Standard, Neural, Langform, Generativ), alle Ausgabeformate, einschließlich Sprachzeichen, und hat ein Textlimit von insgesamt 6.000 Zeichen (von denen nicht mehr als 3.000 fakturierte Zeichen sein können) pro Anfrage. Die Antwort streamt Audio zurück, sobald die ersten Byte verfügbar sind. Verwenden Sie diesen Vorgang, wenn Sie den gesamten Text im Voraus verfügbar haben.

StartSpeechSynthesisStreamist ein bidirektionaler Streaming-Vorgang. Es öffnet eine HTTP/2 Verbindung, über die Sie inkrementell Text senden und Audio empfangen, während es synthetisiert wird. Es gibt kein Textlimit pro Anfrage, da Text kontinuierlich gestreamt wird. Es erfordert die Generative Engine und unterstützt keine Sprachzeichen. Verwenden Sie diesen Vorgang, wenn Text inkrementell ankommt und Sie möchten, dass die Audioausgabe beginnt, bevor alle Eingaben verfügbar sind. Zu den gängigen Szenarien gehören:

- Konversations-KI und Sprachassistenten. Ein großes Sprachmodell generiert Antworttext in kleinen Blöcken (Tokens). Leiten Sie jeden eingehenden Textblock an Amazon Polly weiter, sodass der Benutzer Sprache hört, während das Modell noch generiert wird.
- Real-time Übersetzung. Ein Übersetzungssystem erzeugt den übersetzten Text Segment für Segment. Streamen Sie jedes Segment zur Synthese, ohne auf den Abschluss der vollständigen Übersetzung warten zu müssen.
- Long-form Inhalt, der SynthesizeSpeech Grenzen überschreitet. Text, der länger als 6.000 Zeichen ist, kann kontinuierlich gestreamt werden, ohne dass er in mehrere Anfragen aufgeteilt wird oder Abschnittsgrenzen verwaltet werden müssen.

Vergleich von und SynthesizeSpeech StartSpeechSynthesisStream

Aspekt	SynthesizeSpeech	StartSpeechSynthesisStream
Protocol (Protokoll)	Request-response	Bidirektionaler Eventstream () HTTP/2
Lieferung von Text	Volltext im Anfragetext	Eingabetext über TextEvent Nachrichten streamen
Audioübertragung	Audioantwort über HTTP-Antw orttext streamen	Audioantwort über AudioEvent Nachrichten streamen
Motorunterstützung	Standard, neuronal, langförmig, generativ	nur generativ

Aspekt	SynthesizeSpeech	StartSpeechSynthesisStream
SSML-Unterstützung	Ja (alle Engines; die unterstützten Tags variieren je nach Engine)	Ja (nur generative Engine-Tags)
Lexika	Ja	Ja
Sprachzeichen	Ja	Nein
Textlimit	Insgesamt 6.000 Zeichen (3.000 in Rechnung gestellt) pro Anfrage	Insgesamt 6.000 Zeichen (3.000 in Rechnung gestellt) pro TextEvent
AWS CLI Unterstützung	Ja	Nein (bidirektionales Streaming erfordert ein SDK)

Text senden und Audio empfangen

Eine bidirektionale Streaming-Sitzung beinhaltet das Öffnen einer Verbindung, das gleichzeitige Senden von Text und das Empfangen von Audio und das anschließende Schließen des Streams, wenn die Eingabe abgeschlossen ist. In den folgenden Abschnitten werden die einzelnen Phasen detailliert beschrieben.

Öffne den Stream

Ihre Anwendung ruft den StartSpeechSynthesisStreamVorgang über das SDK auf und gibt Syntheseparameter (`EngineVoiceId`,`OutputFormat`, und optional `LanguageCodeLexiconNames`,`SampleRate`) an. Das SDK stellt eine HTTP/2 Verbindung her und der bidirektionale Stream ist bereit, Eingabeereignisse zu akzeptieren.

Text senden

Der Client sendet eine oder mehrere TextEventNachrichten im Eingabestream. Jedes Ereignis kann gesendet werden, sobald Text verfügbar ist, ohne darauf warten zu müssen, dass die vollständige Eingabe bereit ist. Textereignisse müssen sich nicht an Satz- oder Satzgrenzen orientieren. Amazon Polly setzt den Text intern neu zusammen und erzeugt eine natürlich klingende Sprache, unabhängig davon, wie die Eingabe auf die Ereignisse aufgeteilt wird.

> **Note**
>
> Bei der Verwendung von SSML muss jedes SSML-Dokument in einem einzigen Dokument abgeschlossen sein. `TextEvent` Sie können SSML-Tags nicht auf mehrere Ereignisse aufteilen. Sie können jedoch Klartextereignisse und SSML-Ereignisse innerhalb desselben Streams mischen.

Der Stream unterliegt den folgenden Zeitlimits:

- Maximale Stream-Dauer: 10 Minuten. Amazon Polly schließt den Stream unabhängig von der Aktivität nach 10 Minuten. Wenn Ihr Inhalt mehr Zeit benötigt, öffnen Sie einen neuen Stream für den verbleibenden Text.
- Timeout im Leerlauf zwischen aufeinanderfolgenden Ereignissen: 5 Sekunden. Wenn 5 Sekunden lang kein Eingabeereignis gesendet wird, schließt Amazon Polly den Stream. Wenn Ihre Textquelle Pausen von mehr als 5 Sekunden hat, senden Sie ein Keep-Alive `TextEvent` mit einer leeren Zeichenfolge oder einem Leerzeichen, um das Timeout zu verhindern.

Erzwingen Sie die Synthese von gepuffertem Text durch Leeren

Standardmäßig entscheidet Amazon Polly, wann gepufferter Text auf der Grundlage natürlicher Sprachgrenzen synthetisiert werden soll. Dadurch wird die beste Audioqualität erzielt, aber das bedeutet, dass Audio möglicherweise nicht sofort zurückgegeben wird, nachdem Sie eine gesendet haben. `TextEvent`

Durch Flushing haben Sie die Kontrolle darüber, wann die Synthese stattfindet. Beim Leeren synthetisiert Amazon Polly sofort den gesamten Text, den es bisher gepuffert hat, unabhängig davon, ob der Text an einer natürlichen Grenze endet. Dies ist nützlich, wenn Ihre Textquelle zwischen logischen Abschnitten pausiert und Sie Audio für das, was bisher gesendet wurde, bereitstellen möchten.

Um zu leeren, stellen Sie den. FlushStreamConfiguration `Force`Parameter `true` auf ein`TextEvent`. Sie können auch eine leere Datei `TextEvent` mit gesetztem Flush-Flag senden, um die Synthese auszulösen, ohne neuen Inhalt hinzuzufügen.

Flushing ist ein Kompromiss. Die Einstellung `Force` auf die `true` Mitte des Satzes kann sich auf Aussprache und Intonation auswirken, da dem Synthesizer der Kontext für das Folgende fehlt. Um optimale Ergebnisse zu erzielen, sollten Sie Amazon Polly erlauben, wann immer

möglich bis an natürliche Grenzen zu puffern und die Synthese nur dann zu erzwingen, wenn die Latenzanforderungen dies erfordern.

Audio empfangen

Wenn Amazon Polly Text synthetisiert, gibt es AudioEventNachrichten im Ausgabestrom zurück. Jedes Ereignis enthält einen Teil von Audiodaten. Ihre Anwendung muss diese Datenblöcke akkumulieren (z. B. indem sie nacheinander in eine Datei oder einen Audiopuffer geschrieben werden), um die vollständige Audioausgabe zu erzeugen. Audioereignisse können eintreffen, während Sie noch Textereignisse senden.

Schließe den Stream

Wenn der gesamte Eingabetext gesendet wurde, sendet der Client eine CloseStreamEvent. Amazon Polly beendet die Verarbeitung des verbleibenden gepufferten Texts, sendet letzte Audioereignisse und gibt a zurück, StreamClosedEventdas die Gesamtzahl der synthetisierten Zeichen enthält. Senden Sie immer eine, `CloseStreamEvent` anstatt sich auf das Leeren zu verlassen, um den Stream zu beenden. Beim Schließen wird sichergestellt, dass der gesamte gepufferte Text synthetisiert und zurückgegeben wird.

Vollständige Informationen zu Anforderungsparametern, Ereignistypen und Fehlern finden Sie in der StartSpeechSynthesisStreamAPI-Referenz.

Codebeispiele

Das folgende Beispiel zeigt eine vollständige bidirektionale Streaming-Sitzung. Das Programm erstellt einen asynchronen Amazon Polly Polly-Client, öffnet einen Stream, der mit der Generative Engine und der MP3-Ausgabe konfiguriert ist, sendet Text als eine Reihe von `TextEvent` Nachrichten, sammelt die zurückgegebenen `AudioEvent` Chunks in einer Ausgabedatei und schließt den Stream mit einem. `CloseStreamEvent` Da Eingabe und Ausgabe gleichzeitig erfolgen, kommen Audiodaten an, bevor der gesamte Text gesendet wurde.

Java

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.polly.PollyAsyncClient;
import software.amazon.awssdk.services.polly.model.*;

import java.io.FileOutputStream;
import java.io.OutputStream;
```

```
import java.util.concurrent.CompletableFuture;
import java.util.concurrent.Semaphore;
import org.reactivestreams.Publisher;
import org.reactivestreams.Subscription;

public class BidirectionalStreamExample {

    public static void main(String[] args) throws Exception {
        try (PollyAsyncClient pollyClient = PollyAsyncClient.builder()
                .region(Region.US_EAST_1)
                .build()) {

        StartSpeechSynthesisStreamRequest request =
 StartSpeechSynthesisStreamRequest.builder()
                .engine(Engine.GENERATIVE)
                .voiceId(VoiceId.TIFFANY)
                .outputFormat(OutputFormat.MP3)
                .sampleRate("24000")
                .build();

        try (OutputStream audioOutput = new FileOutputStream("output.mp3")) {

        StartSpeechSynthesisStreamResponseHandler responseHandler =
                StartSpeechSynthesisStreamResponseHandler.builder()

 .subscriber(StartSpeechSynthesisStreamResponseHandler.Visitor.builder()
                                .onAudioEvent(audioEvent -> {
                                    try {
                                        byte[] audio =
 audioEvent.audioChunk().asByteArray();
                                        System.out.println("Received AudioEvent: " +
 audio.length + " bytes");
                                        audioOutput.write(audio);
                                    } catch (Exception e) {
                                        throw new RuntimeException(e);
                                    }
                                })
                                .onStreamClosedEvent(closedEvent -> {
                                    System.out.println("Stream closed. Characters
 synthesized: "
                                            + closedEvent.requestCharacters());
                                })
                                .onDefault(event -> {})
                                .build())
```

```
                        .onError(error -> {
                            System.err.println("Stream error: " +
error.getMessage());
                        })
                        .build();

        String[] textChunks = {
                "The weather forecast for today shows clear skies ",
                "with temperatures reaching twenty five degrees. ",
                "Tomorrow we expect some cloud cover in the morning ",
                "but it should clear up by the afternoon. ",
                "The rest of the week looks mostly sunny ",
                "with a slight chance of rain on Friday. ",
                "Overall a great week to spend time outdoors."
        };

        Publisher<StartSpeechSynthesisStreamActionStream> inputPublisher =
subscriber -> {
            subscriber.onSubscribe(new Subscription() {
                private final Semaphore permits = new Semaphore(0);
                private volatile boolean cancelled = false;

                @Override
                public void request(long n) {
                    permits.release((int) Math.min(n, Integer.MAX_VALUE));
                }

                @Override
                public void cancel() {
                    cancelled = true;
                    permits.release();
                }

                {
                    new Thread(() -> {
                        for (String chunk : textChunks) {
                            try { permits.acquire(); } catch (InterruptedException
e) { return; }
                            if (cancelled) return;
                            System.out.println("Sending TextEvent: " +
chunk.trim());

subscriber.onNext(StartSpeechSynthesisStreamActionStream.textEventBuilder()
                                    .text(chunk).textType(TextType.TEXT).build());
```

```
                            // Simulate delay between chunks (e.g. waiting for LLM
 tokens)
                            try { Thread.sleep(300); } catch (InterruptedException
 e) { return; }
                        }
                        if (!cancelled) {
                            subscriber.onNext(StartSpeechSynthesisStreamActionStream
                                    .closeStreamEventBuilder().build());
                            subscriber.onComplete();
                        }
                    }).start();
                }
            });
        };

        CompletableFuture<Void> future = pollyClient.startSpeechSynthesisStream(
                request, inputPublisher, responseHandler);

        future.join();
        } // audioOutput closed
        } // pollyClient closed
    }
}
```

JavaScript

```
import { PollyClient, StartSpeechSynthesisStreamCommand } from "@aws-sdk/client-
polly";
import { createWriteStream } from "fs";

const client = new PollyClient({ region: "us-east-1" });

const textChunks = [
  "The weather forecast for today shows clear skies ",
  "with temperatures reaching twenty five degrees. ",
  "Tomorrow we expect some cloud cover in the morning ",
  "but it should clear up by the afternoon. ",
  "The rest of the week looks mostly sunny ",
  "with a slight chance of rain on Friday. ",
  "Overall a great week to spend time outdoors.",
];

const sleep = (ms) => new Promise((resolve) => setTimeout(resolve, ms));
```

```
async function* createInputEvents() {
  for (const chunk of textChunks) {
    console.log(`Sending TextEvent: ${chunk.trim()}`);
    yield {
      TextEvent: {
        Text: chunk,
        TextType: "text",
      },
    };
    // Simulate delay between chunks (e.g. waiting for LLM tokens)
    await sleep(300);
  }

  yield { CloseStreamEvent: {} };
}

async function synthesizeStream() {
  const command = new StartSpeechSynthesisStreamCommand({
    Engine: "generative",
    VoiceId: "Tiffany",
    OutputFormat: "mp3",
    SampleRate: "24000",
    ActionStream: createInputEvents(),
  });

  const response = await client.send(command);
  const outputStream = createWriteStream("output.mp3");

  for await (const event of response.EventStream) {
    if (event.AudioEvent) {
      console.log(`Received AudioEvent: ${event.AudioEvent.AudioChunk.length}
 bytes`);
      outputStream.write(event.AudioEvent.AudioChunk);
    } else if (event.StreamClosedEvent) {
      console.log(
        `Stream closed. Characters synthesized:
 ${event.StreamClosedEvent.RequestCharacters}`
      );
    }
  }

  outputStream.end();
}
```

```
synthesizeStream().catch(console.error);
```

Go

```
package main

import (
 "context"
 "fmt"
 "os"
 "strings"
 "time"

 "github.com/aws/aws-sdk-go-v2/aws"
 "github.com/aws/aws-sdk-go-v2/config"
 "github.com/aws/aws-sdk-go-v2/service/polly"
 "github.com/aws/aws-sdk-go-v2/service/polly/types"
)

func main() {
 ctx := context.Background()

 cfg, err := config.LoadDefaultConfig(ctx, config.WithRegion("us-east-1"))
 if err != nil {
  fmt.Fprintf(os.Stderr, "Failed to load config: %v\n", err)
  os.Exit(1)
 }

 client := polly.NewFromConfig(cfg)

 output, err := client.StartSpeechSynthesisStream(ctx,
 &polly.StartSpeechSynthesisStreamInput{
  Engine:       types.EngineGenerative,
  VoiceId:      types.VoiceIdTiffany,
  OutputFormat: types.OutputFormatMp3,
  SampleRate:   aws.String("24000"),
 })
 if err != nil {
  fmt.Fprintf(os.Stderr, "Failed to start stream: %v\n", err)
  os.Exit(1)
 }
```

```
stream := output.GetStream()
defer stream.Close()

audioFile, err := os.Create("output.mp3")
if err != nil {
 fmt.Fprintf(os.Stderr, "Failed to create output file: %v\n", err)
 os.Exit(1)
}
defer audioFile.Close()

textChunks := []string{
 "The weather forecast for today shows clear skies ",
 "with temperatures reaching twenty five degrees. ",
 "Tomorrow we expect some cloud cover in the morning ",
 "but it should clear up by the afternoon. ",
 "The rest of the week looks mostly sunny ",
 "with a slight chance of rain on Friday. ",
 "Overall a great week to spend time outdoors.",
}

// Send text events in a goroutine
go func() {
 for _, chunk := range textChunks {
  fmt.Printf("Sending TextEvent: %s\n", strings.TrimSpace(chunk))
  err := stream.Send(ctx,
&types.StartSpeechSynthesisStreamActionStreamMemberTextEvent{
   Value: types.TextEvent{
    Text:     aws.String(chunk),
    TextType: types.TextTypeText,
   },
  })
  if err != nil {
   fmt.Fprintf(os.Stderr, "Failed to send text event: %v\n", err)
   return
  }
  // Simulate delay between chunks (e.g. waiting for LLM tokens)
  time.Sleep(300 * time.Millisecond)
 }

 // Signal end of input
 stream.Send(ctx,
&types.StartSpeechSynthesisStreamActionStreamMemberCloseStreamEvent{
  Value: types.CloseStreamEvent{},
 })
```

```
 }()

 // Receive audio events
 for event := range stream.Events() {
  switch v := event.(type) {
  case *types.StartSpeechSynthesisStreamEventStreamMemberAudioEvent:
   fmt.Printf("Received AudioEvent: %d bytes\n", len(v.Value.AudioChunk))
   audioFile.Write(v.Value.AudioChunk)
  case *types.StartSpeechSynthesisStreamEventStreamMemberStreamClosedEvent:
   fmt.Printf("Stream closed. Characters synthesized: %d\n",
 v.Value.RequestCharacters)
  }
 }

 if err := stream.Err(); err != nil {
  fmt.Fprintf(os.Stderr, "Stream error: %v\n", err)
  os.Exit(1)
 }
}
```

Rust

```
use aws_sdk_polly::types::{
    CloseStreamEvent, Engine, OutputFormat,
    StartSpeechSynthesisStreamActionStream,
    StartSpeechSynthesisStreamEventStream, TextEvent, TextType, VoiceId,
};
use aws_sdk_polly::Client;
use std::fs::File;
use std::io::Write;
use tokio::time::{sleep, Duration};

#[tokio::main]
async fn main() {
    let config = aws_config::defaults(aws_config::BehaviorVersion::latest())
        .region("us-east-1")
        .load()
        .await;

    let client = Client::new(&config);

    let text_chunks = vec![
        "The weather forecast for today shows clear skies ",
```

```
        "with temperatures reaching twenty five degrees. ",
        "Tomorrow we expect some cloud cover in the morning ",
        "but it should clear up by the afternoon. ",
        "The rest of the week looks mostly sunny ",
        "with a slight chance of rain on Friday. ",
        "Overall a great week to spend time outdoors.",
    ];

    let input_stream = async_stream::stream! {
        for chunk in &text_chunks {
            println!("Sending TextEvent: {}", chunk.trim());
            yield Ok(StartSpeechSynthesisStreamActionStream::TextEvent(

TextEvent::builder().text(*chunk).text_type(TextType::Text).build().unwrap(),
            ));
            // Simulate delay between chunks (e.g. waiting for LLM tokens)
            sleep(Duration::from_millis(300)).await;
        }
        yield Ok(StartSpeechSynthesisStreamActionStream::CloseStreamEvent(
            CloseStreamEvent::builder().build(),
        ));
    };

    let mut output = client
        .start_speech_synthesis_stream()
        .engine(Engine::Generative)
        .voice_id(VoiceId::Tiffany)
        .output_format(OutputFormat::Mp3)
        .sample_rate("24000")
        .action_stream(input_stream.into())
        .send()
        .await
        .expect("Failed to start stream");

    let mut audio_file = File::create("output.mp3").expect("Failed to create output
file");

    while let Ok(Some(event)) = output.event_stream.recv().await {
        match event {
            StartSpeechSynthesisStreamEventStream::AudioEvent(audio_event) => {
                if let Some(chunk) = audio_event.audio_chunk() {
                    let bytes = chunk.as_ref();
                    println!("Received AudioEvent: {} bytes", bytes.len());
                    audio_file.write_all(bytes).unwrap();
```

```
                }
            }
            StartSpeechSynthesisStreamEventStream::StreamClosedEvent(closed_event)
 => {
                println!(
                    "Stream closed. Characters synthesized: {}",
                    closed_event.request_characters()
                );
            }
            _ => {}
        }
    }
}
```

Stimmen in Amazon Polly

Amazon Polly bietet Dutzende lebensechter Stimmen und unterstützt eine Vielzahl von Sprachen. Jede Stimme wird mithilfe von Muttersprachlern erstellt, sodass es je nach Stimme Variationen geben kann, auch in derselben Sprache. Sie können das auch verwenden AWS-Managementkonsole , um jede Stimme mit Text Ihrer Wahl zu testen. In den meisten Sprachen wird es mindestens eine männliche und eine weibliche Stimme geben, oft auch mehr als eine von beiden. Einige Sprachen haben nur eine einzige Stimme.

Das Stimmenverzeichnis und die Anzahl der enthaltenen Sprachen werden laufend aktualisiert, um weitere Optionen einzubeziehen. Wenn Sie eine neue Sprache oder Stimme vorschlagen möchten, geben Sie uns auf dieser Seite Feedback. Leider sind wir nicht in der Lage, Pläne für bestimmte neue Sprachen zu kommentieren, bevor sie veröffentlicht werden.

> **Note**
>
> Beispiele für Amazon Polly Polly-Stimmen in Ihrem Browser finden Sie in der Amazon Polly Polly-Produktübersicht.

Themen

Verfügbare Stimmen

Amazon Polly bietet eine Vielzahl lebensechter Stimmen in mehreren Sprachen, um Sprache aus Text zu synthetisieren. Die folgende Tabelle zeigt alle Stimmen, die Amazon Polly anbietet.

	Sprache und Sprachvarianten	Sprachcode	Name/ID	Gender	Generative Stimme	Stimme in Langform	Neuronale Stimme	Standardstimme
1	Arabisch	arb	Zeina	Weiblich	Nein	Nein	Nein	Ja
2	Arabisch (Golf)	ar-AE	Hala*	Weiblich	Nein	Nein	Ja	Nein
			Zayd*	Männlich	Nein	Nein	Ja	Nein
3	Niederländisch (Belgisch)	nl-BE	Lisa	Weiblich	Ja	Nein	Ja	Nein
4	katalanisch	ca-ES	Arlet	Weiblich	Nein	Nein	Ja	Nein
5	Tschechisch	cs-CZ	Jitka	Weiblich	Nein	Nein	Ja	Nein
6	Chinesisch (Kantonesisch)	Yue-CN	Huujin	Weiblich	Nein	Nein	Ja	Nein
7	Chinesisch (Mandarin)	cmn-CN	Zhiyu	Weiblich	Nein	Nein	Ja	Ja
8	Dänisch	da-DK	Naja	Weiblich	Nein	Nein	Nein	Ja
			Mads	Männlich	Nein	Nein	Nein	Ja
			Sofie	Weiblich	Nein	Nein	Ja	Nein
9	Niederländisch	nl-NL	Laura	Weiblich	Ja	Nein	Ja	Nein

	Sprache und Sprachvarianten	Sprachcode	Name/ID	Gender	Generative Stimme	Stimme in Langform	Neuronale Stimme	Standardstimme
			Lotte	Weiblich	Nein	Nein	Nein	Ja
			Ruben	Männlich	Nein	Nein	Nein	Ja
10	Englisch (australisch)	en-AU	Nicole	Weiblich	Nein	Nein	Nein	Ja
			Olivia	Weiblich	Ja	Nein	Ja	Nein
			Russell	Männlich	Nein	Nein	Nein	Ja
11	Englisch (britisch)	en-GB	Amy**	Weiblich	Ja	Nein	Ja	Ja
			Emma	Weiblich	Nein	Nein	Ja	Ja
			Brian	Männlich	Ja	Nein	Ja	Ja
			Artur	Männlich	Nein	Nein	Ja	Nein
12	Englisch (indisch)	en-IN	Aditi*	Weiblich	Nein	Nein	Nein	Ja
			Raveena	Weiblich	Nein	Nein	Nein	Ja
			Kajal*	Weiblich	Ja	Nein	Ja	Nein
13	Englisch (Irland)	en-IE	Niamh	Weiblich	Ja	Nein	Ja	Nein
14	Englisch (Neuseeland)	en-NZ	Aria	Weiblich	Ja	Nein	Ja	Nein
15	Englisch (Singapurisch)	en-SG	Jasmin	Weiblich	Ja	Nein	Ja	Nein

	Sprache und Sprachvarianten	Sprachcode	Name/ID	Gender	Generative Stimme	Stimme in Langform	Neuronale Stimme	Standardstimme
16	Englisch (Südafrikanisch)	en-ZA	Ayanda	Weiblich	Ja	Nein	Ja	Nein
17	Englisch (amerikanisch)	en-US	Danielle	Weiblich	Ja	Ja	Ja	Nein
			Gregor	Männlich	Nein	Ja	Ja	Nein
			Ivy	Weiblich (Kind)	Nein	Nein	Ja	Ja
			Joanna**		Ja	Nein	Ja	Ja
			Kendra	Weiblich	Nein	Nein	Ja	Ja
			Kimberly	Weiblich	Nein	Nein	Ja	Ja
			Salli	Weiblich	Ja	Nein	Ja	Ja
			Joey	Weiblich	Nein	Nein	Ja	Ja
			Justin	Männlich	Nein	Nein	Ja	Nein
			Kevin	Männlich (Kind)	Nein	Nein	Ja	Ja
			Matthew**	Männlich	Ja	Nein	Ja	Nein
			Ruth	Weiblich (Kind)	Ja	Ja	Ja	Nein
			Stephen	Männlich	Ja	Nein	Ja	Nein
			Tiffany	Weiblich	Ja	Nein	Nein	Nein
			Patrick	Männlich	Nein	Ja	Nein	Nein
				Weiblich				
				Männlich				

	Sprache und Sprachvarianten	Sprachcode	Name/ID	Gender	Generative Stimme	Stimme in Langform	Neuronale Stimme	Standardstimme
18	Englisch (walisisch)	en-GB-WLS	Geraint	Männlich	Nein	Nein	Nein	Ja
19	Finnisch	fi-FI	Suvi	Weiblich	Nein	Nein	Ja	Nein
20	Französisch	fr-FR	Bernstein	Weiblich	Ja	Nein	Nein	Nein
			Céline/Celine	Weiblich	Ja	Nein	Nein	Ja
			Florian	Männlich	Ja	Nein	Nein	Nein
			Léa	Weiblich	Ja	Nein	Ja	Ja
			Mathieu	Männlich	Nein	Nein	Nein	Ja
			Remi	Männlich	Ja	Nein	Ja	Nein
21	Französisch (Belgisch)	fr-BE	Isabelle	Weiblich	Ja	Nein	Ja	Nein
22	Kanadisches Französisch	fr-CA	Chantal	Weiblich	Nein	Nein	Nein	Ja
			Gabrielle	Weiblich	Ja	Nein	Ja	Nein
			Liam	Männlich	Ja	Nein	Ja	Nein

	Sprache und Sprachvarianten	Sprachcode	Name/ID	Gender	Generative Stimme	Stimme in Langform	Neuronale Stimme	Standardstimme
23	Deutsch	de-DE	Marlene	Weiblich	Nein	Nein	Nein	Ja
			Vicki	Weiblich	Ja	Nein	Ja	Ja
			Hans	Männlich	Nein	Nein	Nein	Ja
			Daniel	Männlich	Ja	Nein	Ja	Nein
			Lennart	Männlich	Ja	Nein	Nein	Nein
24	Deutsch (Österreichisch)	de-AT	Hannah	Weiblich	Ja	Nein	Ja	Nein
25	Deutsch (Schweizerisch)	de-CH	Sabrina	Weiblich	Ja	Nein	Ja	Nein
26	Hindi	hi-IN	Aditi*	Weiblich	Nein	Nein	Nein	Ja
			Kajal*	Weiblich	Nein	Nein	Ja	Nein
27	Isländisch	is-IS	Dóra/Dora	Weiblich	Nein	Nein	Nein	Ja
			Karl	Männlich	Nein	Nein	Nein	Ja

	Sprache und Sprachvarianten	Sprachcode	Name/ID	Gender	Generative Stimme	Stimme in Langform	Neuronale Stimme	Standardstimme
28	Italienisch	it-IT	Beatrice	Weiblich	Ja	Nein	Nein	Nein
			Carla	Weiblich	Nein	Nein	Nein	Ja
			Bianca	Weiblich	Ja	Nein	Ja	Ja
			Lorenzo	Männlich	Ja	Nein	Nein	Nein
			Giorgio	Männlich	Nein	Nein	Nein	Ja
			Adriano	Männlich	Nein	Nein	Ja	Nein
29	Japanisch	ja-JP	Mizuki	Weiblich	Nein	Nein	Nein	Ja
			Takumi	Männlich	Nein	Nein	Ja	Ja
			Kazuha	Weiblich	Nein	Nein	Ja	Nein
			Tomoko	Weiblich	Nein	Nein	Ja	Nein
30	Koreanisch	ko-KR	Seoyeon	Weiblich	Ja	Nein	Ja	Ja
			Jihye	Weiblich	Nein	Nein	Ja	Nein
31	Norwegisch	nb-NO	Liv	Weiblich	Nein	Nein	Nein	Ja
			Ida	Weiblich	Nein	Nein	Ja	Nein
32	Polnisch	pl-PL	Ewa	Weiblich	Ja	Nein	Nein	Ja
			Maja	Weiblich	Nein	Nein	Nein	Ja
			Jacek	Männlich	Nein	Nein	Nein	Ja
			. Jan.	Männlich	Nein	Nein	Nein	Ja
			Ola	Weiblich	Ja	Nein	Ja	Nein

	Sprache und Sprachvarianten	Sprachcode	Name/ID	Gender	Generative Stimme	Stimme in Langform	Neuronale Stimme	Standardstimme
33	Portugiesisch (brasilianisch)	pt-BR	Camila	Weiblich	Ja	Nein	Ja	Ja
			Vitória/Vitoria	Weiblich	Nein	Nein	Ja	Ja
			Ricardo	Männlich	Nein	Nein	Nein	Ja
			Thiago	Männlich	Nein	Nein	Ja	Nein
34	Portugiesisch (europäisch)	pt-PT	Inês/Ines	Weiblich	Nein	Nein	Ja	Ja
			Cristiano	Männlich	Nein	Nein	Nein	Ja
35	Rumänisc	ro-RO	Carmen	Weiblich	Nein	Nein	Nein	Ja
36	Russisch	ru-RU	Tatyana	Weiblich	Nein	Nein	Nein	Ja
			Maxim	Männlich	Nein	Nein	Nein	Ja
37	Spanisch (Spanien)	es-ES	Conchita	Weiblich	Nein	Nein	Nein	Ja
			Lucia	Weiblich	Ja	Nein	Ja	Ja
			Alba	Weiblich	Nein	Ja	Nein	Nein
			Enrique	Männlich	Nein	Nein	Nein	Ja
			Sergio	Männlich	Ja	Nein	Ja	Nein
			Raul	Männlich	Nein	Ja	Nein	Nein
38	Spanisch (Mexikanisch)	es-MX	Mia	Weiblich	Ja	Nein	Ja	Ja
			Andrés	Männlich	Ja	Nein	Ja	Nein

	Sprache und Sprachvarianten	Sprachcode	Name/ID	Gender	Generative Stimme	Stimme in Langform	Neuronale Stimme	Standardstimme
39	Spanisch (USA)	es-US	Lupe**	Weiblich	Ja	Nein	Ja	Ja
			Penélope/Penelope	Weiblich	Nein	Nein	Nein	Ja
			Miguel	Männlich	Nein	Nein	Nein	Ja
			Pedro	Männlich	Ja	Nein	Ja	Nein
40	Schwedisch	sv-SE	Astrid	Weiblich	Nein	Nein	Nein	Ja
			Elin	Weiblich	Nein	Nein	Ja	Nein
41	Türkisch	tr-TR	Filiz	Weiblich	Nein	Nein	Nein	Ja
			Burcu	Weiblich	Nein	Nein	Ja	Nein
42	Walisisch	cy-GB	Gwyneth	Weiblich	Nein	Nein	Nein	Ja

* Diese Stimme ist zweisprachig. Weitere Informationen finden Sie unter Zweisprachige Stimmen.

** Diese Stimmen können mit den Sprechstilen von Newscaster verwendet werden, wenn sie mit dem Neural-Format verwendet werden. Weitere Informationen finden Sie unter Die Stimme des Nachrichtensprechers anwenden.

Jede Amazon Polly Voice Engine verfügt über einzigartige Funktionen. Erfahren Sie mehr über die Funktionen und die regionale Verfügbarkeit der von Amazon Polly angebotenen Sprachmodule:

- Generative Stimmen
- Stimmen in Langform
- Neuronale Stimmen
- Standardstimmen

Markenstimmen

Zusätzlich zu den in der vorherigen Tabelle aufgeführten verfügbaren Stimmen können Sie Amazon Polly verwenden, um eine benutzerdefinierte Stimme für Ihre Markenpersönlichkeit zu erstellen. Mit einer Markenstimme können Sie Ihren Kunden einzigartige und exklusive Stimmen bieten. Weitere Informationen zu den Markenstimmen von Amazon Polly finden Sie unter Brand Voice.

Zweisprachige Stimmen

Amazon Polly bietet zwei Möglichkeiten, zweisprachige Stimmen zu erzeugen:

- Akzentuierte zweisprachige Stimmen
- Vollständige zweisprachige Stimmen

Akzentuierte zweisprachige Stimmen

Zweisprachige Stimmen mit Akzent können mit jeder Amazon Polly Polly-Stimme erstellt werden, jedoch nur, wenn SSML-Tags verwendet werden.

Normalerweise werden alle Wörter im Eingabetext in der Standardsprache mit der von Ihnen angegebenen Stimme gesprochen.

Wenn Sie beispielsweise die Stimme von Joanna verwenden (die US-Englisch spricht), spricht Amazon Polly Folgendes mit der Stimme von Joanna ohne französischen Akzent:

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

In diesem Fall werden die Worte Je ne parle pas français so gesprochen, als wären sie Englisch.

Wenn Sie jedoch die Joanna-Stimme mit <lang>dem Tag verwenden, spricht Amazon Polly den Satz in der Joanna-Stimme in Französisch mit amerikanischem Akzent:

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

Da Joanna keine französische Muttersprachlerin ist, basiert die Aussprache auf ihrer Muttersprache, also US-Englisch. Eine Person mit perfekter französischer Aussprache würde beispielsweise das Wort français mit einem uvularen Vibrant (/R/) sprechen. Joannas Stimme (US-Englisch) spricht dieses Phonem dagegen wie /r/.

Wenn Sie die Stimme von Giorgio, der Italienisch spricht, mit dem folgenden Text verwenden, spricht Amazon Polly den Satz in Giorgios Stimme mit italienischer Aussprache:

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

Vollständige zweisprachige Stimmen

Eine vollständig zweisprachige Stimme wie Aditi oder Kajal (indisches Englisch und Hindi) kann zwei Sprachen fließend sprechen. Dadurch haben Sie die Möglichkeit, für Wörter und Sätze aus beiden Sprachen in einem einzigen Text dieselbe Stimme zu verwenden.

Derzeit sind Aditi, Kajal, Hala und Zayd die einzigen vollständig zweisprachigen Stimmen, die verfügbar sind.

Verwendung einer zweisprachigen Stimme (Beispiel: Aditi)

Aditi spricht sowohl indisches Englisch (en-IN) als auch Hindi (hi-IN) fließend. Sie können Sprachinhalte sowohl auf Englisch als auch auf Hindi generieren, und die Stimme kann sogar innerhalb desselben Satzes zwischen den beiden Sprachen wechseln.

Hindi kann in zwei verschiedenen Formen verwendet werden:

- Dewanagari: "उसेन कहाँ, खेल तोह अब शूर होगा"
- Romanagari (mithilfe des lateinischen Alphabets): "Usne kahan, khel toh ab shuru hoga"

Außerdem ist eine Mischung aus Englisch und Hindi in einer oder beiden Formen innerhalb eines einzigen Satzes möglich:

- Dewanagari + Englisch: "This is the song कभी कभी अदति"
- Romanagari + Englisch: "This is the song from the movie Jaane Tu Ya Jaane Na."
- Dewanagari + Romanagari + Englisch: "This is the song कभी कभी अदति from the movie Jaane Tu Ya Jaane Na."

Da Aditi eine zweisprachige Stimme ist, wird Text in all diesen Fällen korrekt gelesen, da Amazon Polly zwischen den Sprachen und Schriften unterscheiden kann.

Amazon Polly unterstützt auch Zahlen, Daten, Uhrzeiten und Währungserweiterungen sowohl in Englisch (arabische Ziffern) als auch in Hindi (Devanagari-Ziffern). Arabische Ziffern werden standardmäßig in indischem Englisch gelesen. Damit Amazon Polly sie auf Hindi lesen kann, müssen Sie den `hi-IN` Sprachcode-Parameter verwenden.

Die Stimme des Nachrichtensprechers anwenden

Je nach Kontext verwenden die Menschen unterschiedliche Sprechstile. Gelegene Unterhaltung klingt beispielsweise sehr anders als eine TV- oder Radio-Newsscast. Aufgrund der Art und Weise, wie Standardstimmen erzeugt werden, können sie keine unterschiedlichen Sprechstile erzeugen. Neuronale Stimmen können das jedoch. Sie können für einen bestimmten Sprechstil trainiert werden, wobei die Variationen und der Schwerpunkt auf bestimmten Wortarten liegen, die diesem Stil innewohnen.

Zusätzlich zu den standardmäßigen neuronalen Stimmen bietet Amazon Polly einen Sprechstil für Nachrichtensprecher, der das neuronale System verwendet, um Sprache im Stil eines Fernseh- oder Radio-Nachrichtensenders zu erzeugen. Der Newscaster-Stil ist mit den Stimmen von Matthew und Joanna in US-Englisch (en-US), der Lupe-Stimme in US-Spanisch (es-US) und der Amy-Stimme in britischem Englisch (en-GB) verfügbar.

Um den Newscaster-Stil zu verwenden, wählen Sie zuerst das neuronale Modul aus und verwenden dann die Syntax, die in den folgenden Schritten beschrieben wird, in Ihrem Eingabetext.

> **Note**
>
> - Um einen beliebigen neuronalen Sprachstil verwenden zu können, müssen Sie eine der Regionen verwenden, die neuronale Stimmen unterstützen. AWS Diese Option ist nicht in allen Regionen verfügbar. Weitere Informationen finden Sie unter Kompatibilität mit Funktionen und Regionen.

Console

Um den Newscaster-Stil anzuwenden

1. Öffnen Sie die Amazon Polly Polly-Konsole unter https://console.aws.amazon.com/polly/.

2. Stellen Sie sicher, dass Sie eine AWS Region verwenden, in der neuronale Stimmen unterstützt werden.
3. Wählen Sie auf der Text-to-Speech Seite für Engine die Option Neural aus.
4. Wählen Sie die Sprache und Stimme aus, die Sie verwenden möchten. Nur Matthew und Joanna für US-Englisch (en-US), Lupe für US-Spanisch (es-US) und Amy für britisches Englisch (en-GB) sind in der Nachrichtensprecherstimme verfügbar.
5. Schalten Sie SSML ein.
6. Fügen Sie Ihrer text-to-speech Anfrage mithilfe der SSML-Syntax im Newscaster-Stil einen Eingabetext hinzu.

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

   Beispielsweise können Sie das Newscaster-Tag wie folgt verwenden:

   ```
   <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. Wählen Sie „Zuhören“.

AWS CLI

Um den Newscaster-Stil anzuwenden

1. Fügen Sie in Ihrer API-Anforderung den Engine-Parameter mit dem Wert `neural` ein:

   ```
   --engine neural
   ```

2. Fügen Sie Ihrer API-Anforderung mithilfe der SSML-Syntax im Newscaster-Stil Eingabetext hinzu.

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

Beispielsweise können Sie das Newscaster-Tag wie folgt verwenden:

```
<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>
```

Weitere Informationen zu SSML finden Sie unter Unterstützte SSML-Tags.

Stimmen hören

Sobald Sie Amazon Polly eingerichtet haben, können Sie Stimmen mit benutzerdefiniertem Text auf der Konsole testen.

Um Amazon Polly-Stimmen auf der Konsole zu hören

1. Melden Sie sich bei der an AWS-Managementkonsole und öffnen Sie die Amazon Polly Polly-Konsole unter https://console.aws.amazon.com/polly/.
2. Wählen Sie die Registerkarte Text-to-Speech.
3. Wählen Sie für Engine Generative, Long Form, Neural oder Standard.
4. Wählen Sie eine Sprache und eine Region aus. Wählen Sie dann eine Stimme.

5. Geben Sie Text ein, den die Stimme sprechen soll, oder verwenden Sie die Standardphrase und wählen Sie dann „Zuhören“.

Timing einer Sprachgeschwindigkeit

Aufgrund der natürlichen Variation zwischen den Stimmen spricht jede verfügbare Stimme mit leicht unterschiedlichen Geschwindigkeiten. Zum Beispiel sind Ivy und Joanna mit US-englischen Stimmen etwas schneller als Matthew und erheblich schneller als Joey. Da es so viele Unterschiede zwischen den Stimmen gibt, ist für Amazon Polly Polly-Stimmen keine Standardgeschwindigkeit (Wörter pro Minute) verfügbar. Mithilfe von Sprachmarken können Sie jedoch herausfinden, wie lange es dauert, bis Ihre Stimme den ausgewählten Text sagt.

Um die Länge einer gesprochenen Textpassage zu messen

1. Öffne das AWS CLI.
2. Führen Sie den folgenden Code aus und füllen Sie ihn nach Bedarf aus.

```
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. Öffnen Sie `LengthOfText.txt`.

Wenn der Text „Mary hatte ein kleines Lamm“ lauten würde, wären die letzten Zeilen, die Amazon Polly zurückgibt, wie folgt:

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

Das letzte Mundbild, d. h. der Klang der letzten Buchstaben in „lamb“, beginnt 1082 Millisekunden nach dem Beginn der Sprachausgabe. Dies ist zwar nicht genau die Länge des Audiomaterials, kommt dem jedoch ziemlich nahe und dient als Grundlage für den Vergleich zwischen Stimmen.

Sprachgeschwindigkeit ändern

Bei bestimmten Anwendungen stellen Sie möglicherweise fest, dass Ihnen die Stimme langsamer oder schneller besser gefallen würde. Wenn die Geschwindigkeit der Stimme ein Problem darstellt, bietet Amazon Polly die Möglichkeit, dies mithilfe von SSML-Tags zu ändern. Wenn Ihre Organisation beispielsweise eine Anwendung entwickelt hat, mit der Bücher für ein Publikum mit Migrationshintergrund vorgelesen werden, möchten Sie möglicherweise die Sprachgeschwindigkeit variieren. Ihr Publikum spricht zwar Englisch, aber seine Sprachkenntnisse sind begrenzt. **<**prosody**>**Amazon Polly hilft Ihnen dabei, die Sprechgeschwindigkeit mithilfe des SSML-Tags zu verlangsamen.

Sie können einen Prozentsatz verwenden:

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

Oder eine voreingestellte Geschwindigkeit:

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

Bei der Verwendung von SSML mit Amazon Polly stehen Ihnen zwei Geschwindigkeitsoptionen zur Verfügung:

- Voreingestellte Geschwindigkeiten:`x-slow`,`slow`, `mediumfast`, und. `x-fast` In diesen Fällen handelt es sich bei der Geschwindigkeit der jeweiligen Option je nach bevorzugter Stimme um einen geschätzten Wert. Die `medium`-Option ist die normale Sprechgeschwindigkeit.
- n% der Sprachgeschwindigkeit: Es kann ein beliebiger Prozentsatz der Sprachgeschwindigkeit zwischen 20 und 200% verwendet werden. In diesen Fällen können Sie genau die gewünschte Geschwindigkeit wählen. Die tatsächliche Geschwindigkeit der Stimme ist jedoch ein ungefährer Wert, abhängig von der ausgewählten Stimme. 100% wird als normale Geschwindigkeit der Stimme angesehen.

> **Note**
>
> Testen Sie die von Ihnen gewählte Stimme mit verschiedenen Geschwindigkeiten. Die Geschwindigkeit jeder Option ist eine ungefähre Angabe und hängt von der ausgewählten Stimme ab.

Weitere Informationen zur Verwendung des `prosody` Tags finden Sie unterSteuerung von Lautstärke, Sprechgeschwindigkeit und Tonhöhe.

Sprachen bei Amazon Polly

Die folgenden Sprachen werden von Amazon Polly unterstützt und können zur Sprachsynthese verwendet werden. Jede Sprache hat einen eindeutigen Sprachcode. Diese Sprachcodes sind W3C-Tags zur Sprachenidentifikation (*ISO 639-3*für den Sprachnamen und *ISO 3166* für den Ländercode).

Wählen Sie eine Sprache aus der folgenden Tabelle aus, um weitere Informationen zu den von Amazon Polly bereitgestellten Phonemen und Visemen zu erhalten.

Sprach	Sprache
Arabisc	arb
Arabisc (Golf)	Ar-ae
katalan ch	CA-es
Chines h (Kantor isch)	Yue-CN
Chines h (Manda)	cmn-CN
Tschec ch	cs-CZ
Dänisc	da-DK
Niederl disch	Nl-BE

Sprach	Sprache
(Belgis)	
Niederl disch	nl-NL
Englisc (austra sch)	en-AU
Englisc (britisch)	en-GB
Englisc (indisch	en-IN
Englisc (Neuse nd)	en-NZ
Englisc (Singap isch)	en-SG
Englisc (Südafr anisch)	en-ZA
Englisc (amerik isch)	en-US
Englisc (walisis h)	en-GB-WLS

Sprach	Sprache
Finnisc	Wi-Fi
Französ ch	fr-FR
Französ ch (Belgis)	fr-BE
Kanadi es Französ ch	fr-CA
Hindi	hi-IN
Deutsc	de-DE
Deutsc (Österr chisch)	de-AT
Deutsc (Schwe r Standa	de-CH
Isländis h	is-IS
Italienis ch	it-IT
Japanis	ja-JP

Sprach	Sprache
Koreanisch	ko-KR
Norwegisch	nb-NO
Polnisch	pl-PL
Portugiesisch (brasilianisch)	pt-BR
Portugiesisch (europäisch)	pt-PT
Rumänisch	ro-RO
Russisch	ru-RU
Spanisch (Spanien)	es-ES
Spanisch (Mexikanisch)	es-MX
Spanisch (USA)	es-US
Schwedisch	sv-SE

Sprach	Sprach e
Türkisc	tr-TR
Walisis	cy-GB

Arabisch (arb)

In der folgenden Tabelle sind die Phoneme des Internationalen Phonetischen Alphabets (IPA), die X-SAMPA-Symbole (Extended Speech Assessment Methods Phonetic Alphabet) und die entsprechenden Viseme für die arabische Stimme von Zeina aufgeführt, die von Amazon Polly unterstützt wird.

Tabelle der Phoneme/Viseme

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
Konsonanten				
ʔ	?	Glottallaut	أنا	
ʕ	?\	stimmhafter pharyngaler Frikativ	عُمَر	k
b	b	stimmhafter bilabialer Verschlus slaut	بَلَد	p
d	d	stimmhaft er alveolarer Verschlusslaut	داري	t
dˤ	d_?\	emphatisc her stimmhaft	ضَوء	t

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
		er alveolarer Verschlusslaut		
ʧ	dZ	stimmhafte postalveolare Affrikate	لِيمَجَ	S
ð	D	stimmhafter dentaler Frikativ	ذلِكَ	T
ðˤ	D_?\	emphatischer stimmhafter dentaler Frikativ	مالَظَ	T
f	f	stimmloser labiodentaler Frikativ	لصَفَ	f
g	g	stimmhafter velarer Verschlusslaut	إرتلجنإ	k
ɣ	G	stimmhafter velarer Frikativ	برَغَ	k
h	h	stimmloser glottaler Frikativ	اذه	k
j	j	palataler Approximant	يِشمَيَ	i
k	k	stimmloser velarer Verschlusslaut	بلَكَ	k
l	l	alveolarer lateraler Approximant	ىقال	t

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
lˠ	l_G	emphatischer alveolarer lateraler Approximant	هللادبع	t
m	m	bilabialer Nasal	اذام	p
n	n	alveolarer Nasal	رون	t
p	p	stimmloser bilabialer Verschlus slaut	سبَحَ	p
q	q	stimmloser uvularer Verschlus slaut	بيرَقَ	k
r	r	alveolarer Vibrant	لمَرَ	r
s	s	stimmlose r alveolarer Reibelaut	لاؤُس	s
sˤ	s_?\	emphatisc her stimmlose r alveolarer Reibelaut	بحاص	s
ʃ	S	stimmloser postalveolarer Reibelaut	رُكشُ	S
t	t	stimmlose r alveolarer Verschlusslaut	رمَتَ	t

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
tˤ	t_?\	emphatischer stimmloser alveolarer Verschlusslaut	بالطا	t
θ	T	stimmloser dentaler Reibelaut	ثالَثَا	T
V	V	stimmhafter labiodentaler Reibelaut	فيمانيي	f
w	w	labiovelarer Approximant	دلَوَ	u
x	x	stimmloser velarer Reibelaut	فْوَخَ	k
ħ	X\	stimmloser pharyngaler Reibelaut	لَوْلَحَ	k
z	z	stimmhafter alveolarer Reibelaut	رورُهُز	S
Vokale				
a	a	ungerundeter offener vorderer Vokal	دَرَبَ	a
a:	a:	langer ungerundeter offener vorderer Vokal	دار	a

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ɑˤ	A_?\	emphatischer ungerundeter offener hinterer Vokal	طَبل	a
ɑˤː	A_?\:	emphatischer langer ungerunde ter offener hinterer Vokal	ظالِم	a
u	u	gerundeter geschlossener hinterer Vokal	شُرب	u
uː	u:	langer gerundete r geschlossener hinterer Vokal	سور	u
uˤ	u_?\	emphatischer gerundeter geschlossener hinterer Vokal	بُدّ	u
uˤː	u_?\:	emphatischer langer gerundete r geschlossener hinterer Vokal	طول	u
i	i	ungerundeter geschlossener vorderer Vokal	بِنت	i
iː	i:	langer ungerunde ter geschlossener vorderer Vokal	حَزين	i

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
iˤ	i_?\	emphatischer ungerundeter geschlossener vorderer Vokal	ضدّ	i
iˤː	i_?\:	emphatischer langer ungerunde ter geschlossener vorderer Vokal	ياضام	i
e	e	ungerundeter halbgeschlossener vorderer Vokal	تكارم	e
eː	e:	langer ungerunde ter halbgesch lossener vorderer Vokal	ليدوم	e
ɔ	O	gerundeter halboffener Hinterzungenvokal	يجولونكت	O
ɔː	O:	langer gerundeter halboffener hinterer Vokal	نويزفيلت	O

Arabisch (Golf) (ar-AE)

In der folgenden Tabelle sind die Phoneme des Internationalen Phonetischen Alphabets (IPA), die X-SAMPA-Symbole (Extended Speech Assessment Methods Phonetic Alphabet) und die entsprechenden Viseme für die arabische Stimme von Hala aufgeführt, die von Amazon Polly unterstützt wird.

Tabelle der Phoneme/Viseme

IPA	X-SAMPA	Beschreibung	Beispiel	Aussprache	Viseme
Konsonanten					
b	b	stimmhafter bilabialer Verschlusslaut	دلب	/"b a l a d/	b
d	d	stimmhafter alveolarer Verschlusslaut	رد	/"r a d d/	d
dˤ	d_?\	pharyngealisiertes stimmiges alveoläres Plosivum	ضوء	/"d_? \ a w? /	D
f	f	stimmloser labiodentaler Frikativ	497	/"f I oder I n/	f
g	g	stimmhafter velarer Verschlusslaut	16	/"g a: l/	k
j	j	stimmhafter palatinaler Approximant	42.	/"j Ich bin. S i:/	i
k	k	stimmloser velarer Verschlusslaut	لماك	/"k a:. m i l/	k
l	l	stimmiger alveoläre	46	/"l e: l/	t

IPA	X-SAMPA	Beschreibung	Beispiel	Aussprache	Viseme
		r lateraler Approximant			
l	l_G	alveoläre r lateraler Approximant mit pharyngea lisiertem Stimmkopf	4.944	/? \ a b. “ von A_? \ l_G. L_G A_? \/	t
m	m	bilabialer Nasenstopp	مئة	/"m l j. j a/	p
n	n	alveolärer Nasenstopp	نور	/"u: r/	t
p	p	stimmlose r bilabialer Verschlusslaut	أوبرا	/"? O. p e. r a:/	p
q	q	stimmlose r uvularer Verschlusslaut	42.	/"q A_? \ s_? \ r/	k
r	r	alveolarer Vibrant	164	/"r a. m l l/	r
s	s	stimmlose r alveolarer Reibelaut	497	/„s Ich bin. Ich bin/	S
sˤ	s_?\	pharyngea lisierter, stimmlose r alveolärer Frikativ	497	/"s_? \ A_? :. X \ l b/	S

IPA	X-SAMPA	Beschreibung	Beispiel	Aussprache	Viseme
t	t	stimmlose r alveolarer Verschlusslaut	42.	/„t a m a r/	t
tˤ	t_?\	pharyngea lisierter, stimmlose r alveolärer Frikativ	497	/"t_? \ A_? :. l l b/	t
V	V	stimmhafter labiodentaler Reibelaut	نيمات يف	/v i:. t A. “ Ich bin: n/	f
w	w	stimmhafter labiovelarer Approximant	497	/"w a:. j l d/	u
x	x	stimmlose r velarer Reibelaut	4.918	/x a. “ r u: f/	k
z	z	stimmlose r velarer Reibelaut	زهور	/"z h u: r/	S
ð	D	stimmhaft interdentaler Frikativ	كلذ	/"D a:. l l k/	D
ðˤ	D_?\	pharyngea lisierter stimmiger interdentaler Frikativ	697	/D_? \ A_? \ . “ Ich bin: m/	D

IPA	X-SAMPA	Beschreibung	Beispiel	Aussprache	Viseme
ħ	X\	stimmloser pharyngaler Reibelaut	447	/? ein l. “ X\ i: n/	k
ŋ	N	Velarer Nasenstopp	غنوك غنوه	/h O N. “ k O N g/	k
ɣ	G	stimmhafter velarer Frikativ	446	/G l. “ Oder in:. b a/	k
ʃ	S	stimmloser postalveolarer Reibelaut	شمس	/"S a m s/	S
ʒ	Z	stimmhafter postalveolarer Frikativ	تيكاج	/Z a. “ k e: t/	S
ʔ	?	Glottallaut	42.	/m u. “ ? a s s a. s a/	
ʕ	?\	stimmhafter pharyngaler Frikativ	16.	/"? \ a: m m/	k
ʤ	dZ	stimmhafte postalveolare Affrikate	جامعة	/"dZ a: m.? \ a/	S
θ	T	stimmhaft interdentaler Frikativ	ثالثة	/T a. “ Ich bin:. T a/	T
ɦ	h	stimmhafter glottaler Frikativ	42.	/"h l a: l/	k

IPA	X-SAMPA	Beschreibung	Beispiel	Aussprache	Viseme
Vokale					
æ	a	mitteloffener vorderer, ungerundeter kurzer Vokal	497	/"s a. f a r/	a
ɑˤ	A_?\	pharyngea lisierter, offener Rücken, ungerundeter kurzer Vokal	497	/"s_? \ A_? \ l b/	a
æː	a:	mitteloffener vorderer ungerundeter langer Vokal	46	/"b a: b/	a
ɑˤː	A_?\:	pharyngea lisierter offener Rücken, ungerundeter langer Vokal	جضان	/"n A_? :. D_? \ i_? \ dZ/	a
a	A	offener zentraler ungerundeter kurzer Vokal	wlan	/"w A j. f A j/	a
i	i	eng anliegend er ungerunde ter kurzer Vokal (MSA)	إسحاق	/? es ist. “ X\ A_? \: q/	i

IPA	X-SAMPA	Beschreibung	Beispiel	Aussprache	Viseme
ɪ	I	lockerer vorderer ungerundeter kurzer Vokal	تنپ	/"b I n t/	i
iˤ	i_?\	pharyngea lisierter, enger vorderer, ungerundeter kurzer Vokal	497	/"t_? \ i_? \ f I l/	i
iː	iː	ungerunde ter langer Vokal vorne schließen	497	/ist ein. “ von i: l/	i
iˤː	i_? :	pharyngea lisierter, enger vorderer ungerundeter langer Vokal	497	/oder A_? \ . “ t_? \ i_? : b/	i
u	u	eng angespann ter hinterer abgerundeter kurzer Vokal (MSA)	4.918	/"m u x. t a. oder i? \/	u
ʊ	U	lockerer, geschloss ener, hinterer, abgerundeter kurzer Vokal	497	/oder U. “ Ist u: m/	u

IPA	X-SAMPA	Beschreibung	Beispiel	Aussprache	Viseme
uˤ	u_?\	pharyngea lisierter, enger Rücken, abgerundeter kurzer Vokal	4.918	/? \ u_? \ s_? \ . “ von uns: oder/	u
u:	u:	geschloss ener hinterer abgerundeter langer Vokal	46	/"t u:/	u
uˤː	u_?\:	pharyngea lisierter, enger Rücken, abgerundeter langer Vokal	497	/"s_? \ u_? \: r/	u
e	e	mittlerer vorderer ungerundeter kurzer Vokal	ـ	/"ist er n/	e
e:	e:	mittlerer vorderer ungerundeter langer Vokal	إيش	/"? e: S/	e
ɔ	O	offener, mittlerer Rücken, abgerundeter kurzer Vokal	دولار	/von O. “ l A oder/	O

IPA	X-SAMPA	Beschreibung	Beispiel	Aussprache	Viseme
ɔː	O:	offener, mittlerer Rücken, abgerundeter langer Vokal	لون	/"L O: n/	O

Katalanisch (ca-ES)

In der folgenden Tabelle sind die Phoneme des Internationalen Phonetischen Alphabets (IPA), die X-SAMPA-Symbole (Extended Speech Assessment Methods Phonetic Alphabet) und die entsprechenden Viseme für die katalanische Stimme von Arlet aufgeführt, die von Amazon Polly unterstützt wird.

Tabelle der Phoneme/Viseme

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
Konsonanten				
p	p	stimmloser bilabialer Verschlusslaut	Ploure	p
t	t	stimmlose r alveolarer Verschlusslaut	Nach Tarragona	t
k	k	stimmloser velarer Verschlusslaut	c Tom	k
b	b	stimmhafter bilabialer Verschlusslaut	von Data	p

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
d	d	stimmhaft er alveolarer Verschlusslaut	Ben de Roll	t
g	g	stimmhafter velarer Verschlusslaut	Ich bin rosig	k
m	m	stimmte bilabiale Nase	Ich bin Manera	p
n	n	stimmhaft alveolär nasal	Ein Hund im Auto	t
ɲ	J	stimmte palatinale Nase	ein Mädchen	J
ŋ	N	stimmhafte Velarnasale	Pi n Güí	k
ɬ	5	stimmiger velarisie rter alveolärer lateraler Approxima nt (dunkles L)	als El Bercoc	l
ʎ	L	stimmhafter palatinaler lateraler Approximant	Alles top	J
r	r	stimmiger Alveolart hriller	von rr a	r
ɾ	4	stimmiger Alveolarzapfen	pa oder a	t
f	f	stimmloser labiodentaler Frikativ	Pèm aus Asien	f

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
s	s	stimmlose r alveolarer Reibelaut	als Mac	S
z	z	stimmhaft er alveolarer Reibelaut	nenne z ja	S
ʃ	S	stimmloser postalveolarer Reibelaut	gui x	S
ʒ	Z	stimmhafter postalveolarer Frikativ	col·le g i	S
t͡ʃ	tS	stimmlose postalveolare Affrikate	co tax e	S
d͡ʒ	dZ	stimmhafte postalveolare Affrikate	spiele tj a	S
β	B	stimmlich er bilabialer Approximant	Um Bert zu sein	B
ð	D	stimmiger zahnärztlicher Approximant	Ein Bett und eine Puppe	T
j	j	stimmhafter palatinaler Approximant	nein, ich bin	i
ɣ	G	stimmiger kelarer Approximant	pega	k

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
V	V	stimmhafter labiodentaler Reibelaut	Ich bin von Gà	f
w	w	stimmlich labiovela r approximant	Rufen Sie uns an	u
x	x	stimmloser velarer Reibelaut	J Jiménez	k
ʝ	j\	stimmhafter palataler Frikativ	Von Meso	J
l	l	stimmiger alveolärer lateraler Approximant	Ich bin El Londra	t
θ	T	stimmloser dentaler Reibelaut	Gon z Valez	T
Vokale				
a	a	Vokal öffnen	casa	a
e	e	ungerundeter halbgeschlossener Vorderzungenvokal	Ich nenne dich	e
ɛ	E	ungerundeter halboffener Vorderzungenvokal	x e c	E
i	i	geschloss ener vorderer ungerundeter Vokal	v ist ein Betrug	i

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
o	o	gerundeter halbgeschlossener Hinterzungenvokal	gehe zu uns	o
ɔ	O	gerundeter halboffener Hinterzungenvokal	j oder c	O
u	u	geschlossener, runder Vokal auf der Rückseite	u n	u
ə	@	Schwa	casa	@
Zusätzliche Symbole				
ˈ	"	Hauptakzent	Alabama	
ˌ	%	Nebenakzent	Alabama	
.	.	Silbengrenze	A.la.ba.ma	

Chinesisch (Kantonesisch) (Yue-CN)

In der folgenden Tabelle sind die Phoneme Jyutping und IPA (International Phonetic Alphabet) für die kantonesische Stimme aufgeführt, die von Amazon Polly unterstützt werden. Jyutping ist ein Romanisierungssystem für Kantonesisch, das in der Wissenschaft und unter kantonesischsprachigen Menschen häufig verwendet wird. IPA und X-SAMPA werden nicht häufig verwendet, sind jedoch für die Unterstützung des Englischen verfügbar. Die IPA- und X-SAMPA-Symbole in der Tabelle dienen lediglich Referenzzwecken und sollten nicht für die Transkribierung des Chinesischen verwendet werden. Jyutping-Beispiele und die entsprechenden Viseme werden ebenfalls gezeigt.

Verwenden Sie das Tag, damit Amazon Polly die phonetische Aussprache mit Jyutping verwendet. `phoneme alphabet="x-amazon-jyutping"`

Die folgenden Beispiele zeigen dies für die verschiedenen Standards.

Jyutping:

```
<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 akzeptiert nur kantonesische Eingaben, die in UTF-8 codiert sind.

Tabelle der Phoneme/Viseme

Springen	IPA	X-SAMPA	Beschreibung	Beispiel für Jutping	Viseme
Konsonanten					
b	p	p	stimmloser bilabialer Verschlusslaut	, b aa1	p
c	ts	ts_h	aspirierte stimmlose alveolare Affrikate	, c aa1	S

Springen	IPA	X-SAMPA	Beschreibung	Beispiel für Jutping	Viseme
d	t	t	stimmloser alveolarer Verschlusslaut	, d aa2	t
f	f	f	stimmloser labiodentaler Frikativ	, f aa1	f
g	k	k	stimmloser velarer Verschlusslaut	g, aa1	k
gw	k	k_w	labialisiert, stimmlos, hellhörig, plosiv	, gw aa1	u
h	h	h	stimmloser glottaler Frikativ	h, aa1	k
k	k^h	k_h	aspirierter stimmloser velarer Plosiv	k, aa1	k
KW	k	k_wh	labialisiert, abgesaugt, stimmlos, velar, plosiv	, kw aa1	u
l	l	l	alveolarer lateraler Approximant	l, aa1	t
m	m	m	bilabialer Nasal	, m aa1	p
m	m	i=	silbischer bilabialer Nasal	, m 4	p
ng	ŋ	N	velarer Nasal	, ng aa4	k
ng	ŋ	N =	Silbe velar nasal	, ng 4	k
n	n	n	alveolarer Nasal	, n aa4	t
p	p^h	p_h	aspirierter stimmloser bilabialer Plosiv	, p aa1	p

Springen	IPA	X-SAMPA	Beschreibung	Beispiel für Jutping	Viseme
s	S	s	stimmloser alveolarer Reibelaut	, s aa1	S
t	t^h	t_h	aspirierter stimmloser alveolarer Plosiv	, t aa1	t
w	w	w	labiovelarer Approximant	, w aa1	u
y	j	j	palataler Approximant	, j aa5	i
z	ts	ts	stimmlose alveolare Affrikate	, z aa1	S
Vokale					
a	ɐ	6	fast offener Zentralvokal	, g, a t1	a
aa	ɑ	A	ungerundeter offener Hinterzungenvokal	, g aa 1	a
aai	i	Ai	Dipthong	g, ai 1	a
aau	u	Au	Dipthong	g, Tag 1	a
ai	i	6i	Dipthong	g, Mai 1	a
au	u	6u	Dipthong	, vor 1	a
e	ɛ	E	ungerundeter halboffener Vorderzungenvokal	, d e 1	E
ei	ei	ei	Dipthong	, Ei 1	e
eo	ɵ	8	gerundeter halbgeschlossener Zentralvokal	, c eo n1	o
eoi	y	8 Jahre	Diphthong	g, eoi 1	o

Springen	IPA	X-SAMPA	Beschreibung	Beispiel für Jutping	Viseme
eu	ɛu	Eu	Diphthong	in d, eu 6	E
i	i	i	ungerundeter geschlossener Vorderzungenvokal	, ist 1	i
i	I	I	ungerundeter zentralisierter fast geschlossener Vorderzungenvokal	, Gik 1	i
iu	iu	iu	Diphthong	, g iu 1	i
o	ɔ	O	gerundeter halboffener Hinterzungenvokal	, g oder 1	O
Eins	œ	9	gerundeter halboffener Vorderzungenvokal	, g oder 3	O
Öl	ɔi	Oi	Dipthong	, 1 g Öl	O
ou	ou	ou	Dipthong	, geh raus 1	o
u	u	u	gerundeter geschlossener Hinterzungenvokal	, g u 1	u
u	ʊ	U	gerundeter zentralisierter fast geschlossener Hinterzungenvokal	, g u k5	u
ui	ui	ui	Dipthong	, g, ui 6	u
yu	y	y	geschlossener gerundeter Vorderzungenvokal	Jahr, j yu 1	u
Tonmarkierungen und zusätzliche Symbole					
1			hohes Niveau	, si 1	

Springen	IPA	X-SAMPA	Beschreibung	Beispiel für Jutping	Viseme
2			mittel steigend	, si 2	
3			mittleres Niveau	, si 3	
4			sehr niedriges Niveau	, si 4	
5			niedrig steigend	, ist 5	
6			niedriges Niveau	, ist 6	
-	.	.	Silbengrenze	jyu5-jam1	

Chinesisch (Mandarin) (cmn-CN)

In der folgenden Tabelle sind die Phoneme Pinyin und IPA (International Phonetic Alphabet) für Mandarin-Chinesisch aufgeführt, die von Amazon Polly unterstützt werden. Pinyin ist der internationale Standard für die Standardtranskribierung des Chinesischen. IPA und X-SAMPA werden nicht häufig verwendet, sind jedoch für die Unterstützung des Englischen verfügbar. Die IPA- und X-SAMPA-Symbole in der Tabelle dienen lediglich Referenzzwecken und sollten nicht für die Transkribierung des Chinesischen verwendet werden. Pinyin-Beispiele und die entsprechenden Viseme werden ebenfalls angezeigt.

Verwenden Sie das Tag, damit Amazon Polly die phonetische Aussprache mit Pinyin verwendet. `phoneme alphabet="x-amazon-`*`phonetic standard used`*`"`

Die folgenden Beispiele zeigen dies für die verschiedenen Standards.

Pinyin:

```
<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 akzeptiert nur Eingaben in Mandarin-Chinesisch, die in UTF-8 codiert sind. Der Kodierungsstandard GB 18030 wird derzeit von Amazon Polly nicht unterstützt.

Tabelle der Phoneme/Viseme

Pinyin	IPA	X-SAMPA	Beschreibung	Pinyin-Beispiel	Viseme
Konsonanten					
f	f	f	stimmloser labiodentaler Frikativ	发, fa1	f
h	h	h	stimmloser glottaler Frikativ	和, he2	k
g	k	k	stimmloser velarer Verschlusslaut	古, gu3	k
k	kʰ	k_h	aspirierter stimmloser velarer Plosiv	苦, ku3	k
l	l	l	alveolarer lateraler Approximant	拉, la1	t
m	m	m	bilabialer Nasal	骂, ma4	p

Pinyin	IPA	X-SAMPA	Beschreibung	Pinyin-Beispiel	Viseme
n	n	n	alveolarer Nasal	那, na4	t
ng	ŋ	N	velarer Nasal	正, zheng4	k
b	p	p	stimmloser bilabialer Verschlusslaut	爸, ba4	p
p	pʰ	p_h	aspirierter stimmloser bilabialer Plosiv	怕, pa4	p
s	S	s	stimmloser alveolarer Reibelaut	四, si4	S
x	ɕ	S\	stimmloser alveolopalataler Frikativ	西, xi1	J
sh	ʂ	S`	stimmloser retroflexer Frikativ	是, shi4	S
d	t	t	stimmloser alveolarer Verschlusslaut	打, da3	t
t	tʰ	t_h	aspirierter stimmloser alveolarer Plosiv	他, ta1	t
zh	͡tʂ	t`s`	stimmlose retroflexe Affrikate	之, zhi1	S
ch	͡tʂʰ	t`s`_h	aspirierter stimmloser retroflexer Plosiv	吃, chi1	S
S	͡ts	ts	stimmlose alveolare Affrikate	字, zi4	S
j	͡tɕ	ts\	stimmlose alveolopalatale Affrikate	鸡, ji1	J

Pinyin	IPA	X-SAMPA	Beschreibung	Pinyin-Beispiel	Viseme
q	͡tɕʰ	ts_h	aspirierte stimmlose alveolopalatale Affrikate	七, qi1	J
c	͡tsʰ	ts_h	aspirierte stimmlose alveolare Affrikate	次, ci4	S
w	w	w	labiovelarer Approximant	我, wo3	u
r	ʐ	z`	stimmhafter retroflexer Frikativ	日, ri4	S
„ht“- und „r“-farbige Silben					
er	ɚ	@`	rhotisches mittlerer zentraler Vokal	二, er4	@
-r			r-farbige Silbe	馅儿, xianr4	@
Vokale					
e	ɤ	7	halbgeschlossener ungerundeter Hinterzungenvokal	恶, e4	e
e	ə	@	Schwa	恩, en1	@
a	a	a	ungerundeter offener Vor	安, an1	a
ai	aɪ	aI	Diphthong	爱, ai4	a
ao	aʊ	aU	Diphthong	奥, ao4	a
ei	eɪ	e	Diphthong	诶, ei4	e
e	ɛ	E	ungerundeter halboffener Vorderzungenvokal	姐, jie3	E

Pinyin	IPA	X-SAMPA	Beschreibung	Pinyin-Beispiel	Viseme
i	i	i	ungerundeter geschlossener Vorderzungenvokal	鸡, ji1	i
ou	oʊ	oU	Diphthong	欧, ou1	o
o	ɔ	O	gerundeter halboffener Hinterzungenvokal	哦, o4	o
u	u	u	gerundeter geschlossener Hinterzungenvokal	主, zhu3	u
yu	y	y	geschlossener gerundeter Vorderzungenvokal	于, yu2	u
Tonmarkierungen und zusätzliche Symbole					
1			hoher Ton	淤, yu1	
2			steigender Ton	鱼, yu2	
3			niedriger (fallend-steigender) Ton	语, yu3	
4			fallender Ton	育, yu4	
0			neutraler Ton	的, de0	
-	.	.	Silbengrenze	语音 yu3-yin1	

Tschechisch (cs-CZ)

In der folgenden Tabelle sind die Phoneme des Internationalen Phonetischen Alphabets (IPA), die X-SAMPA-Symbole (Extended Speech Assessment Methods Phonetic Alphabet) und die entsprechenden Viseme für die tschechische Stimme aufgeführt, die von Amazon Polly unterstützt wird.

Tabelle der Phoneme/Viseme

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
Konsonanten				
p	p	stimmloser bilabialer Verschlusslaut	p Ja	p
t	t	stimmloser alveolarer Verschlusslaut	t ok	t
c	c	stimmloser palataler Plosiv	t? uk	J
k	k	stimmloser velarer Verschlusslaut	k os	k
b	b	stimmhafter bilabialer Verschlusslaut	von Bez	p
d	d	stimmhafter alveolarer Verschlusslaut	d) ok	t
ɟ	J\	stimmhafter palataler Plosiv	d’as	J
g	g	stimmhafter velarer Verschlusslaut	Ich bin Mama	k
f	f	stimmloser labiodentaler Frikativ	von einem Film	f

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
V	V	stimmhafter labiodentaler Reibelaut	v ja	f
s	s	stimmlose r alveolarer Reibelaut	als Männer	S
z	z	stimmhaft er alveolarer Reibelaut	z el	S
ʃ	S	stimmloser postalveolarer Reibelaut	Ich bin el	S
ʒ	Z	stimmhafter postalveolarer Frikativ	ž en	S
x	x	stimmloser velarer Reibelaut	chat	k
ɦ	h	stimmhafter glottaler Frikativ	h uns	k
ʦ	ts	stimmlose alveolare Affrikate	c o	S
ʧ	tS	stimmlose postalveolare Affrikate	č in	S
ʣ	dz	stimmhafte alveolare Affrikate	špi c berský	S

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ʤ	dZ	stimmlose alveolare Affrikate	dž in	S
m	m	bilabialer Nasal	m oder	p
n	n	alveolarer Nasal	Auf iOS	t
ɲ	J	palataler Nasal	In Nader	J
ŋ	N	velarer Nasal	Bar auf Ka	k
r	r	stimmiger Alveolar-Triller	Oder ja	r
r	r_r	äußerte erhabenen alveolären Frikativtriller	Herr Bez	r
r	r_0_r	stimmloser erhöhter alveolärer Frikativtriller	Schlüssel ř	r
l	l	alveolarer lateraler Approximant	l ja	t
j	j	palataler Approximant	j en	i
w	w	labiovelärer Näherungswert	Ein Watson	u
Wr	r_=	alveolärer Triller mit silbischer Stimme	k oder k	r
l̩	l_=	silbischer alveolarer lateraler Approximant	v l na	t

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
Vokale				
a	a	ungerundeter offener Vor	Ich bin ein	a
aː	a:	langer ungerunde ter offener Vorderzungenvokal	Ich bin ein	a
ɛ	E	ungerundeter halboffener Vorderzungenvokal	Ich bin t	E
ɛː	E:	langer ungerunde ter halboffener Vorderzungenvokal	Ich bin es	E
ɪ	I	ungerundeter zentralisierter fast geschlossener Vorderzungenvokal	Ich bin es	i
iː	i:	langer ungerunde ter geschlossener Vorderzungenvokal	von í t	i
o	o	gerundeter halbgeschlossener Hinterzungenvokal	h oder l	o
oː	o:	langer gerundeter halbgeschlossener Hinterzungenvokal	g o l	o
u	u	gerundeter geschlossener Hinterzungenvokal	p u l	u

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
u:	u:	langer gerundeter geschlossener hinterer Vokal	p ů l	u
a x u	au	Diphthong	au zu	a
168.u	Eu	Diphthong	eu oder	E
o? u	ou	Diphthong	Bin ich okay	o
Zusätzliche Symbole				
ˈ	“	Hauptakzent		
.	.	Silbengrenze		

Dänisch (da-DK)

In der folgenden Tabelle sind die Phoneme des Internationalen Phonetischen Alphabets (IPA), die X-SAMPA-Symbole (Extended Speech Assessment Methods Phonetic Alphabet) und die entsprechenden Viseme für dänische Stimmen aufgeführt, die von Amazon Polly unterstützt werden.

Tabelle der Phoneme/Viseme

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
Konsonanten				
b	b	stimmhafter bilabialer Verschlusslaut	bat	p
d	d	stimmhafter alveolarer Verschlusslaut	da	t

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ð	D	stimmhafter dentaler Frikativ	mad, thriller	T
f	f	stimmloser labiodentaler Frikativ	fat	f
g	g	stimmhafter velarer Verschlusslaut	gat	k
h	h	stimmloser glottaler Frikativ	hat	k
j	j	palataler Approximant	jo	i
k	k	stimmloser velarer Verschlusslaut	kat	k
l	l	alveolarer lateraler Approximant	ladt	t
m	m	bilabialer Nasal	mat	p
n	n	alveolarer Nasal	nay	t
ŋ	N	velarer Nasal	lang	k
p	p	stimmloser bilabialer Verschlus slaut	pande	p
r	r	alveolarer Vibrant	thriller, story	r
ʁ	R	stimmhafter uvularer Frikativ	rat	k

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
s	s	stimmlose r alveolarer Reibelaut	sat	S
t	t	stimmlose r alveolarer Verschlusslaut	tal	t
V	V	stimmhafter labiodentaler Reibelaut	vat	f
w	w	labiovelarer Approximant	hav, weekend	u
Vokale				
ø	2	gerundeter halbgeschlossener Vorderzungenvokal	øst	o
ø:	2:	langer gerundeter halbgeschlossener Vorderzungenvokal	øse	o
ɐ	6	fast offener Zentralvokal	mor	a
œ	9	gerundeter halboffener Vorderzungenvokal	skøn, grønt	O
œ:	9:	langer gerundete r halboffener Vorderzungenvokal	høne, gøre	O
ə	@	Schwa	ane	@

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
æː	{:	langer ungerunde ter fast offener Vorderzungenvokal	male	a
a	a	ungerundeter offener Vor	man	a
æ	{	ungerundeter fast offener Vorderzun genvokal	adresse	a
ɑ	A	ungerundeter offener Hinterzun genvokal	lak, tak	a
ɑː	A:	langer ungerunde ter offener Hinterzungenvokal	rase	a
e	e	ungerundeter halbgeschlossener Vorderzungenvokal	midt	e
eː	e:	langer ungerunde ter halbgesch lossener Vorderzun genvokal	mele	e
ɛ	E	ungerundeter halboffener Vorderzungenvokal	mæt	E
ɛː	E:	langer ungerunde ter halboffener Vorderzungenvokal	mæle	E

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
i	i	ungerundeter geschlossener Vorderzungenvokal	mit	i
iː	i:	langer ungerunde ter geschlossener Vorderzungenvokal	mile	i
o	o	gerundeter halbgeschlossener Hinterzungenvokal	foto	o
oː	o:	langer gerundeter halbgeschlossener Hinterzungenvokal	mole	o
ɔ	O	gerundeter halboffener Hinterzungenvokal	mund	O
ɔː	O:	langer gerundeter halboffener hinterer Vokal	måle	O
ɒː	Q:	langer gerundeter offener Hinterzun genvokal	morse	O
u	u	gerundeter geschlossener Hinterzungenvokal	lusk	u
uː	u:	langer gerundete r geschlossener hinterer Vokal	mule	u

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ʌ	V	ungerundet halboffen an Hinterzunge	kører	E
y	y	geschlossener gerundeter Vorderzungenvokal	yt	u
y:	y:	langer gerundete r geschlossener Vorderzungenvokal	hyle	u
Zusätzliche Symbole				
ˈ	"	Hauptakzent	Alabama	
ˌ	%	Nebenakzent	Alabama	
.	.	Silbengrenze	A.la.ba.ma	

Niederländisch (Belgisch) (nl-BE)

In der folgenden Tabelle sind die Phoneme des Internationalen Phonetischen Alphabets (IPA), die X-SAMPA-Symbole (Extended Speech Assessment Methods Phonetic Alphabet) und die entsprechenden Viseme für die belgisch-niederländischen (flämischen) Stimmen aufgeführt, die von Amazon Polly unterstützt werden.

Tabelle der Phoneme/Viseme

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
Konsonanten				
b	b	stimmhafter bilabialer Verschlus slaut	bak	p

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
d	d	stimmhaft er alveolarer Verschlusslaut	dak	t
ʤ	dZ	stimmhafte postalveolare Affrikate	manager	S
f	f	stimmloser labiodentaler Frikativ	fel	f
g	g	stimmhafter velarer Verschlusslaut	goal	k
ɣ	G	stimmhafter velarer Frikativ	hoed	k
ɦ	h\	stimmhafter glottaler Frikativ	hand	k
j	j	palataler Approximant	ja	i
k	k	stimmloser velarer Verschlusslaut	kap	k
l	l	alveolarer lateraler Approximant	land	t
m	m	bilabialer Nasal	met	p
n	n	alveolarer Nasal	net	t
ŋ	N	velarer Nasal	Ba Ng	k

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
p	p	stimmloser bilabialer Verschlusslaut	pak	p
r	r	alveolarer Vibrant	rand	r
s	s	stimmloser alveolarer Reibelaut	sein	S
ʃ	S	stimmloser postalveolarer Reibelaut	show	S
t	t	stimmloser alveolarer Verschlusslaut	tak	t
V	V	stimmhafter labiodentaler Reibelaut	vel	f
ʋ	V\	labiodentaler Approximant	wit	f
x	x	stimmloser velarer Reibelaut	zu ch	k
z	z	stimmhafter alveolarer Reibelaut	ziin	s
ʒ	Z	stimmhafter postalveolarer Frikativ	bagage	S
Vokale				

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
øː	2:	langer gerundeter halbgeschlossener Vorderzungenvokal	neus	o
œy	9y	Dipthong	buit	O
ə	@	Schwa	de	@
aː	a:	langer ungerunde ter offener Vorderzungenvokal	baad	a
ɑː	A	ungerundeter offener Hinterzun genvokal	bad	a
eː	e:	langer ungerunde ter halbgesch lossener Vorderzun genvokal	beet	e
ɜː	3:	langer ungerunde ter halboffener Zentralvokal	barrière	E
ɛ	E	ungerundeter halboffener Vorderzungenvokal	bed	E
ɛi	Ei	Dipthong	beet	E
i	i	ungerundeter geschlossener Vorderzungenvokal	vier	i

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ɪ	I	ungerundeter zentralisierter fast geschlossener Vorderzungenvokal	pit	i
o:	o:	langer gerundeter halbgeschlossener Hinterzungenvokal	boot	o
ɔ	O	gerundeter halboffener Hinterzungenvokal	pot	O
u	u	gerundeter geschlossener Hinterzungenvokal	hoed	u
ʌu	Vu	Dipthong	fout	E
y:	y:	langer gerundete r geschlossener Vorderzungenvokal	fuut	u
ʏ	Y	gerundeter zentralisierter fast geschlossener Vorderzungenvokal	hut	u
Zusätzliche Symbole				
ˈ	"	Hauptakzent	Alabama	
ˌ	%	Nebenakzent	Alabama	
.	.	Silbengrenze	A.la.ba.ma	

Niederländisch (nl-NL)

In der folgenden Tabelle sind die Phoneme des Internationalen Phonetischen Alphabets (IPA), die X-SAMPA-Symbole (Extended Speech Assessment Methods Phonetic Alphabet) und die entsprechenden Viseme für die niederländischen Stimmen aufgeführt, die von Amazon Polly unterstützt werden.

Tabelle der Phoneme/Viseme

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
Konsonanten				
b	b	stimmhafter bilabialer Verschlus slaut	bak	p
d	d	stimmhaft er alveolarer Verschlusslaut	dak	t
ʤ	dZ	stimmhafte postalveolare Affrikate	manager	S
f	f	stimmloser labiodentaler Frikativ	fel	f
g	g	stimmhafter velarer Verschlusslaut	goal	k
ɣ	G	stimmhafter velarer Frikativ	hoed	k
ɦ	h\	stimmhafter glottaler Frikativ	hand	k

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
j	j	palataler Approximant	ja	i
k	k	stimmloser velarer Verschlusslaut	kap	k
l	l	alveolarer lateraler Approximant	land	t
m	m	bilabialer Nasal	met	p
n	n	alveolarer Nasal	net	t
ŋ	N	velarer Nasal	Ba Ng	k
p	p	stimmloser bilabialer Verschlus slaut	pak	p
r	r	alveolarer Vibrant	rand	r
s	s	stimmlose r alveolarer Reibelaut	sein	S
ʃ	S	stimmloser postalveolarer Reibelaut	show	S
t	t	stimmlose r alveolarer Verschlusslaut	tak	t
V	V	stimmhafter labiodentaler Reibelaut	vel	f

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ʋ	V\	labiodentaler Approximant	wit	f
x	x	stimmloser velarer Reibelaut	zu ch	k
z	z	stimmhaft er alveolarer Reibelaut	ziin	s
ʒ	Z	stimmhafter postalveolarer Frikativ	bagage	S
Vokale				
øː	2:	langer gerundeter halbgeschlossener Vorderzungenvokal	neus	o
œy	9y	Dipthong	buit	O
ə	@	Schwa	de	@
aː	a:	langer ungerunde ter offener Vorderzungenvokal	baad	a
ɑː	A	ungerundeter offener Hinterzun genvokal	bad	a
eː	e:	langer ungerunde ter halbgesch lossener Vorderzun genvokal	beet	e

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ɜː	3:	langer ungerunde ter halboffener Zentralvokal	barrière	E
ɛ	E	ungerundeter halboffener Vorderzungenvokal	bed	E
ɛi	Ei	Dipthong	beet	E
i	i	ungerundeter geschlossener Vorderzungenvokal	vier	i
ɪ	I	ungerundeter zentralisierter fast geschlossener Vorderzungenvokal	pit	i
oː	oː	langer gerundeter halbgeschlossener Hinterzungenvokal	boot	o
ɔ	O	gerundeter halboffener Hinterzungenvokal	pot	O
u	u	gerundeter geschlossener Hinterzungenvokal	hoed	u
ʌu	Vu	Dipthong	fout	E
yː	yː	langer gerundete r geschlossener Vorderzungenvokal	fuut	u

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ʏ	Y	gerundeter zentralisierter fast geschlossener Vorderzungenvokal	hut	u
Zusätzliche Symbole				
ˈ	"	Hauptakzent	Alabama	
ˌ	%	Nebenakzent	Alabama	
.	.	Silbengrenze	A.la.ba.ma	

Englisch (amerikanisch) (en-US)

In der folgenden Tabelle sind die Phoneme des Internationalen Phonetischen Alphabets (IPA), die X-SAMPA-Symbole (Extended Speech Assessment Methods Phonetic Alphabet) und die entsprechenden Viseme für amerikanische englische Stimmen aufgeführt, die von Amazon Polly unterstützt werden.

Tabelle der Phoneme/Viseme

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
Konsonanten				
b	b	stimmhafter bilabialer Verschlus slaut	bed	p
d	d	stimmhaft er alveolarer Verschlusslaut	dig	t
ʤ	dZ	stimmhafte postalveolare Affrikate	jump	S

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ð	D	stimmhafter dentaler Frikativ	then	T
f	f	stimmloser labiodentaler Frikativ	five	f
g	g	stimmhafter velarer Verschlusslaut	game	k
h	h	stimmloser glottaler Frikativ	house	k
j	j	palataler Approximant	yes	i
k	k	stimmloser velarer Verschlusslaut	cat	k
l	l	alveolarer lateraler Approximant	lay	l
m	m	bilabialer Nasal	mouse	p
n	n	alveolarer Nasal	nap	t
ŋ	N	velarer Nasal	thing	k
p	p	stimmloser bilabialer Verschlus slaut	speak	p
ɹ	r\	alveolarer Approximant	red	r
s	s	stimmlose r alveolarer Reibelaut	seem	S

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ʃ	S	stimmloser postalveolarer Reibelaut	ship	S
t	t	stimmlose r alveolarer Verschlusslaut	trap	t
͡tʃ	tS	stimmlose postalveolare Affrikate	chart	S
θ	T	stimmloser dentaler Reibelaut	thin	T
V	V	stimmhafter labiodentaler Reibelaut	vest	f
w	w	labiovelarer Approximant	west	u
z	z	stimmhaft er alveolarer Reibelaut	zero	s
ʒ	Z	stimmhafter postalveolarer Frikativ	vision	S
Vokale				
ə	@	Schwa	arena	@
ɚ	@`	rhotisches Schwa	reader	@

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
æ	{	ungerundeter fast offener Vorderzungenvokal	trap	a
aɪ	aI	Diphthong	price	a
aʊ	aU	Diphthong	mouth	a
ɑ	A	langer ungerundeter offener Hinterzungenvokal	father	a
eɪ	eI	Diphthong	face	e
ɝ	3`	offenes ungerundetes rhotisches Schwa	nurse	E
ɛ	E	ungerundeter halboffener Vorderzungenvokal	dress	E
i	i	langer ungerundeter geschlossener vorderer Vokal	fleece	i
ɪ	I	ungerundeter zentralisierter fast geschlossener Vorderzungenvokal	kit	i
oʊ	oU	Diphthong	goat	o
ɔ	O	langer gerundeter halboffener Hinterzungenvokal	thought	O

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ɔɪ	OI	Diphthong	choice	O
u	u	langer gerundeter geschlossener Hinterzungenvokal	goose	u
ʊ	U	gerundeter zentralisierter fast geschlossener Hinterzungenvokal	foot	u
ʌ	V	open-mid-back ungerundeter Vokal	strut	E
Zusätzliche Symbole				
ˈ	"	Hauptakzent	Alabama	
ˌ	%	Nebenakzent	Alabama	
.	.	Silbengrenze	A.la.ba.ma	

Englisch (australisch) (en-AU)

In der folgenden Tabelle sind die Phoneme des Internationalen Phonetischen Alphabets (IPA), die X-SAMPA-Symbole (Extended Speech Assessment Methods Phonetic Alphabet) und die entsprechenden Viseme für australische englische Stimmen aufgeführt, die von Amazon Polly unterstützt werden.

Tabelle der Phoneme/Viseme

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
Konsonanten				

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
b	b	stimmhafter bilabialer Verschlus slaut	bed	p
d	d	stimmhaft er alveolarer Verschlusslaut	dig	t
ʤ	dZ	stimmhafte postalveolare Affrikate	jump	S
ð	D	stimmhafter dentaler Frikativ	then	T
f	f	stimmloser labiodentaler Frikativ	five	f
g	g	stimmhafter velarer Verschlusslaut	game	k
h	h	stimmloser glottaler Frikativ	house	k
j	j	palataler Approximant	yes	i
k	k	stimmloser velarer Verschlusslaut	cat	k
l	l	alveolarer lateraler Approximant	lay	t
l̩	l=	silbischer alveolare r lateraler Approximant	battle	t

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
m	m	bilabialer Nasal	mouse	p
m̩	i=	silbischer bilabialer Nasal	anthem	p
n	n	alveolarer Nasal	nap	t
n̩	n=	silbischer alveolarer Nasal	button	t
ŋ	N	velarer Nasal	thing	k
p	p	stimmloser bilabialer Verschlusslaut	pin	p
ɹ	r\	alveolarer Approximant	red	r
s	s	stimmloser alveolarer Reibelaut	seem	S
ʃ	S	stimmloser postalveolarer Reibelaut	ship	S
t	t	stimmloser alveolarer Verschlusslaut	task	t
t͡ʃ	tS	stimmlose postalveolare Affrikate	chart	S
Θ	T	stimmloser dentaler Reibelaut	thin	T

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
V	V	stimmhafter labiodentaler Reibelaut	vest	f
w	w	labiovelarer Approximant	west	u
z	z	stimmhafter alveolarer Reibelaut	zero	s
ʒ	Z	stimmhafter postalveolarer Frikativ	vision	S
Vokale				
ə	@	Schwa	arena	@
əʊ	@U	Diphthong	goat	@
æ	{	ungerundeter fast offener Vorderzungenvokal	trap	a
aɪ	aI	Diphthong	price	a
aʊ	aU	Diphthong	mouth	a
ɑː	A:	langer ungerundeter offener Hinterzungenvokal	father	a
eɪ	eI	Diphthong	face	e
ɜː	3:	langer ungerundeter halboffener Zentralvokal	nurse	E

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ɛ	E	ungerundeter halboffener Vorderzungenvokal	dress	E
ɛə	E@	Diphthong	square	E
iː	i	langer ungerundeter geschlossener vorderer Vokal	fleece	i
ɪ	I	ungerundeter zentralisierter fast geschlossener Vorderzungenvokal	kit	i
ɪə	I@	Diphthong	near	i
ɔː	OI	langer gerundeter halboffener hinterer Vokal	thought	O
ɔɪ	OI	Diphthong	choice	O
ɒ	Q	gerundeter offener Hinterzungenvokal	lot	O
uː	u:	langer gerundeter geschlossener Hinterzungenvokal	goose	u
ʊ	U	gerundeter zentralisierter fast geschlossener Hinterzungenvokal	foot	u
ʊə	U@	Diphthong	cure	u

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ʌ	V	Open-mid-back ungerundeter Vokal	strut	E
Zusätzliche Symbole				
ˈ	"	Hauptakzent	Alabama	
ˌ	%	Nebenakzent	Alabama	
.	.	Silbengrenze	A.la.ba.ma	

Englisch (britisch) (en-GB)

In der folgenden Tabelle sind die Phoneme des Internationalen Phonetischen Alphabets (IPA), die X-SAMPA-Symbole (Extended Speech Assessment Methods Phonetic Alphabet) und die entsprechenden Viseme für britisch-englische Stimmen aufgeführt, die von Amazon Polly unterstützt werden.

Tabelle der Phoneme/Viseme

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
Konsonanten				
b	b	stimmhafter bilabialer Verschlus slaut	bed	p
d	d	stimmhaft er alveolarer Verschlusslaut	dig	t
ʤ	dZ	stimmhafte postalveolare Affrikate	jump	S

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ð	D	stimmhafter dentaler Frikativ	then	T
f	f	stimmloser labiodentaler Frikativ	five	f
g	g	stimmhafter velarer Verschlusslaut	game	k
h	h	stimmloser glottaler Frikativ	house	k
j	j	palataler Approximant	yes	i
k	k	stimmloser velarer Verschlusslaut	cat	k
l	l	alveolarer lateraler Approximant	lay	t
l̩	l=	silbischer alveolarer lateraler Approximant	battle	t
m	m	bilabialer Nasal	mouse	p
i̩	i=	silbischer bilabialer Nasal	anthem	p
n	n	alveolarer Nasal	nap	t
n̩	n=	silbischer alveolarer Nasal	button	t
ŋ	N	velarer Nasal	thing	k

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
p	p	stimmloser bilabialer Verschlusslaut	pin	p
ɹ	r\	alveolarer Approximant	red	r
s	s	stimmloser alveolarer Reibelaut	seem	S
ʃ	S	stimmloser postalveolarer Reibelaut	ship	S
t	t	stimmloser alveolarer Verschlusslaut	task	t
͡tʃ	tS	stimmlose postalveolare Affrikate	chart	S
Θ	T	stimmloser dentaler Reibelaut	thin	T
V	V	stimmhafter labiodentaler Reibelaut	vest	f
w	w	labiovelarer Approximant	west	u
z	z	stimmhafter alveolarer Reibelaut	zero	s

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ʒ	Z	stimmhafter postalveolarer Frikativ	vision	S
Vokale				
ə	@	Schwa	arena	@
əʊ	@U	Diphthong	goat	@
æ	{	ungerundeter fast offener Vorderzungenvokal	trap	a
aɪ	aI	Diphthong	price	a
aʊ	aU	Diphthong	mouth	a
ɑː	A:	langer ungerundeter offener Hinterzungenvokal	father	a
eɪ	eI	Diphthong	face	e
ɜː	3:	langer ungerundeter halboffener Zentralvokal	nurse	E
ɛ	E	ungerundeter halboffener Vorderzungenvokal	dress	E
ɛə	E@	Diphthong	square	E
iː	i	langer ungerundeter geschlossener vorderer Vokal	fleece	i

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ɪ	I	ungerundeter zentralisierter fast geschlossener Vorderzungenvokal	kit	i
ɪə	I@	Diphthong	near	i
ɔː	O:	langer gerundeter halboffener hinterer Vokal	thought	O
ɔɪ	OI	Diphthong	choice	O
ɒ	Q	gerundeter offener Hinterzungenvokal	lot	O
uː	u:	langer gerundeter geschlossener Hinterzungenvokal	goose	u
ʊ	U	gerundeter zentralisierter fast geschlossener Hinterzungenvokal	foot	u
ʊə	U@	Diphthong	cure	u
ʌ	V	Open-mid-back ungerundeter Vokal	strut	E
Zusätzliche Symbole				
ˈ	"	Hauptakzent	Alabama	
ˌ	%	Nebenakzent	Alabama	
.	.	Silbengrenze	A.la.ba.ma	

Englisch (indisch) (en-IN)

In der folgenden Tabelle sind die Phoneme des Internationalen Phonetischen Alphabets (IPA), die X-SAMPA-Symbole (Extended Speech Assessment Methods Phonetic Alphabet) und die entsprechenden Viseme für die indische englische Stimme aufgeführt, die von Amazon Polly unterstützt werden.

Zusätzliche Phoneme in Verbindung mit indischem Englisch finden Sie unter Hindi (hi-IN).

Tabelle der Phoneme/Viseme

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
Konsonanten				
b	b	stimmhafter bilabialer Verschlus slaut	bed	p
d	d	stimmhaft er alveolarer Verschlusslaut	dig	t
͡dʒ	dZ	stimmhafte postalveolare Affrikate	jump	S
ð	D	stimmhafter dentaler Frikativ	then	T
f	f	stimmloser labiodentaler Frikativ	five	f
g	g	stimmhafter velarer Verschlusslaut	game	k
h	h	stimmloser glottaler Frikativ	house	k

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
j	j	palataler Approximant	yes	i
k	k	stimmloser velarer Verschlusslaut	cat	k
l	l	alveolarer lateraler Approximant	lay	t
l̩	l=	silbischer alveolarer lateraler Approximant	battle	t
m	m	bilabialer Nasal	mouse	p
m̩	m=	silbischer bilabialer Nasal	anthem	p
n	n	alveolarer Nasal	nap	t
n̩	n=	silbischer alveolarer Nasal	nap	t
ŋ	N	velarer Nasal	thing	k
p	p	stimmloser bilabialer Verschlusslaut	pin	p
ɹ	r\	alveolarer Approximant	red	r
s	s	stimmloser alveolarer Reibelaut	seem	S

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ʃ	S	stimmloser postalveolarer Reibelaut	ship	S
t	t	stimmlose r alveolarer Verschlusslaut	task	t
͡tʃ	tS	stimmlose postalveolare Affrikate	chart	S
Θ	T	stimmloser dentaler Reibelaut	thin	T
V	V	stimmhafter labiodentaler Reibelaut	vest	f
w	w	labiovelarer Approximant	west	u
z	z	stimmhaft er alveolarer Reibelaut	zero	s
ʒ	Z	stimmhafter postalveolarer Frikativ	vision	S
Vokale				
ə	@	Schwa	arena	@
əʊ	@U	Diphthong	goat	@

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
æ	{	ungerundeter fast offener Vorderzungenvokal	trap	a
aɪ	aI	Diphthong	price	a
aʊ	aU	Diphthong	mouth	a
ɑː	A:	langer ungerundeter offener Hinterzungenvokal	father	a
eɪ	eI	Diphthong	face	e
ɜː	3:	langer ungerundeter halboffener Zentralvokal	nurse	E
ɛ	E	ungerundeter halboffener Vorderzungenvokal	dress	E
ɛə	E@	Diphthong	square	E
iː	i	langer ungerundeter geschlossener vorderer Vokal	fleece	i
ɪ	I	ungerundeter zentralisierter fast geschlossener Vorderzungenvokal	kit	i
ɪə	I@	Diphthong	near	i

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ɔː	OI	langer gerundeter halboffener hinterer Vokal	thought	O
ɔɪ	OI	Diphthong	choice	O
ɒ	Q	gerundeter offener Hinterzungenvokal	lot	O
uː	u:	langer gerundeter geschlossener Hinterzungenvokal	goose	u
ʊ	U	gerundeter zentralisierter fast geschlossener Hinterzungenvokal	foot	u
ʊə	U@	Diphthong	cure	u
ʌ	V	Open-mid-back ungerundeter Vokal	strut	E
Zusätzliche Symbole				
ˈ	"	Hauptakzent	Alabama	
ˌ	%	Nebenakzent	Alabama	
.	.	Silbengrenze	A.la.ba.ma	

Englisch (Irland) (en-IE)

In der folgenden Tabelle sind die Phoneme des Internationalen Phonetischen Alphabets (IPA), die X-SAMPA-Symbole (Extended Speech Assessment Methods Phonetic Alphabet) und die

entsprechenden Viseme für die Stimmen in irischem Englisch aufgeführt, die von Amazon Polly unterstützt werden.

Tabelle der Phoneme/Viseme

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
Konsonanten				
b	b	stimmhafter bilabialer Verschlusslaut	bed	p
d	d	stimmhaft er alveolarer Verschlusslaut	dig	t
ʤ	dZ	stimmhafte postalveolare Affrikate	jump	S
ð	D	stimmhafter dentaler Frikativ	then	T
f	f	stimmloser labiodentaler Frikativ	five	f
g	g	stimmhafter velarer Verschlusslaut	game	k
h	h	stimmloser glottaler Frikativ	house	k
j	j	palataler Approximant	yes	i
k	k	stimmloser velarer Verschlusslaut	cat	k

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
l	l	alveolarer lateraler Approximant	lay	t
m	m	bilabialer Nasal	mouse	p
n	n	alveolarer Nasal	nap	t
ŋ	N	velarer Nasal	thing	k
p	p	stimmloser bilabialer Verschlusslaut	speak	p
ɹ	r\	alveolarer Approximant	red	r
s	s	stimmloser alveolarer Reibelaut	seem	S
ʃ	S	stimmloser postalveolarer Reibelaut	ship	S
t	t	stimmloser alveolarer Verschlusslaut	trap	t
t͡ʃ	tS	stimmlose postalveolare Affrikate	chart	S
θ	T	stimmloser dentaler Reibelaut	thin	T

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
v	V	stimmhafter labiodentaler Reibelaut	vest	f
w	w	labiovelarer Approximant	west	u
z	z	stimmhafter alveolarer Reibelaut	zero	s
ʒ	Z	stimmhafter postalveolarer Frikativ	vision	S
Vokale				
ə	@	Schwa	arena	@
ɚ	@`	rhotisches Schwa	reader	@
æ	{	ungerundeter fast offener Vorderzungenvokal	trap	a
aɪ	aI	Diphthong	price	a
aʊ	aU	Diphthong	mouth	a
ɑ	A	langer ungerundeter offener Hinterzungenvokal	father	a
eɪ	eI	Diphthong	face	e
ɝ	3`	offenes ungerundetes rhotisches Schwa	nurse	E

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ɛ	E	ungerundeter halboffener Vorderzungenvokal	dress	E
i	i	langer ungerunde ter geschlossener vorderer Vokal	fleece	i
ɪ	I	ungerundeter zentralisierter fast geschlossener Vorderzungenvokal	kit	i
oʊ	oU	Diphthong	goat	o
ɔ	O	langer gerundete r halboffener Hinterzungenvokal	thought	O
ɔɪ	OI	Diphthong	choice	O
u	u	langer gerundete r geschlossener Hinterzungenvokal	goose	u
ʊ	U	gerundeter zentralisierter fast geschlossener Hinterzungenvokal	foot	u
ʌ	V	open-mid-back ungerundeter Vokal	strut	E
Zusätzliche Symbole				
ˈ	"	Hauptakzent	Alabama	

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ˌ	%	Nebenakzent	Alabama	
.	.	Silbengrenze	A.la.ba.ma	

Englisch (Neuseeland) (en-NZ)

In der folgenden Tabelle sind die Phoneme des Internationalen Phonetischen Alphabets (IPA), die X-SAMPA-Symbole (Extended Speech Assessment Methods Phonetic Alphabet) und die entsprechenden Viseme für die Neuseeland englischen Stimmen aufgeführt, die von Amazon Polly unterstützt werden.

Tabelle der Phoneme/Viseme

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
Konsonanten				
b	b	stimmhafter bilabialer Verschlus slaut	bed	p
d	d	stimmhaft er alveolarer Verschlusslaut	dig	t
ʤ	dZ	stimmhafte postalveolare Affrikate	jump	S
ð	D	stimmhafter dentaler Frikativ	then	T
f	f	stimmloser labiodentaler Frikativ	five	f

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
g	g	stimmhafter velarer Verschlusslaut	game	k
h	h	stimmloser glottaler Frikativ	house	k
j	j	palataler Approximant	yes	i
k	k	stimmloser velarer Verschlusslaut	cat	k
l	l	alveolarer lateraler Approximant	lay	t
l̩	l=	silbischer alveolarer lateraler Approximant	battle	t
m	m	bilabialer Nasal	mouse	p
i̩	i=	silbischer bilabialer Nasal	anthem	p
n	n	alveolarer Nasal	nap	t
n̩	n=	silbischer alveolarer Nasal	button	t
ŋ	N	velarer Nasal	thing	k
p	p	stimmloser bilabialer Verschlusslaut	pin	p
ɹ	r\	alveolarer Approximant	red	r

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
s	s	stimmlose r alveolarer Reibelaut	seem	S
ʃ	S	stimmloser postalveolarer Reibelaut	ship	S
t	t	stimmlose r alveolarer Verschlusslaut	task	t
͡tʃ	tS	stimmlose postalveolare Affrikate	chart	S
Θ	T	stimmloser dentaler Reibelaut	thin	T
V	V	stimmhafter labiodentaler Reibelaut	vest	f
w	w	labiovelarer Approximant	west	u
z	z	stimmhaft er alveolarer Reibelaut	zero	s
ʒ	Z	stimmhafter postalveolarer Frikativ	vision	S
Vokale				
ə	@	Schwa	arena	@

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
əʊ	@U	Diphthong	goat	@
æ	{	ungerundeter fast offener Vorderzungenvokal	trap	a
aɪ	aI	Diphthong	price	a
aʊ	aU	Diphthong	mouth	a
ɑː	A:	langer ungerundeter offener Hinterzungenvokal	father	a
eɪ	eI	Diphthong	face	e
ɜː	3:	langer ungerundeter halboffener Zentralvokal	nurse	E
ɛ	E	ungerundeter halboffener Vorderzungenvokal	dress	E
ɛə	E@	Diphthong	square	E
iː	i	langer ungerundeter geschlossener vorderer Vokal	fleece	i
ɪ	I	ungerundeter zentralisierter fast geschlossener Vorderzungenvokal	kit	i
ɪə	I@	Diphthong	near	i

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ɔː	O:	langer gerundeter halboffener hinterer Vokal	thought	O
ɔɪ	OI	Diphthong	choice	O
ɒ	Q	gerundeter offener Hinterzungenvokal	lot	O
uː	u:	langer gerundeter geschlossener Hinterzungenvokal	goose	u
ʊ	U	gerundeter zentralisierter fast geschlossener Hinterzungenvokal	foot	u
ʊə	U@	Diphthong	cure	u
ʌ	V	Open-mid-back ungerundeter Vokal	strut	E
Zusätzliche Symbole				
ˈ	"	Hauptakzent	Alabama	
ˌ	%	Nebenakzent	Alabama	
.	.	Silbengrenze	A.la.ba.ma	

Die Aria-Stimme spricht Neuseeland Englisch und bietet Maori nur begrenzte Unterstützung. Es kann die folgenden Maori-Wörter und -Sätze aussprechen. Bei den Maori-Sätzen wird zwischen Groß- und Kleinschreibung unterschieden.

Englisch	Maori
Hallo/Prost	Kia Ora
Willkommen (bei)	Nau Mai (Ki)
Hallo (eine Person) /danke	Tēnā koe
Hallo (drei oder mehr Personen) /Danke	Tēnā Koutou
Guten Morgen	Ata Mārie
Guten Morgen	Morena
Danke	Ngā mihi
Pass auf dich auf	Ngā manaakitanga
Wir sehen uns	Ka Kite
Wir sehen uns später	Mā te wā
Hab einen schönen Tag	Kia Pai Tō Rā
Fröhliche Weihnachten	Meri Kirihimete
Maori	Māori
Sprache der Maori	die Areo-Māori
Woche der Maori-Sprache	Das Wiki der Kreo Māori
Neuseeland	Aotearoa
Neues Jahr der Maori	Mātariki
Stadt in Neuseeland//Der Waitangi Day ist der Nationalfeiertag Neuseeland	Waitangi
One	Tahi
Zwei	Rua

Englisch	Maori
Drei	Toru
Vier	whā
Five	Rima
Sechs	Sono
Sieben	weiß
Acht	Waru
Neun	iwa
Zehn	Tekau
Zwanzig	Rua Tekau
Dreißig	Toru Tekau

Englisch (Singapurisch) (en-SG)

In der folgenden Tabelle sind die Phoneme des Internationalen Phonetischen Alphabets (IPA), die X-SAMPA-Symbole (Extended Speech Assessment Methods Phonetic Alphabet) und die entsprechenden Viseme für die englischen Stimmen in Singapur aufgeführt, die von Amazon Polly unterstützt werden.

Tabelle der Phoneme/Viseme

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
Konsonanten				
b	b	stimmhafter bilabialer Verschlus slaut	bed	p

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
d	d	stimmhaft er alveolarer Verschlusslaut	dig	t
͡dʒ	dZ	stimmhafte postalveolare Affrikate	jump	S
ð	D	stimmhafter dentaler Frikativ	then	T
f	f	stimmloser labiodentaler Frikativ	five	f
g	g	stimmhafter velarer Verschlusslaut	game	k
h	h	stimmloser glottaler Frikativ	house	k
j	j	palataler Approximant	yes	i
k	k	stimmloser velarer Verschlusslaut	cat	k
l	l	alveolarer lateraler Approximant	lay	t
l̩	l=	silbischer alveolarer lateraler Approximant	battle	t
m	m	bilabialer Nasal	mouse	p

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ˌi	i=	silbischer bilabialer Nasal	anthem	p
n	n	alveolarer Nasal	nap	t
n̩	n=	silbischer alveolarer Nasal	button	t
ŋ	N	velarer Nasal	thing	k
p	p	stimmloser bilabialer Verschlusslaut	pin	p
ɹ	r\	alveolarer Approximant	red	r
s	s	stimmloser alveolarer Reibelaut	seem	S
ʃ	S	stimmloser postalveolarer Reibelaut	ship	S
t	t	stimmloser alveolarer Verschlusslaut	task	t
t͡ʃ	tS	stimmlose postalveolare Affrikate	chart	S
Θ	T	stimmloser dentaler Reibelaut	thin	T

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
v	v	stimmhafter labiodentaler Reibelaut	vest	f
w	w	labiovelarer Approximant	west	u
z	z	stimmhafter alveolarer Reibelaut	zero	s
ʒ	Z	stimmhafter postalveolarer Frikativ	vision	S
Vokale				
ə	@	Schwa	arena	@
əʊ	@U	Diphthong	goat	@
æ	{	ungerundeter fast offener Vorderzungenvokal	trap	a
aɪ	aI	Diphthong	price	a
aʊ	aU	Diphthong	mouth	a
ɑː	A:	langer ungerundeter offener Hinterzungenvokal	father	a
eɪ	eI	Diphthong	face	e
ɜː	3:	langer ungerundeter halboffener Zentralvokal	nurse	E

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ɛ	E	ungerundeter halboffener Vorderzungenvokal	dress	E
ɛə	E@	Diphthong	square	E
iː	i	langer ungerundeter geschlossener vorderer Vokal	fleece	i
ɪ	I	ungerundeter zentralisierter fast geschlossener Vorderzungenvokal	kit	i
ɪə	I@	Diphthong	near	i
ɔː	O:	langer gerundeter halboffener hinterer Vokal	thought	O
ɔɪ	OI	Diphthong	choice	O
ɒ	Q	gerundeter offener Hinterzungenvokal	lot	O
uː	u:	langer gerundeter geschlossener Hinterzungenvokal	goose	u
ʊ	U	gerundeter zentralisierter fast geschlossener Hinterzungenvokal	foot	u
ʊə	U@	Diphthong	cure	u

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ʌ	V	Open-mid-back ungerundeter Vokal	strut	E
Zusätzliche Symbole				
ˈ	"	Hauptakzent	Alabama	
ˌ	%	Nebenakzent	Alabama	
.	.	Silbengrenze	A.la.ba.ma	

Englisch (Südafrikanisch) (en-ZA)

In der folgenden Tabelle sind die Phoneme des Internationalen Phonetischen Alphabets (IPA), die X-SAMPA-Symbole (Extended Speech Assessment Methods Phonetic Alphabet) und die entsprechenden Viseme für die südafrikanischen englischen Stimmen aufgeführt, die von Amazon Polly unterstützt werden.

Tabelle der Phoneme/Viseme

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
Konsonanten				
b	b	stimmhafter bilabialer Verschlus slaut	bed	p
d	d	stimmhaft er alveolarer Verschlusslaut	dig	t
ʤ	dZ	stimmhafte postalveolare Affrikate	jump	S

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ð	D	stimmhafter dentaler Frikativ	then	T
f	f	stimmloser labiodentaler Frikativ	five	f
g	g	stimmhafter velarer Verschlusslaut	game	k
h	h	stimmloser glottaler Frikativ	house	k
j	j	palataler Approximant	yes	i
k	k	stimmloser velarer Verschlusslaut	cat	k
l	l	alveolarer lateraler Approximant	lay	t
l̩	l=	silbischer alveolarer lateraler Approximant	battle	t
ɬ̩	K	stimmloser lateraler Frikativ	Umm Hl Manga	t
m	m	bilabialer Nasal	mouse	p
i̩	i=	silbischer bilabialer Nasal	anthem	p
n	n	alveolarer Nasal	nap	t
n̩	n=	silbischer alveolarer Nasal	button	t

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ŋ	N	velarer Nasal	thing	k
p	p	stimmloser bilabialer Verschlus slaut	pin	p
ɹ	r\	alveolarer Approximant	red	r
r	r	alveolarer Vibrant	Pa oder Reis	r
s	s	stimmlose r alveolarer Reibelaut	seem	S
ʃ	S	stimmloser postalveolarer Reibelaut	ship	S
t	t	stimmlose r alveolarer Verschlusslaut	task	t
‿tʃ	tS	stimmlose postalveolare Affrikate	chart	S
Θ	T	stimmloser dentaler Reibelaut	thin	T
V	V	stimmhafter labiodentaler Reibelaut	vest	f
w	w	labiovelarer Approximant	west	u

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
x	x	stimmloser velarer Reibelaut	von Tauteng	k
z	z	stimmhaft er alveolarer Reibelaut	zero	S
!	!\	postalveolärer Klick	gq Eberha	k
|	|\	zahnärztlicher Klick	Auf einem RC-Würfel	t
||	||\	seitlicher Klick	XH Osa	t
Vokale				
ə	@	Schwa	eine Arena	@
Məi	@i	Diphthong	nelspr ist	i
əʊ	@U	Diphthong	goat	@
æ	{	ungerundeter fast offener Vorderzungenvokal	trap	a
aɪ	aI	Diphthong	price	a
aʊ	aU	Diphthong	mouth	a
ɑː	A:	langer ungerunder offener Hinterzungenvokal	father	a
eɪ	eI	Diphthong	face	e

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ɜː	3:	langer ungerunde ter halboffener Zentralvokal	nurse	E
ɛ	E	ungerundeter halboffener Vorderzungenvokal	dress	E
ɛə	E@	Diphthong	square	E
iː	i	langer ungerunde ter geschlossener vorderer Vokal	fleece	i
i ə	I@	Diphthong	du oder eez	i
ɪ	I	ungerundeter zentralisierter fast geschlossener Vorderzungenvokal	kit	i
ɪə	I@	Diphthong	near	i
ɔː	O:	langer gerundeter halboffener hinterer Vokal	thought	O
ɔɪ	OI	Diphthong	choice	O
ɒ	Q	gerundeter offener Hinterzungenvokal	lot	O
uː	u:	langer gerundete r geschlossener Hinterzungenvokal	goose	u

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ʊ	U	gerundeter zentralisierter fast geschlossener Hinterzungenvokal	foot	u
ʊə	U@	Diphthong	cure	u
ʌ	V	Open-mid-back ungerundeter Vokal	strut	E
y	y	geschlossener gerundeter Vorderzungenvokal	van v uu ren	u
Zusätzliche Symbole				
ˈ	"	Hauptakzent	Alabama	
ˌ	%	Nebenakzent	Alabama	
.	.	Silbengrenze	A.la.ba.ma	

Englisch (walisisch) (en-GB-WLS)

In der folgenden Tabelle sind die Phoneme des Internationalen Phonetischen Alphabets (IPA), die X-SAMPA-Symbole (Extended Speech Assessment Methods Phonetic Alphabet) und die entsprechenden Viseme für die walisisch-englische Stimme aufgeführt, die von Amazon Polly unterstützt werden.

Tabelle der Phoneme/Viseme

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
Konsonanten				

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
b	b	stimmhafter bilabialer Verschlusslaut	bed	p
d	d	stimmhafter alveolarer Verschlusslaut	dig	t
ʤ	dZ	stimmhafte postalveolare Affrikate	jump	S
ð	D	stimmhafter dentaler Frikativ	then	T
f	f	stimmloser labiodentaler Frikativ	five	f
g	g	stimmhafter velarer Verschlusslaut	game	k
h	h	stimmloser glottaler Frikativ	house	k
j	j	palataler Approximant	yes	i
k	k	stimmloser velarer Verschlusslaut	cat	k
l	l	alveolarer lateraler Approximant	lay	t
l̩	l=	silbischer alveolarer lateraler Approximant	battle	t

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
m	m	bilabialer Nasal	mouse	p
m̩	i=	silbischer bilabialer Nasal	anthem	p
n	n	alveolarer Nasal	nap	t
n̩	n=	silbischer alveolarer Nasal	nap	t
ŋ	N	velarer Nasal	thing	k
p	p	stimmloser bilabialer Verschlusslaut	pin	p
ɹ	r\	alveolarer Approximant	red	r
s	s	stimmloser alveolarer Reibelaut	seem	S
ʃ	S	stimmloser postalveolarer Reibelaut	ship	S
t	t	stimmloser alveolarer Verschlusslaut	task	t
t͡ʃ	tS	stimmlose postalveolare Affrikate	chart	S
Θ	T	stimmloser dentaler Reibelaut	thin	T

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
v	v	stimmhafter labiodentaler Reibelaut	vest	f
w	w	labiovelarer Approximant	west	u
z	z	stimmhafter alveolarer Reibelaut	zero	s
ʒ	Z	stimmhafter postalveolarer Frikativ	vision	S
Vokale				
ə	@	Schwa	arena	@
əʊ	@U	Diphthong	goat	@
æ	{	ungerundeter fast offener Vorderzungenvokal	trap	a
aɪ	aI	Diphthong	price	a
aʊ	aU	Diphthong	mouth	a
ɑː	A:	langer ungerundeter offener Hinterzungenvokal	father	a
eɪ	eI	Diphthong	face	e
ɜː	3:	langer ungerundeter halboffener Zentralvokal	nurse	E

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ɛ	E	ungerundeter halboffener Vorderzungenvokal	dress	E
ɛə	E@	Diphthong	square	E
iː	i	langer ungerundeter geschlossener vorderer Vokal	fleece	i
ɪ	I	ungerundeter zentralisierter fast geschlossener Vorderzungenvokal	kit	i
ɪə	I@	Diphthong	near	i
ɔː	OI	langer gerundeter halboffener hinterer Vokal	thought	O
ɔɪ	OI	Diphthong	choice	O
ɒ	Q	gerundeter offener Hinterzungenvokal	lot	O
uː	u:	langer gerundeter geschlossener Hinterzungenvokal	goose	u
ʊ	U	gerundeter zentralisierter fast geschlossener Hinterzungenvokal	foot	u
ʊə	U@	Diphthong	cure	u

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ʌ	V	Open-mid-back ungerundeter Vokal	strut	E
Zusätzliche Symbole				
ˈ	"	Hauptakzent	Alabama	
ˌ	%	Nebenakzent	Alabama	
.	.	Silbengrenze	A.la.ba.ma	

Finnisch (Fi-FI)

In der folgenden Tabelle sind die Phoneme des Internationalen Phonetischen Alphabets (IPA), die X-SAMPA-Symbole (Extended Speech Assessment Methods Phonetic Alphabet) und die entsprechenden Viseme für die finnische Stimme aufgeführt, die von Amazon Polly unterstützt wird.

Tabelle der Phoneme/Viseme

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
Finnische Konsonanten				
p	p	stimmloser bilabialer Verschlus slaut	[p] ankki	p
t	t	stimmlose r alveolarer Verschlusslaut	[t] auch	t
k	k	stimmloser velarer Verschlusslaut	[k] aali	k

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
d	d	stimmhaft er alveolarer Verschlusslaut	[d] Daten	t
S	s	stimmlose r alveolarer Reibelaut	[s] Ali	S
h	h	stimmloser glottaler Frikativ	[h] attu	k
ʋ	V\	stimmiger labiodent aler Approximant	[v] aiva	V
j	j	palataler Approximant	[j] oki	i
l	l	alveolarer lateraler Approximant	[l] Oma	t
r	r	stimmiger Alveolart hriller	[r] iita	r
m	m	bilabialer Nasal	[m] Auto	p
n	n	alveolarer Nasal	[n] enää	t
ŋ	N	velarer Nasal	Das [n] Kind	k
Konsonanten, die in Lehnwörtern gefunden wurden				
b	b	stimmhafter bilabialer Verschlus slaut	[b] ussi	p
f	f	stimmloser labiodentaler Frikativ	[f] Firma	V

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
w	w	labiovelarer Approximant	[w] Wiki	u
z	z	stimmhaft er alveolarer Reibelaut	[z] ulu	S
g	g	stimmhafter velarer Verschlusslaut	[g] Aala	k
ʃ	S	stimmloser postalveolarer Reibelaut	[sh] akki	S
ʒ	Z	stimmhafter postalveolarer Frikativ	[g] Genre	S
θ	T	stimmloser dentaler Reibelaut	nahe [th]	T
ð	D	stimmhafter dentaler Frikativ	ei [th] er	T
Kurze Vokale				
i	i	ungerundeter geschlossener Vorderzungenvokal	k [i] lo	i
ɛ	E	ungerundeter halboffener Vorderzungenvokal	k [e] sä	E
æ	{	ungerundeter fast offener Vorderzun genvokal	k [ä]	A

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
y	y	geschlossener gerundeter Vorderzungenvokal	k [y] l	u
ø	2	geschlossener abgerundeter Vokal in der Mitte	p [ö] ly	O
u	u	gerundeter geschlossener Hinterzungenvokal	[u] lo	u
ɔ	O	offener abgerunde ter Vokal in der Mitte des Rückens	k [o] niedrig	O
ɑ	A	ungerundeter offener Hinterzun genvokal	[ka] la	A
Lange Vokale				
iː	iː	langer ungerunde ter geschlossener Vorderzungenvokal	[ii] li	i
ɛː	E:	langer offener ungerundeter Vokal in der Mitte vorne	[ee] tu	E
æː	{:	langer, fast offener, ungerundeter Vokal	t [ää] llä	A

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
y:	y:	langer ungerunde ter geschlossener Vorderzungenvokal	t [jj] li	u
øː	2:	langer, geschloss ener, abgerundeter Vokal in der Mitte	t [öö] ö	O
uː	u:	langer gerundete r geschlossener hinterer Vokal	[uu] li	u
ɔː	O:	langer gerundete r halboffener Hinterzungenvokal	[Roo] li	O
ɑː	A:	langer ungerunde ter offener Hinterzungenvokal	[ka] us	A
Dipthongs				
ɛi	Ei	Dipthong	l [ei] ä	E
æi	{i	Dipthong	[äi] ti	A
ui	ui	Dipthong	[kuin]	u
i	Ai	Dipthong	[Kai] kki	A
ɔi	Oi	Dipthong	[Poi] ka	O
øi	2i	Dipthong	[söin]	O
yi	yi	Dipthong	l [yi] jy	u
u	Au	Dipthong	[Saun]	A

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ɔu	Ou	Dipthong	[du] lu	O
õu	Eu	Dipthong	r [eu] na	E
iu	iu	Dipthong	[iu] lu	i
Væy	{y	Dipthong	[äy] nä	A
øy	2 Jahre	Dipthong	[köy] hä	O
ɛy	Ey	Dipthong	pes [ey] tyä	E
es	es	Dipthong	kär [it] tyä	i
i	Ie	Dipthong	[sterben]	i
jö	y2	Dipthong	[du]	u
uɔ	Uo	Dipthong	[zu]	u
Vokale, die in englischen Lehnwörtern vorkommen				
ɪ	I	ungerundeter zentralisierter fast geschlossener Vorderzungenvokal	[b it]	i
ʊ	U	gerundeter zentralisierter fast geschlossener Hinterzungenvokal	b [ok]	u
ə	@	Schwa	[a] ungefähr	@
ʌ	V	open-mid-back ungerundeter Vokal	[u] t	E

Französisch (fr-FR)

In der folgenden Tabelle sind die Phoneme des Internationalen Phonetischen Alphabets (IPA), die X-SAMPA-Symbole (Extended Speech Assessment Methods Phonetic Alphabet) und die entsprechenden Viseme für die französischen Stimmen aufgeführt, die von Amazon Polly unterstützt werden.

Tabelle der Phoneme/Viseme

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
Konsonanten				
b	b	stimmhafter bilabialer Verschlusslaut	boire	p
d	d	stimmhafter alveolarer Verschlusslaut	madame	t
f	f	stimmloser labiodentaler Frikativ	femme	f
g	g	stimmhafter velarer Verschlusslaut	grand	k
ɥ	H	labiopalataler Approximant	bruit	u
j	j	palataler Approximant	meilleur	i
k	k	stimmloser velarer Verschlusslaut	quatre	k
l	l	alveolarer lateraler Approximant	malade	t

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
m	m	bilabialer Nasal	maison	p
n	n	alveolarer Nasal	astronome	t
ɲ	J	palataler Nasal	baigner	J
ŋ	N	velarer Nasal	parking	k
p	p	stimmloser bilabialer Verschlusslaut	pomme	p
ʁ	R	stimmhafter uvularer Frikativ	amoureux	k
s	s	stimmloser alveolarer Reibelaut	santé	S
ʃ	S	stimmloser postalveolarer Reibelaut	chat	S
t	t	stimmloser alveolarer Verschlusslaut	téléphone	t
V	V	stimmhafter labiodentaler Reibelaut	vrai	f
w	w	labiovelarer Approximant	soir	u
z	z	stimmhafter alveolarer Reibelaut	raison	s

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ʒ	Z	stimmhafter postalveolarer Frikativ	aubergine	S
Vokale				
ø	2	gerundeter halbgeschlossener Vorderzungenvokal	deux	o
œ	9	gerundeter halboffener Vorderzungenvokal	neuf	O
œ̃	9~	nasaler halboffen er gerundeter Vorderzungenvokal	brun	O
ə	@	Schwa	je	@
a	a	ungerundeter offener Vor	table	a
ɑ̃	A~	nasaler offener ungerundeter Hinterzungenvokal	camembert	a
e	e	ungerundeter halbgeschlossener Vorderzungenvokal	marché	e
ɛ	E	ungerundeter halboffener Vorderzungenvokal	neige	E

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ɛ̃	E~	nasaler ungerundeter halboffener Vorderzungenvokal	sapin	E
i	i	ungerundeter geschlossener Vorderzungenvokal	mille	i
o	o	gerundeter halbgeschlossener Hinterzungenvokal	hôpital	o
ɔ	O	gerundeter halboffener Hinterzungenvokal	homme	O
ɔ̃	O~	nasaler gerundeter halboffener Hinterzungenvokal	bon	O
u	u	gerundeter geschlossener Hinterzungenvokal	sous	u
y	y	geschlossener gerundeter Vorderzungenvokal	dur	u
Zusätzliche Symbole				
ˈ	"	Hauptakzent	Alabama	
ˌ	%	Nebenakzent	Alabama	
.	.	Silbengrenze	A.la.ba.ma	

Französisch (Belgisch) (fr-BE)

In der folgenden Tabelle sind die Phoneme des Internationalen Phonetischen Alphabets (IPA), die X-SAMPA-Symbole (Extended Speech Assessment Methods Phonetic Alphabet) und die entsprechenden Viseme für die belgischen französischen Stimmen aufgeführt, die von Amazon Polly unterstützt werden.

Tabelle der Phoneme/Viseme

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
Konsonanten				
b	b	stimmhafter bilabialer Verschlusslaut	boire	p
d	d	stimmhafter alveolarer Verschlusslaut	madame	t
f	f	stimmloser labiodentaler Frikativ	femme	f
g	g	stimmhafter velarer Verschlusslaut	grand	k
ɥ	H	labiopalataler Approximant	bruit	u
j	j	palataler Approximant	meilleur	i
k	k	stimmloser velarer Verschlusslaut	quatre	k
l	l	alveolarer lateraler Approximant	malade	t

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
m	m	bilabialer Nasal	maison	p
n	n	alveolarer Nasal	astronome	t
ɲ	J	palataler Nasal	baigner	J
ŋ	N	velarer Nasal	parking	k
p	p	stimmloser bilabialer Verschlusslaut	pomme	p
ʁ	R	stimmhafter uvularer Frikativ	amoureux	k
s	s	stimmlose r alveolarer Reibelaut	santé	S
ʃ	S	stimmloser postalveolarer Reibelaut	chat	S
t	t	stimmlose r alveolarer Verschlusslaut	téléphone	t
V	V	stimmhafter labiodentaler Reibelaut	vrai	f
w	w	labiovelarer Approximant	soir	u
z	z	stimmhafter alveolarer Reibelaut	raison	s

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ʒ	Z	stimmhafter postalveolarer Frikativ	aubergine	S
Vokale				
ø	2	gerundeter halbgeschlossener Vorderzungenvokal	deux	o
œ	9	gerundeter halboffener Vorderzungenvokal	neuf	O
œ̃	9~	nasaler halboffen er gerundeter Vorderzungenvokal	brun	O
ə	@	Schwa	je	@
a	a	ungerundeter offener Vor	table	a
ɑ̃	A~	nasaler offener ungerundeter Hinterzungenvokal	camembert	a
e	e	ungerundeter halbgeschlossener Vorderzungenvokal	marché	e
ɛ	E	ungerundeter halboffener Vorderzungenvokal	neige	E

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ɛ̃	E~	nasaler ungerundeter halboffener Vorderzungenvokal	sapin	E
i	i	ungerundeter geschlossener Vorderzungenvokal	mille	i
o	o	gerundeter halbgeschlossener Hinterzungenvokal	hôpital	o
ɔ	O	gerundeter halboffener Hinterzungenvokal	homme	O
ɔ̃	O~	nasaler gerundeter halboffener Hinterzungenvokal	bon	O
u	u	gerundeter geschlossener Hinterzungenvokal	sous	u
y	y	geschlossener gerundeter Vorderzungenvokal	dur	u
Zusätzliche Symbole				
ˈ	"	Hauptakzent	Alabama	
ˌ	%	Nebenakzent	Alabama	
.	.	Silbengrenze	A.la.ba.ma	

Kanadisches Französisch (fr-CA)

In der folgenden Tabelle sind die Phoneme des Internationalen Phonetischen Alphabets (IPA), die X-SAMPA-Symbole (Extended Speech Assessment Methods Phonetic Alphabet) und die entsprechenden Viseme für die französisch-kanadische Stimme aufgeführt, die von Amazon Polly unterstützt werden.

Tabelle der Phoneme/Viseme

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
Konsonanten				
b	b	stimmhafter bilabialer Verschlusslaut	boire	p
d	d	stimmhafter alveolarer Verschlusslaut	madame	t
f	f	stimmloser labiodentaler Frikativ	femme	f
g	g	stimmhafter velarer Verschlusslaut	grand	k
ɥ	H	labiopalataler Approximant	bruit	u
j	j	palataler Approximant	meilleur	i
k	k	stimmloser velarer Verschlusslaut	quatre	k
l	l	alveolarer lateraler Approximant	malade	t

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
m	m	bilabialer Nasal	maison	p
n	n	alveolarer Nasal	astronome	t
ɲ	J	palataler Nasal	baigner	J
ŋ	N	velarer Nasal	parking	k
p	p	stimmloser bilabialer Verschlusslaut	pomme	p
ʁ	R	stimmhafter uvularer Frikativ	amoureux	k
s	s	stimmlose r alveolarer Reibelaut	santé	S
ʃ	S	stimmloser postalveolarer Reibelaut	chat	S
t	t	stimmlose r alveolarer Verschlusslaut	téléphone	t
V	V	stimmhafter labiodentaler Reibelaut	vrai	f
w	w	labiovelarer Approximant	soir	u
z	z	stimmhafter alveolarer Reibelaut	raison	s

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ʒ	Z	stimmhafter postalveolarer Frikativ	aubergine	S
Vokale				
ø	2	gerundeter halbgeschlossener Vorderzungenvokal	deux	o
œ	9	gerundeter halboffener Vorderzungenvokal	neuf	O
œ̃	9~	nasaler halboffen er gerundeter Vorderzungenvokal	brun	O
ə	@	Schwa	je	@
a	a	ungerundeter offener Vor	table	a
ɑ̃	A~	nasaler offener ungerundeter Hinterzungenvokal	camembert	a
e	e	ungerundeter halbgeschlossener Vorderzungenvokal	marché	e
ɛ	E	ungerundeter halboffener Vorderzungenvokal	neige	E

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ɛ̃	E~	nasaler ungerundeter halboffener Vorderzungenvokal	sapin	E
i	i	ungerundeter geschlossener Vorderzungenvokal	mille	i
o	o	gerundeter halbgeschlossener Hinterzungenvokal	hôpital	o
ɔ	O	gerundeter halboffener Hinterzungenvokal	homme	O
ɔ̃	O~	nasaler gerundeter halboffener Hinterzungenvokal	bon	O
u	u	gerundeter geschlossener Hinterzungenvokal	sous	u
y	y	geschlossener gerundeter Vorderzungenvokal	dur	u
Zusätzliche Symbole				
ˈ	"	Hauptakzent	Alabama	
ˌ	%	Nebenakzent	Alabama	
.	.	Silbengrenze	A.la.ba.ma	

Deutsch (de-DE)

In der folgenden Tabelle sind die Phoneme des Internationalen Phonetischen Alphabets (IPA), die X-SAMPA-Symbole (Extended Speech Assessment Methods Phonetic Alphabet) und die entsprechenden Viseme für die deutschen Stimmen aufgeführt, die von Amazon Polly unterstützt werden.

Tabelle der Phoneme/Viseme

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
Konsonanten				
ʔ	?	Glottallaut		
b	b	stimmhafter bilabialer Verschlus slaut	Bier	p
d	d	stimmhaft er alveolarer Verschlusslaut	Dach	t
ç	C	stimmloser palataler Frikativ	ich	k
͡dʒ	dZ	stimmhafte postalveolare Affrikate	Dschungel	S
f	f	stimmloser labiodentaler Frikativ	Vogel	f
g	g	stimmhafter velarer Plosiv	Gabel	k
h	h	stimmloser glottaler Frikativ	Haus	k

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
j	j	stimmloser glottaler Frikativ	jemand	i
k	k	stimmloser velarer Plosiv	Kleid	k
l	l	alveolarer lateraler Approximant	Loch	t
m	m	bilabialer Nasal	Milch	p
n	n	alveolarer Nasal	Natur	t
ŋ	N	velarer Nasal	klingen	k
p	p	stimmloser bilabialer Plosiv	Park	p
‿pf	pf	stimmlose labiodentale Affrikate	Apfel	
ʀ	R	uvularer Vibrant	Regen	
S	s	stimmlose r alveolarer Reibelaut	Messer	S
ʃ	S	stimmloser postalveolarer Frikativ	Fischer	S
t	t	stimmloser alveolarer Plosiv	Topf	T
‿ts	Ts	stimmlose alveolare Affrikate	Zahl	

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
͡tʃ	tS	stimmlose postalveolare Affrikate	deutsch	S
V	V	stimmhafter labiodentaler Frikativ	Wasser	f
x	x	stimmloser velarer Frikativ	kochen	k
z	z	stimmhafter alveolarer Frikativ	See	s
ʒ	Z	stimmhafter postalveolarer Frikativ	Orange	S
Vokale				
øː	2:	langer gerundeter halbgeschlossener Vorderzungenvokal	böse	o
ɐ	6	fast offener Zentralvokal	besser	a
ɐ̯	6_^	nicht silbischer fast offener Zentralvokal	Klar	a
œ	9	gerundeter halboffener Vorderzungenvokal	können	O
ə	@	Schwa	Rede	@

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
a	a	ungerundeter offener Vor	Salz	a
a:	a:	langer ungerunde ter offener Vorderzungenvokal	Sahne	a
aɪ	aI	Diphthong	nein	a
aʊ	aU	Diphthong	Augen	a
ɑ̃	A~	nasaler offener ungerundeter Hinterzungenvokal	Restaurant	a
e:	e:	langer ungerunde ter halbgesch lossener Vorderzun genvokal	Rede	e
ɛ	E	ungerundeter halboffener Vorderzungenvokal	Keller	E
ɛ̃	E~	nasaler ungerunde ter halboffener Vorderzungenvokal	Terrain	E
i:	i:	langer ungerunde ter geschlossener Vorderzungenvokal	Lied	i
ɪ	I	ungerundeter zentralisierter fast geschlossener Vorderzungenvokal	bitte	i

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
oː	o:	langer gerundeter halbgeschlossener Hinterzungenvokal	Kohl	o
ɔ	O	gerundeter halboffener Hinterzungenvokal	Koffer	O
ɔ̃	O~	nasaler gerundeter halboffener Hinterzungenvokal	Annonce	O
ɔʏ	OY	Diphthong	neu	O
uː	u:	langer gerundeter geschlossener hinterer Vokal	Bruder	u
ʊ	U	gerundeter zentralisierter fast geschlossener Hinterzungenvokal	Wunder	u
yː	y:	langer gerundeter geschlossener Vorderzungenvokal	kühl	u
ʏ	Y	gerundeter zentralisierter fast geschlossener Vorderzungenvokal	Küche	u
Zusätzliche Symbole				
ˈ	"	Hauptakzent	Alabama	
ˌ	%	Nebenakzent	Alabama	

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
.	.	Silbengrenze	A.la.ba.ma	

Deutsch (Österreichisch) (de-AT)

In der folgenden Tabelle sind die Phoneme des Internationalen Phonetischen Alphabets (IPA), die X-SAMPA-Symbole (Extended Speech Assessment Methods Phonetic Alphabet) und die entsprechenden Viseme für die österreichischen deutschen Stimmen aufgeführt, die von Amazon Polly unterstützt werden.

Tabelle der Phoneme/Viseme

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
Konsonanten				
ʔ	?	Glottallaut		
b	b	stimmhafter bilabialer Verschlus slaut	Bier	p
d	d	stimmhaft er alveolarer Verschlusslaut	Dach	t
ç	C	stimmloser palataler Frikativ	ich	k
͡dʒ	dZ	stimmhafte postalveolare Affrikate	Dschungel	S
f	f	stimmloser labiodentaler Frikativ	Vogel	f

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
g	g	stimmhafter velarer Plosiv	Gabel	k
h	h	stimmloser glottaler Frikativ	Haus	k
j	j	stimmloser glottaler Frikativ	jemand	i
k	k	stimmloser velarer Plosiv	Kleid	k
l	l	alveolarer lateraler Approximant	Loch	t
m	m	bilabialer Nasal	Milch	p
n	n	alveolarer Nasal	Natur	t
ŋ	N	velarer Nasal	klingen	k
p	p	stimmloser bilabialer Plosiv	Park	p
p͡f	pf	stimmlose labiodentale Affrikate	Apfel	
ʀ	R	uvularer Vibrant	Regen	
S	s	stimmlose r alveolarer Reibelaut	Messer	S
ʃ	S	stimmloser postalveolarer Frikativ	Fischer	S

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
t	t	stimmloser alveolarer Plosiv	Topf	T
͡ts	Ts	stimmlose alveolare Affrikate	Zahl	
͡tʃ	tS	stimmlose postalveolare Affrikate	deutsch	S
V	V	stimmhafter labiodentaler Frikativ	Wasser	f
x	x	stimmloser velarer Frikativ	kochen	k
z	z	stimmhafter alveolarer Frikativ	See	s
ʒ	Z	stimmhafter postalveolarer Frikativ	Orange	S
Vokale				
øː	2:	langer gerundeter halbgeschlossener Vorderzungenvokal	böse	o
ɐ	6	fast offener Zentralvokal	besser	a
ɐ̯	6_^	nicht silbischer fast offener Zentralvokal	Klar	a

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
œ	9	gerundeter halboffener Vorderzungenvokal	können	O
ə	@	Schwa	Rede	@
a	a	ungerundeter offener Vor	Salz	a
a:	a:	langer ungerunde ter offener Vorderzungenvokal	Sahne	a
aɪ	aI	Diphthong	nein	a
aʊ	aU	Diphthong	Augen	a
ɑ̃	A~	nasaler offener ungerundeter Hinterzungenvokal	Restaurant	a
e:	e:	langer ungerunde ter halbgesch lossener Vorderzun genvokal	Rede	e
ɛ	E	ungerundeter halboffener Vorderzungenvokal	Keller	E
ɛ̃	E~	nasaler ungerunde ter halboffener Vorderzungenvokal	Terrain	E
i:	i:	langer ungerunde ter geschlossener Vorderzungenvokal	Lied	i

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ɪ	I	ungerundeter zentralisierter fast geschlossener Vorderzungenvokal	bitte	i
o:	o:	langer gerundeter halbgeschlossener Hinterzungenvokal	Kohl	o
ɔ	O	gerundeter halboffener Hinterzungenvokal	Koffer	O
ɔ̃	O~	nasaler gerundeter halboffener Hinterzungenvokal	Annonce	O
ɔʏ	OY	Diphthong	neu	O
u:	u:	langer gerundeter geschlossener hinterer Vokal	Bruder	u
ʊ	U	gerundeter zentralisierter fast geschlossener Hinterzungenvokal	Wunder	u
y:	y:	langer gerundeter geschlossener Vorderzungenvokal	kühl	u
ʏ	Y	gerundeter zentralisierter fast geschlossener Vorderzungenvokal	Küche	u

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
Zusätzliche Symbole				
ˈ	"	Hauptakzent	Alabama	
ˌ	%	Nebenakzent	Alabama	
.	.	Silbengrenze	A.la.ba.ma	

Deutsch (Schweizer Standard) (de-CH)

In der folgenden Tabelle sind die Phoneme des Internationalen Phonetischen Alphabets (IPA), die X-SAMPA-Symbole (Extended Speech Assessment Methods Phonetic Alphabet) und die entsprechenden Viseme für die deutsche (Schweizer Standard-) Stimme aufgeführt, die von Amazon Polly unterstützt wird.

Tabelle der Phoneme/Viseme

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
Konsonanten				
ʔ	?	Glottallaut		
b	b	stimmhafter bilabialer Verschlus slaut	Bier	p
d	d	stimmhaft er alveolarer Verschlusslaut	Dach	t
ç	C	stimmloser palataler Frikativ	ich	k
͡dʒ	dZ	stimmhafte postalveolare Affrikate	Dschungel	S

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
f	f	stimmloser labiodentaler Frikativ	Vogel	f
g	g	stimmhafter velarer Plosiv	Gabel	k
h	h	stimmloser glottaler Frikativ	Haus	k
j	j	stimmloser glottaler Frikativ	jemand	i
k	k	stimmloser velarer Plosiv	Kleid	k
l	l	alveolarer lateraler Approximant	Loch	t
m	m	bilabialer Nasal	Milch	p
n	n	alveolarer Nasal	Natur	t
ŋ	N	velarer Nasal	klingen	k
p	p	stimmloser bilabialer Plosiv	Park	p
p͡f	pf	stimmlose labiodentale Affrikate	Apfel	
ʀ	R	uvularer Vibrant	Regen	
S	s	stimmlose r alveolarer Reibelaut	Messer	S

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ʃ	S	stimmloser postalveolarer Frikativ	Fischer	S
t	t	stimmloser alveolarer Plosiv	Topf	T
͡ts	Ts	stimmlose alveolare Affrikate	Zahl	
͡tʃ	tS	stimmlose postalveolare Affrikate	deutsch	S
V	V	stimmhafter labiodentaler Frikativ	Wasser	f
x	x	stimmloser velarer Frikativ	kochen	k
z	z	stimmhafter alveolarer Frikativ	See	s
ʒ	Z	stimmhafter postalveolarer Frikativ	Orange	S
Vokale				
øː	2:	langer gerundeter halbgeschlossener Vorderzungenvokal	böse	o
ɐ	6	fast offener Zentralvokal	besser	a

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ɐ̯	6_^	nicht silbischer fast offener Zentralvokal	Klar	a
œ	9	gerundeter halboffener Vorderzungenvokal	können	O
ə	@	Schwa	Rede	@
a	a	ungerundeter offener Vor	Salz	a
a:	a:	langer ungerunder offener Vorderzungenvokal	Sahne	a
aɪ	aI	Diphthong	nein	a
aʊ	aU	Diphthong	Augen	a
ɑ̃	A~	nasaler offener ungerundeter Hinterzungenvokal	Restaurant	a
e:	e:	langer ungerunder halbgeschlossener Vorderzungenvokal	Rede	e
ɛ	E	ungerundeter halboffener Vorderzungenvokal	Keller	E
ɛ̃	E~	nasaler ungerunder halboffener Vorderzungenvokal	Terrain	E

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
iː	i:	langer ungerundeter geschlossener Vorderzungenvokal	Lied	i
ɪ	I	ungerundeter zentralisierter fast geschlossener Vorderzungenvokal	bitte	i
oː	o:	langer gerundeter halbgeschlossener Hinterzungenvokal	Kohl	o
ɔ	O	gerundeter halboffener Hinterzungenvokal	Koffer	O
ɔ̃	O~	nasaler gerundeter halboffener Hinterzungenvokal	Annonce	O
ɔʏ	OY	Diphthong	neu	O
uː	u:	langer gerundeter geschlossener hinterer Vokal	Bruder	u
ʊ	U	gerundeter zentralisierter fast geschlossener Hinterzungenvokal	Wunder	u
yː	y:	langer gerundeter geschlossener Vorderzungenvokal	kühl	u

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ʏ	Y	gerundeter zentralisierter fast geschlossener Vorderzungenvokal	Küche	u
Zusätzliche Symbole				
ˈ	“	Hauptakzent	Alabama	
ˌ	%	Nebenakzent	Alabama	
.	.	Silbengrenze	A.la.ba.ma	

Hindi (hi-IN)

In der folgenden Tabelle sind die Phoneme des Internationalen Phonetischen Alphabets (IPA), die X-SAMPA-Symbole (Extended Speech Assessment Methods Phonetic Alphabet) und der Klangtyp des Phonems für die Hindi-Stimmen aufgeführt, die von Amazon Polly unterstützt werden.

Zusätzliche Phoneme in Verbindung mit Hindi finden Sie unter Englisch (indisch) (en-IN).

Tabelle der Phoneme/Viseme

IPA	X-SAMPA	Beschreibung	Beispiel
Konsonanten			
pʰ	p_h	stimmloser aspirierter bilabialer Plosiv	फूल (phool)
bʱ	b_h	stimmhafter aspirierter bilabialer Plosiv	भारी (bhaari)
t̪	t_d	stimmloser dentaler Plosiv	तापमान (taapmaan)

IPA	X-SAMPA	Beschreibung	Beispiel
t̪ʰ	t_d_h	stimmloser aspirierter dentaler Plosiv	थोड़ा (thoda)
d̪	d_d	stimmhafter dentaler Plosiv	दल्ली (dilli)
d̪ʱ	d_d_h	stimmhafter aspirierter dentaler Plosiv	धोबी (dhobi)
ʈ	t`	stimmloser retroflexer Plosiv	कटोरा (katora)
ʈʰ	t`_h	stimmloser aspirierter retroflexer Plosiv	ठंड (thand)
ɖ	d`	stimmhafter retroflexer Verschlusslaut	डर (darr)
ɖʱ	d`_h	stimmhafter aspirierter retroflexer Plosiv	ढाल (dhal)
tʃʰ	tS_h	stimmlose aspirierte palatale Affrikate	छाल (chaal)
dʒʱ	dZ_h	stimmhafte aspirierte palatale Affrikate	झाल (jhaal)
kʰ	k_h	stimmloser aspirierter velarer Plosiv	खान (khan)
gʱ	g_h	stimmhafter aspirierter velarer Plosiv	घान (ghaan)
ɳ	n`	retroflexer Nasal	क्षण (kshan)
ɾ	4	alveolarer Tap	राम (ram)
ɽ	r`	einfacher retroflexer Flap	बड़ा (bada)

IPA	X-SAMPA	Beschreibung	Beispiel
ɽʱ	r`_h	stimmhafter aspirierter retroflexer Flap	बढ़ी (barhi)
ʋ	V\	bilabialer Approximant	वसूल (wasool)
Vokale			
ə	@_o	Schwa	अच्छा (achhaa)
ə̃	@~	nasalisierter Mittelzun genvokal	हँसना (hansnaa)
a	A_o	ungerundeter offener Vor	आग (aag)
ã	A~	nasalisierter ungerunde ter offener Vorderzun genvokal	घ़डयिँा (ghariyaan)
ɪ	I_o	ungerundeter zentralis ierter fast geschlossener Vorderzungenvokal	इक्कीस (ikkees)
ɪ̃	I~	nasalisierter ungerunde ter fast geschlossener Vorderzungenvokal	संचिाई (sinchai)
i	i_o	ungerundeter geschloss ener Vorderzungenvokal	बल्िली (billee)
ĩ	i~	nasalisierter ungerunde ter geschlossener Vorderzungenvokal	नही (nahin)
ʊ	U_o	gerundeter zentralis ierter fast geschlossener Hinterzungenvokal	उलूल (ullu)

IPA	X-SAMPA	Beschreibung	Beispiel
ʊ̃	U~	nasalisierter gerundeter fast geschlossener Hinterzungenvokal	मुँह (munh)
u	u_o	gerundeter geschlossener Hinterzungenvokal	फूल (phool)
ũ	u~	nasalisierter gerundeter geschlossener Hinterzungenvokal	ऊँट (oont)
ɔ	O_o	gerundeter halboffener Hinterzungenvokal	कौन (kaun)
ɔ̃	O~	nasalisierter gerundeter halboffener Hinterzungenvokal	भौ (bhaun)
o	o	gerundeter halbgeschlossener Hinterzungenvokal	सोना (sona)
õ	o~	nasalisierter gerundeter halbgeschlossener Hinterzungenvokal	क्यो (kyon)
ɛ	E_o	ungerundeter halboffener Vorderzungenvokal	पैसा (paisa)
ɛ̃	E~	nasalisierter ungerundeter halboffener Vorderzungenvokal	मैं (main)
e	e	ungerundeter halbgeschlossener Vorderzungenvokal	एक (ek)

IPA	X-SAMPA	Beschreibung	Beispiel
ẽ	e~	nasalisierter ungerunde ter halbgeschlossener Vorderzungenvokal	कितोबें (kitabein)

Isländisch (is-IS)

In der folgenden Tabelle sind die Phoneme des Internationalen Phonetischen Alphabets (IPA), die X-SAMPA-Symbole (Extended Speech Assessment Methods Phonetic Alphabet) und die entsprechenden Viseme für die isländischen Stimmen aufgeführt, die von Amazon Polly unterstützt werden.

Tabelle der Phoneme/Viseme

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
Konsonanten				
b	b	stimmhafter bilabialer Verschlus slaut	grasbakkanum	0
c	c	stimmloser palataler Plosiv	pakkin	k
c^h	c_h	aspirierter stimmloser palataler Plosiv	anarkistai	k
ç	C	stimmloser palataler Frikativ	héðan	k
d	d	stimmhaft er alveolarer Verschlusslaut	bóndi	t

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ð	D	stimmhafter dentaler Frikativ	borð	T
f	f	stimmloser labiodentaler Frikativ	duft	f
g	g	stimmhafter velarer Verschlusslaut	holgóma	k
ɣ	G	stimmhafter velarer Frikativ	hugur	k
h	h	stimmloser glottaler Frikativ	heili	k
j	j	palataler Approximant	jökull	i
k^h	k_h	aspirierter stimmloser velarer Plosiv	ósköpunum	k
l	l	alveolarer lateraler Approximant	gólf	t
l̥	l_0	stimmloser alveolarer lateraler Approximant	fólk	t
m	m	bilabialer Nasal	september	p
m̥	m_0	stimmloser bilabialer Nasal	kompa	p
n	n	alveolarer Nasal	númer	t

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
n̥	n_0	stimmloser alveolarer Nasal	pöntun	t
ɲ	J	palataler Nasal	pælingar	J
ŋ	N	velarer Nasal	söngvarann	k
ŋ̊	N_0	stimmloser velarer Nasal	frænka	k
pʰ	p_h	aspirierter stimmloser bilabialer Plosiv	afplánun	p
r	r	alveolarer Vibrant	afskrifta	r
r̥	r_0	stimmloser alveolarer Vibrant	andvörpum	r
s	s	stimmlose r alveolarer Reibelaut	baðhús	S
tʰ	t_h	aspirierter stimmloser alveolarer Plosiv	tanki	t
θ	T	stimmloser dentaler Reibelaut	þeldökki	T
V	V	stimmhafter labiodentaler Reibelaut	silfur	f
w	w	labialisierter velarer Approximant		u

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
x	x	stimmloser velarer Reibelaut	samfélags	k
Vokale				
œ	9	gerundeter halboffener Vorderzungenvokal	þröskuldinum	O
œ:	9:	langer gerundeter halboffener Vorderzungenvokal	tvö	O
a	a	ungerundeter offener Vor	nefna	a
a:	a:	langer ungerundeter offener Vorderzungenvokal	fara	a
au	au	Diphthong	átta	a
au:	au:	Diphthong	átján	a
ɛ	E	ungerundeter halboffener Vorderzungenvokal	kennari	E
ɛ:	E:	langer ungerundeter halboffener Vorderzungenvokal	dreka	E
i	i	ungerundeter geschlossener Vorderzungenvokal	Gúlíver	i

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
iː	i:	langer ungerunde ter geschlossener Vorderzungenvokal	þrír	i
ɪ	I	ungerundeter zentralisierter fast geschlossener Vorderzungenvokal	samspil	i
ɪː	I:	langer ungerunde ter zentralisierter fast geschlossener Vorderzungenvokal	stig	i
ɔ	O	gerundeter halboffener Hinterzungenvokal	regndropar	O
ɔː	O:	langer gerundeter halboffener hinterer Vokal	ullarbolur	O
ɔu	Ou	Diphthong	tólf	O
ɔuː	Ou:	Diphthong	fjórir	O
u	u	gerundeter geschlossener Hinterzungenvokal	stúlkan	u
uː	u:	langer gerundete r geschlossener hinterer Vokal	frú	u

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ʏ	Y	gerundeter zentralisierter fast geschlossener Vorderzungenvokal	tíu	u
ʏː	Y	langer gerundeter zentralisierter fast geschlossener Vorderzungenvokal	gruninn	u
Zusätzliche Symbole				
ˈ	"	Hauptakzent	Alabama	
ˌ	%	Nebenakzent	Alabama	
.	.	Silbengrenze	A.la.ba.ma	

Italienisch (it-IT)

In der folgenden Tabelle sind die Phoneme des Internationalen Phonetischen Alphabets (IPA), die X-SAMPA-Symbole (Extended Speech Assessment Methods Phonetic Alphabet) und die entsprechenden Viseme für die italienischen Stimmen aufgeführt, die von Amazon Polly unterstützt werden.

Tabelle der Phoneme/Viseme

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
Konsonanten				
b	b	stimmhafter bilabialer Verschlusslaut	bacca	p

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
d	d	stimmhaft er alveolarer Verschlusslaut	dama	t
͡dz	dz	stimmhafte alveolare Affrikate	zero	S
͡dʒ	dZ	stimmhafte postalveolare Affrikate	giro	S
f	f	stimmloser labiodentaler Frikativ	famiglia	f
g	g	stimmhafter velarer Verschlusslaut	gatto	k
h	h	stimmloser glottaler Frikativ	horror	k
j	j	palataler Approximant	dieci	i
k	k	stimmloser velarer Verschlusslaut	campo	k
l	l	alveolarer lateraler Approximant	lido	t
ʎ	L	palataler lateraler Approximant	aglio	J
m	m	bilabialer Nasal	mille	p
n	n	alveolarer Nasal	nove	t
ɲ	J	palataler Nasal	lasagne	J

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
p	p	stimmloser bilabialer Verschlusslaut	pizza	p
r	r	alveolarer Vibrant	risata	r
s	s	stimmloser alveolarer Reibelaut	sei	S
ʃ	S	stimmloser postalveolarer Reibelaut	scienza	S
t	t	stimmloser alveolarer Verschlusslaut	tavola	t
͡ts	ts	stimmlose alveolare Affrikate	forza	S
͡tʃ	tS	stimmlose postalveolare Affrikate	cielo	S
V	V	stimmhafter labiodentaler Reibelaut	venti	f
w	w	labiovelarer Approximant	quattro	u
z	z	stimmhafter alveolarer Reibelaut	bisogno	s

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ʒ	Z	stimmhafter postalveolarer Frikativ	bijou	S
Vokale				
a	a	ungerundeter offener Vor	arco	a
e	e	ungerundeter halbgeschlossener Vorderzungenvokal	tre	e
ɛ	E	ungerundeter halboffener Vorderzungenvokal	ettaro	E
i	i	ungerundeter geschlossener Vorderzungenvokal	impero	i
o	o	gerundeter halbgeschlossener Hinterzungenvokal	cento	o
ɔ	O	gerundeter halboffener Hinterzungenvokal	otto	O
u	u	gerundeter geschlossener Hinterzungenvokal	uno	u
Zusätzliche Symbole				
ˈ	"	Hauptakzent	Alabama	

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ˌ	%	Nebenakzent	Alabama	
.	.	Silbengrenze	A.la.ba.ma	

Japanisch (ja-JP)

Amazon Polly unterstützt die Alphabete Pronunciation Kana und Yomigana für Japanisch. Verwenden Sie das Phonem-Attribut, damit Amazon Polly die phonetische Aussprache für diese Alphabete verwendet. `alphabet="x-amazon-`*`phonetic standard used`*`"`

- `x-amazon-pron-kana`— gibt an, dass die Aussprache Kana verwendet wird. Aussprache Kana sind spezielle Katakana-Zeichen, die für die phonetische Transkription verwendet werden und Tonhöhenakzente kodieren können.
- `x-amazon-yomigana`— weist darauf hin, dass Yomigana verwendet wird. Bei Yomigana kann es sich um konventionelle Katakana-, Hiragana- und lateinische Alphabete handeln, die als Hepburn-Romanisierung interpretiert werden.

Die folgenden Beispiele zeigen, wie diese verwendet werden:

Aussprache: Kana

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

Yomigana

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

In der folgenden Tabelle sind die Phoneme des Internationalen Phonetischen Alphabets (IPA), die X-SAMPA-Symbole (Extended Speech Assessment Methods Phonetic Alphabet) und die

entsprechenden Viseme für die japanische Stimme aufgeführt, die von Amazon Polly unterstützt werden.

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
Konsonanten				
ɾ	4	alveolarer Tap	練習, renshuu	t
ʔ	?	Glottallaut	あつっ, atsu'	
b	b	stimmhafter bilabialer Verschlus slaut	舞踊, buyou	p
β	B	stimmhafter bilabialer Frikativ	ヴィンテージ, vinteeji	B
c	c	stimmloser palataler Plosiv	ききょう, kikyou	k
ç	C	stimmloser palataler Frikativ	人, hito	k
d	d	stimmhaft er alveolarer Verschlusslaut	濁点, dakuten	t
͡dʑ	dz\	stimmhafte alveolopalatale Affrikate	純, jun	J
g	g	stimmhafter velarer Verschlusslaut	ご飯, gohan	k
h	h	stimmloser glottaler Frikativ	本, hon	k
j	j	palataler Approximant	屋根, yane	i

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ɟ	J\	stimmhafter palataler Plosiv	行儀, gyougi	J
k	k	stimmloser velarer Verschlusslaut	漢字, kanji	k
ɺ	l\	alveolarer lateraler Tap	釣り, tsuri	r
ɺj	l\j	alveolarer lateraler Tap, palataler Approximant	流行, ryuukou	r
m	m	bilabialer Nasal	飯, meshi	p
n	n	alveolarer Nasal	猫, neko	t
ɲ	J	palataler Nasal	日本, nippon	J
ɴ	N\	uvularer Nasal	缶, kan	k
p	p	stimmloser bilabialer Verschlus slaut	パン, pan	p
ɸ	p\	stimmloser bilabialer Frikativ	福, huku	f
s	s	stimmlose r alveolarer Reibelaut	層, sou	S
ɕ	S\	stimmloser alveolopalataler Frikativ	書簡, shokan	J

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
t	t	stimmlose r alveolarer Verschlusslaut	手紙, tegami	t
͡ts	ts	stimmlose alveolare Affrikate	釣り, tsuri	S
͡tɕ	ts\	stimmlose alveolopalatale Affrikate	吉, kichi	J
w	w	labiovelarer Approximant	電話, denwa	u
z	z	stimmhaft er alveolarer Reibelaut	座敷, zashiki	S
Vokale				
ä:	a:_"	langer ungerunde ter offener Zentralvokal	羽蟻, haari	a
ä	a_"	ungerundeter offener Zentralvo kal	仮名, kana	a
e:	e:_o	langer mittlerer ungerundeter Vorderzungenvokal	学生, gakusei	@
e	e_o	ungerunderter mittlerer Vorderzun genvokal	歴, reki	@

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
i	i	ungerundeter geschlossener Vorderzungenvokal	気, ki	i
iː	i:	langer ungerunde ter geschlossener Vorderzungenvokal	詩歌, shiika	i
ɯ	Mio.	geschlossener ungerundeter Hinterzungenvokal	運, un	i
ɯː	M:	langer geschloss ener ungerundeter Hinterzungenvokal	宗教, shuukyou	i
oː	o:_o	langer gerundeter mittlerer Hinterzun genvokal	購読, koodoku	o
o	o_o	mittlerer gerundete r Hinterzun genvokal	読者, dokusha	o

Koreanisch (ko-KR)

In der folgenden Tabelle sind die Phoneme des Internationalen Phonetischen Alphabets (IPA), die X-SAMPA-Symbole (Extended Speech Assessment Methods Phonetic Alphabet) und die entsprechenden Viseme für die koreanische Stimme aufgeführt, die von Amazon Polly unterstützt werden.

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
Konsonanten				

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
k	k	stimmloser velarer Verschlusslaut	강, [g]ang	k
k#	k_t	starker stimmloser velarer Plosiv	깨, [kk]e	k
n	n	alveolarer Nasal	남, [n]am	t
t	t	stimmlose r alveolarer Verschlusslaut	도, [d]o	t
t#	t_t	starker stimmloser alveolarer Plosiv	때, [tt]e	t
ɾ	4	alveolarer Tap	사랑, sa[r]ang	t
l	l	alveolarer lateraler Approximant	돌, do[l]	t
m	m	bilabialer Nasal	무, [m]u	p
p	p	stimmloser bilabialer Verschlus slaut	봄, [b]om	p
p#	p_t	starker stimmloser bilabialer Plosiv	뻘, [pp]eol	p
s	s	stimmlose r alveolarer Reibelaut	새, [s]e	s
s#	s_t	starker stimmloser alveolarer Frikativ	씨, [ss]i	s
ŋ	N	velarer Nasal	방, ba[ng]	k

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
t͡ɕ	ts\	stimmlose alveolopalatale Affrikate	조, [j]o	J
t͡ɕ͈	ts_t	starke stimmlose alveolopalatale Affrikate	찌, [jj]i	J
t͡ɕʰ	ts_h	aspirierte stimmlose alveolopalatale Affrikate	차, [ch]a	J
kʰ	k_h	aspirierter stimmloser velarer Plosiv	코, [k]o	k
tʰ	t_h	aspirierter stimmloser alveolarer Plosiv	통, [t]ong	t
pʰ	p_h	aspirierter stimmloser bilabialer Plosiv	패, [p]e	p
h	h	stimmloser glottaler Frikativ	힘, [h]im	k
j	j	palataler Approximant	양, [y]ang	i
w	w	labiovelarer Approximant	왕, [w]ang	u
ɯɻ	M\	velarer Approximant>	의, [wj]i	i

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
Vokale				
a	a	ungerundeter offener Vor	밥, b[a]b	a
ʌ	V	ungerundeter halboffener Hinterzungenvokal	정, j[eo]ng	E
ɛ	E	ungerundeter halboffener Vorderzungenvokal	배, b[e]	E
o	o	gerundeter halbgeschlossener Hinterzungenvokal	노, n[o]	o
u	u	gerundeter geschlossener Hinterzungenvokal	둘, d[u]l	u
ɯ	M	geschlossener ungerundeter Hinterzungenvokal	은, [eu]n	i
i	i	ungerundeter geschlossener Vorderzungenvokal	김, k[i]m	i

Norwegisch (nb-NO)

In der folgenden Tabelle sind der vollständige Satz der Phoneme des Internationalen Phonetischen Alphabets (IPA) und der X-SAMPA-Symbole (Extended Speech Assessment Methods Phonetic Alphabet) sowie die entsprechenden Viseme aufgeführt, die von Amazon Polly für norwegische Stimmen unterstützt werden.

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
Konsonanten				
ɾ	4	alveolarer Tap	prøv	t
b	b	stimmhafter bilabialer Verschlusslaut	labb	p
ç	C	stimmloser palataler Frikativ	kino	k
d	d	stimmhafter alveolarer Verschlusslaut	ladd	t
ɖ	d`	stimmhafter retroflexer Verschlusslaut	verdi	t
f	f	stimmloser labiodentaler Frikativ	fot	f
g	g	stimmhafter velarer Verschlusslaut	tagg	k
h	h	stimmloser glottaler Frikativ	ha	k
j	j	palataler Approximant	gi	i
k	k	stimmloser velarer Verschlusslaut	takk	k
l	l	alveolarer lateraler Approximant	fall, ball	t

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ɭ	l`	lateraler retroflexer Approximant	ærlig	t
m	m	bilabialer Nasal	lam	p
n	n	alveolarer Nasal	vann	t
ɳ	n`	retroflexer Nasal	garn	t
ŋ	N	velarer Nasal	sang	k
p	p	stimmloser bilabialer Verschlusslaut	hopp	p
s	s	stimmlose r alveolarer Reibelaut	lass	S
ʂ	S`	stimmloser retroflexer Frikativ	års	S
ʃ	S	stimmloser postalveolarer Reibelaut	skyt	S
t	t	stimmlose r alveolarer Verschlusslaut	lat	t
ʈ	t`	stimmloser retroflexer Plosiv	hardt	t
ʊ	V\	labiodentaler Approximant	vin	f
w	w	labiovelarer Approximant	will	x

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
Vokale				
øː	2:	langer gerundeter halbgeschlossener Vorderzungenvokal	søt	o
œ	9	gerundeter halboffener Vorderzungenvokal	søtt	O
ə	@	Schwa	ape	@
æː	{:	langer ungerunde ter fast offener Vorderzungenvokal	vær	a
ʉ	}	gerundeter geschlossener Zentralvokal	lund	u
ʉː	}:	langer gerundete r geschlossener Zentralvokal	lun	u
æ	{	ungerundeter fast offener Vorderzun genvokal	vært	a
ɑ	A	ungerundeter offener Hinterzun genvokal	hatt	a
ɑː	A:	langer ungerunde ter offener Hinterzungenvokal	hat	a

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
eː	e:	langer ungerundeter halbgeschlossener Vorderzungenvokal	sen	e
ɛ	E	ungerundeter halboffener Vorderzungenvokal	send	E
iː	i:	langer ungerundeter geschlossener Vorderzungenvokal	vin	i
ɪ	I	ungerundeter zentralisierter fast geschlossener Vorderzungenvokal	vind	i
oː	o:	langer gerundeter halbgeschlossener Hinterzungenvokal	våt	o
ɔ	O	gerundeter halboffener Hinterzungenvokal	vått	O
uː	u:	langer gerundeter geschlossener hinterer Vokal	bok	u
ʊ	U	gerundeter zentralisierter fast geschlossener Hinterzungenvokal	bukk	u

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
y:	y:	langer gerundeter geschlossener Vorderzungenvokal	lyn	u
ʏ	Y	gerundeter zentralisierter fast geschlossener Vorderzungenvokal	lynne	u
Zusätzliche Symbole				
ˈ	"	Hauptakzent	Alabama	
ˌ	%	Nebenakzent	Alabama	
.	.	Silbengrenze	A.la.ba.ma	

Polnisch (pl-PL)

In der folgenden Tabelle sind die Phoneme des Internationalen Phonetischen Alphabets (IPA), die X-SAMPA-Symbole (Extended Speech Assessment Methods Phonetic Alphabet) und die entsprechenden Viseme für die polnischen Stimmen aufgeführt, die von Amazon Polly unterstützt werden.

Tabelle der Phoneme/Viseme

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
Konsonanten				
b	b	stimmhafter bilabialer Verschlusslaut	bobas, belka	p
d	d	stimmhafter alveolarer Verschlusslaut	dar, do	t

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
͡dz	dz	stimmhafte alveolare Affrikate	dzwon, widzowie	S
͡dʑ	dz\	stimmhafte alveolopalatale Affrikate	dźwięk	J
͡dʐ	dz`	stimmhafte retroflexe Affrikate	dżem, dżungla	S
f	f	stimmloser labiodentaler Frikativ	furtka, film	f
g	g	stimmhafter velarer Verschlusslaut	gazeta, waga	k
h	h	stimmloser glottaler Frikativ	chleb, handel	k
j	j	palataler Approximant	jak, maja	i
k	k	stimmloser velarer Verschlusslaut	kura, marek	k
l	l	alveolarer lateraler Approximant	lipa, alicja	t
m	m	bilabialer Nasal	matka, molo	p
n	n	alveolarer Nasal	norka	t
ɲ	J	palataler Nasal	koń, toruń	J
p	p	stimmloser bilabialer Verschlus slaut	pora, stop	p

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
r	r	alveolarer Vibrant	rok, park	r
s	s	stimmlose r alveolarer Reibelaut	sum, pas	S
ɕ	S\	stimmloser alveolopalataler Frikativ	śruba, śnieg	J
ʂ	S`	stimmloser retroflexer Frikativ	szum, masz	S
t	t	stimmlose r alveolarer Verschlusslaut	tok, stół	t
͡ts	ts	stimmlose alveolare Affrikate	car, co	S
͡tɕ	ts\	stimmlose alveolopalatale Affrikate	ćma, mieć	J
͡tʂ	ts`	stimmlose retroflexe Affrikate	czas, raczej	S
V	V	stimmhafter labiodentaler Reibelaut	worek, mewa	f
w	w	labiovelarer Approximant	łaska, mało	u
z	z	stimmhafter alveolarer Reibelaut	zero	S

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ʑ	z\	stimmhafter alveolopalataler Frikativ	źrebię, bieliźnie	J
ʐ	z`	stimmhafter retroflexer Frikativ	żar, żona	S
Vokale				
a	a	ungerundeter offener Vor	ja	a
ɛ	E	ungerundeter halboffener Vorderzungenvokal	echo	E
ɛ̃	E~	nasaler ungerunde ter halboffener Vorderzungenvokal	węże	E
i	i	ungerundeter geschlossener Vorderzungenvokal	ile	i
ɔ	O	gerundeter halboffener Hinterzungenvokal	oczy	O
ɔ̃	O~	nasaler gerundete r halboffener Hinterzungenvokal	wąż	O
u	u	gerundeter geschlossener Hinterzungenvokal	uczta	u

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ɨ	1	ungerundeter geschlossener Zentralvokal	byk	i
Zusätzliche Symbole				
ˈ	"	Hauptakzent	Alabama	
ˌ	%	Nebenakzent	Alabama	
.	.	Silbengrenze	A.la.ba.ma	

Portugiesisch (pt-PT)

In der folgenden Tabelle sind die Phoneme des Internationalen Phonetischen Alphabets (IPA), die X-SAMPA-Symbole (Extended Speech Assessment Methods Phonetic Alphabet) und die entsprechenden Viseme für die portugiesischen Stimmen aufgeführt, die von Amazon Polly unterstützt werden.

Tabelle der Phoneme/Viseme

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
Konsonanten				
ɾ	4	alveolarer Tap	pira	t
b	b	stimmhafter bilabialer Verschlus slaut	dato	p
d	d	stimmhaft er alveolarer Verschlusslaut	dato	t

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
f	f	stimmloser labiodentaler Frikativ	facto	f
g	g	stimmhafter velarer Verschlusslaut	gato	k
j	j	palataler Approximant	paraguay	i
k	k	stimmloser velarer Verschlusslaut	cacto	k
l	l	alveolarer lateraler Approximant	galo	t
ʎ	L	palataler lateraler Approximant	galho	J
m	m	bilabialer Nasal	mato	p
n	n	alveolarer Nasal	nato	t
ɲ	J	palataler Nasal	pinha	J
p	p	stimmloser bilabialer Verschlus slaut	pato	p
ʁ	R\	uvularer Vibrant	barroso	k
s	s	stimmlose r alveolarer Reibelaut	saca	S
ʃ	S	stimmloser postalveolarer Reibelaut	chato	S

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
t	t	stimmlose r alveolarer Verschlusslaut	tacto	t
V	V	stimmhafter labiodentaler Reibelaut	vaca	f
w	w	labiovelarer Approximant	mau	u
z	z	stimmhaft er alveolarer Reibelaut	zaca	s
ʒ	Z	stimmhafter postalveolarer Frikativ	jacto	S
Vokale				
a	a	ungerundeter offener Vor	parto	a
ã	a~	nasaler ungerunde ter offener Vorderzungenvokal	pega	a
e	e	ungerundeter halbgeschlossener Vorderzungenvokal	pega	e
ẽ	e~	nasaler ungerunde ter halbgesch lossener Vorderzun genvokal	movem	e

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ɛ	E	ungerundeter halboffener Vorderzungenvokal	café	E
i	i	ungerundeter geschlossener Vorderzungenvokal	lingueta	i
ĩ	i~	nasaler ungerunde ter geschlossener Vorderzungenvokal	cinto	i
o	o	gerundeter halbgeschlossener Hinterzungenvokal	poder	o
õ	o~	nasaler gerundeter halbgeschlossener Hinterzungenvokal	compra	o
ɔ	O	gerundeter halboffener Hinterzungenvokal	cotó	O
u	u	gerundeter geschlossener Hinterzungenvokal	fui	u
ũ	u~	nasaler gerundete r geschlossener Hinterzungenvokal	sunto	u
Zusätzliche Symbole				
ˈ	"	Hauptakzent	Alabama	
ˌ	%	Nebenakzent	Alabama	

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
.	.	Silbengrenze	A.la.ba.ma	

Portugiesisch (brasilianisch) (pt-BR)

In der folgenden Tabelle sind die Phoneme des Internationalen Phonetischen Alphabets (IPA), die X-SAMPA-Symbole (Extended Speech Assessment Methods Phonetic Alphabet) und die entsprechenden Viseme für die brasilianischen portugiesischen Stimmen aufgeführt, die von Amazon Polly unterstützt werden.

Tabelle der Phoneme/Viseme

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
Konsonanten				
ɾ	4	alveolarer Tap	pira	t
b	b	stimmhafter bilabialer Verschlus slaut	bato	p
d	d	stimmhaft er alveolarer Verschlusslaut	dato	t
ʤ	dZ	stimmhafte postalveolare Affrikate	idade	S
f	f	stimmloser labiodentaler Frikativ	facto	f
g	g	stimmhafter velarer Verschlusslaut	gato	k

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
j	j	palataler Approximant	paraguay	i
k	k	stimmloser velarer Verschlusslaut	cacto	k
l	l	alveolarer lateraler Approximant	galo	t
ʎ	L	palataler lateraler Approximant	galho	J
m	m	bilabialer Nasal	mato	p
n	n	alveolarer Nasal	nato	t
ɲ	J	palataler Nasal	pinha	J
p	p	stimmloser bilabialer Verschlus slaut	pato	p
s	s	stimmlose r alveolarer Reibelaut	saca	S
ʃ	S	stimmloser postalveolarer Reibelaut	chato	S
t	t	stimmlose r alveolarer Verschlusslaut	tacto	t
‿tʃ	tS	stimmlose postalveolare Affrikate	noite	S

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
v	V	stimmhafter labiodentaler Reibelaut	vaca	f
w	w	labiovelarer Approximant	mau	u
χ	X	stimmloser uvularer Frikativ	carro	k
z	z	stimmhafter alveolarer Reibelaut	zaca	s
ʒ	Z	stimmhafter postalveolarer Frikativ	jacto	S
Vokale				
a	a	ungerundeter offener Vor	parto	a
ã	a~	nasaler ungerundeter offener Vorderzungenvokal	pensamos	a
e	e	ungerundeter halbgeschlossener Vorderzungenvokal	pega	e
ẽ	e~	nasaler ungerundeter halbgeschlossener Vorderzungenvokal	movem	e

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ɛ	E	ungerundeter halboffener Vorderzungenvokal	café	E
i	i	ungerundeter geschlossener Vorderzungenvokal	lingueta	i
ĩ	i~	nasaler ungerunde ter geschlossener Vorderzungenvokal	cinto	i
o	o	gerundeter halbgeschlossener Hinterzungenvokal	poder	o
õ	o~	nasaler gerundeter halbgeschlossener Hinterzungenvokal	compra	o
ɔ	O	gerundeter halboffener Hinterzungenvokal	cotó	O
u	u	gerundeter geschlossener Hinterzungenvokal	fui	u
ũ	u~	nasaler gerundete r geschlossener Hinterzungenvokal	sunto	u
Zusätzliche Symbole				
ˈ	"	Hauptakzent	Alabama	
ˌ	%	Nebenakzent	Alabama	

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
.	.	Silbengrenze	A.la.ba.ma	

Rumänisch (ro-RO)

In der folgenden Tabelle sind die Phoneme des Internationalen Phonetischen Alphabets (IPA), die X-SAMPA-Symbole (Extended Speech Assessment Methods Phonetic Alphabet) und die entsprechenden Viseme für die rumänische Stimme aufgeführt, die von Amazon Polly unterstützt werden.

Tabelle der Phoneme/Viseme

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
Konsonanten				
b	b	stimmhafter bilabialer Verschlus slaut	bubă	p
d	d	stimmhaft er alveolarer Verschlusslaut	după	t
͡dʒ	dZ	stimmhafte postalveolare Affrikate	george	S
f	f	stimmloser labiodentaler Frikativ	afacere	f
g	g	stimmhafter velarer Verschlusslaut	agri#	k
h	h	stimmloser glottaler Frikativ	harpă	k

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
j	j	palataler Approximant	baie	i
k	k	stimmloser velarer Verschlusslaut	co#	k
l	l	alveolarer lateraler Approximant	lampa	t
m	m	bilabialer Nasal	mama	p
n	n	alveolarer Nasal	nor	t
p	p	stimmloser bilabialer Verschlus slaut	pilă	p
r	r	alveolarer Vibrant	rampă	r
s	s	stimmlose r alveolarer Reibelaut	soare	S
ʃ	S	stimmloser postalveolarer Reibelaut	ma#ină	S
t	t	stimmlose r alveolarer Verschlusslaut	tata	t
͡ts	ts	stimmlose alveolare Affrikate	#ară	S
͡tʃ	tS	stimmlose postalveolare Affrikate	ceai	S

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
v	V	stimmhafter labiodentaler Reibelaut	via#ă	f
w	w	labiovelarer Approximant	beau	u
z	z	stimmhafter alveolarer Reibelaut	mozol	s
ʒ	Z	stimmhafter postalveolarer Frikativ	joacă	S
Vokale				
ə	@	Schwa	babă	@
a	a	ungerundeter offener Vor	casa	a
e	e	ungerundeter halbgeschlossener Vorderzungenvokal	elan	e
e̯	e_^	nicht silbische r ungerundeter halbgeschlossener Vorderzungenvokal	beau	e
i	i	ungerundeter geschlossener Vorderzungenvokal	mie	i

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
o	o	gerundeter halbgeschlossener Hinterzungenvokal	ură	o
oa	o_^a	Diphthong	oare	o
u	u	gerundeter geschlossener Hinterzungenvokal	unde	u
ɨ	1	ungerundeter geschlossener Zentralvokal	România	i
Zusätzliche Symbole				
ˈ	"	Hauptakzent	Alabama	
ˌ	%	Nebenakzent	Alabama	
.	.	Silbengrenze	A.la.ba.ma	

Russisch (ru-RU)

In der folgenden Tabelle sind die Phoneme des Internationalen Phonetischen Alphabets (IPA), die X-SAMPA-Symbole (Extended Speech Assessment Methods Phonetic Alphabet) und die entsprechenden Viseme für die russischen Stimmen aufgeführt, die von Amazon Polly unterstützt werden.

Tabelle der Phoneme/Viseme

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
Konsonanten				

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
b	b	stimmhafter bilabialer Verschlusslaut	борт	p
b^j	b'	mouillierter stimmhafter bilabialer Plosiv	бюро	p
d	d	stimmhafter alveolarer Verschlusslaut	дом	t
d^j	d'	mouillierter stimmhafter alveolarer Plosiv	дядя	t
f	f	stimmloser labiodentaler Frikativ	флаг	f
f^j	f'	mouillierter stimmloser labiodentaler Frikativ	февраль	f
g	g	stimmhafter velarer Verschlusslaut	нога	k
g^j	g'	mouillierter stimmhafter velarer Plosiv	герой	k
j	j	palataler Approximant	дизайн, ящик	i
k	k	stimmloser velarer Verschlusslaut	кот	k

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
k^j	k'	mouillierter stimmloser velarer Plosiv	кино	k
l	l	alveolarer lateraler Approximant	лампа	t
l^j	l'	mouillierter alveolarer lateraler Approximant	лес	t
m	m	bilabialer Nasal	мама	p
m^j	i'	mouillierter bilabialer Nasal	мяч	p
n	n	alveolarer Nasal	нос	t
n^j	n'	mouillierter alveolarer Nasal	няня	t
p	p	stimmloser bilabialer Verschlus slaut	папа	p
p^j	p'	mouillierter stimmloser bilabialer Plosiv	перо	p
r	r	alveolarer Vibrant	роза	r
r^j	r'	mouillierter alveolarer Vibrant	рюмка	r
s	s	stimmlose r alveolarer Reibelaut	сыр	S

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
sʲ	S'	mouillierter stimmloser alveolarer Frikativ	сердце, русь	S
ɕː	S\:	langer stimmlose r alveolopalataler Frikativ	щека	J
ʂ	S`	stimmloser retroflexer Frikativ	шум	S
t	t	stimmlose r alveolarer Verschlusslaut	точка	t
tʲ	t'	mouillierter stimmloser alveolarer Plosiv	тётя	t
͡ts	ts	stimmlose alveolare Affrikate	царь	S
͡tɕ	ts\	stimmlose alveolopalatale Affrikate	час	J
V	V	stimmhafter labiodentaler Reibelaut	вор	f
vʲ	V'	mouillierter stimmhafter labiodentaler Frikativ	верфь	f
x	x	stimmloser velarer Reibelaut	хор	k

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
xʲ	x'	mouillierter stimmloser velarer Frikativ	химия	k
z	z	stimmhaft er alveolarer Reibelaut	зуб	S
zʲ	z'	mouillierter stimmhafter alveolarer Frikativ	зима	S
ʑː	z\:	langer stimmhaft er alveolopalataler Frikativ	уезжать	J
ʐ	z`	stimmhafter retroflexer Frikativ	жена	S
Vokale				
ə	@	Schwa	канарейка	@
a	a	ungerundeter offener Vor	два, яблоко	a
e	e	ungerundeter halbgeschlossener Vorderzungenvokal	печь	e
ɛ	E	ungerundeter halboffener Vorderzungenvokal	это	E
i	i	ungerundeter geschlossener Vorderzungenvokal	один, четыре	i

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
o	o	gerundeter halbgeschlossener Hinterzungenvokal	кот	o
u	u	gerundeter geschlossener Hinterzungenvokal	муж, вьюга	u
ɨ	1	ungerundeter geschlossener Zentralvokal	мышь	i

Spanisch (es-ES)

In der folgenden Tabelle sind die Phoneme des Internationalen Phonetischen Alphabets (IPA), die X-SAMPA-Symbole (Extended Speech Assessment Methods Phonetic Alphabet) und die entsprechenden Viseme für die spanischen Stimmen aufgeführt, die von Amazon Polly unterstützt werden.

Tabelle der Phoneme/Viseme

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
Konsonanten				
ɾ	4	alveolarer Tap	pero, bravo, amor, eterno	t
b	b	stimmhafter bilabialer Verschlus slaut	bestia	p
β	B	stimmhafter bilabialer Frikativ	bebé	B

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
d	d	stimmhaft er alveolarer Verschlusslaut	cuando	t
ð	D	stimmhafter dentaler Frikativ	arder	T
f	f	stimmloser labiodentaler Frikativ	fase, café	f
g	g	stimmhafter velarer Verschlusslaut	gato, lengua, guerra	k
ɣ	G	stimmhafter velarer Frikativ	trigo, Argos	k
j	j	palataler Approximant	hacia, tierra, radio, viuda	i
ʝ	j\	stimmhafter palataler Frikativ	enhielar, sayo, inyectado, desyerba	J
k	k	stimmloser velarer Verschlusslaut	caña, laca, quisimos	k
l	l	alveolarer lateraler Approximant	lino, calor, principal	t
ʎ	L	palataler lateraler Approximant	llave, pollo	J
m	m	bilabialer Nasal	madre, comer, anfibio	p
n	n	alveolarer Nasal	nido, anillo, sin	t

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ɲ	J	palataler Nasal	cabaña, ñoquis	J
ŋ	N	velarer Nasal	cinco, venga	k
p	p	stimmloser bilabialer Verschlusslaut	pozo, topo	p
r	r	alveolarer Vibrant	perro, enrachado	r
s	s	stimmlose r alveolarer Reibelaut	saco, casa, puertas	S
t	t	stimmlose r alveolarer Verschlusslaut	tamiz, átomo	t
͡tʃ	tS	stimmlose postalveolare Affrikate	chubasco	S
θ	T	stimmloser dentaler Reibelaut	cereza, zorro, lacero, paz	T
w	w	labiovelarer Approximant	fuego, fuimos, cuota, cuadro	u
x	x	stimmloser velarer Reibelaut	jamón, general, suje, reloj	k
z	z	stimmhafter alveolarer Reibelaut	rasgo, mismo	S
Vokale				

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
a	a	ungerundeter offener Vor	tanque	a
e	e	ungerundeter halbgeschlossener Vorderzungenvokal	peso	e
i	i	ungerundeter geschlossener Vorderzungenvokal	cinco	i
o	o	gerundeter halbgeschlossener Hinterzungenvokal	bosque	o
u	u	ungerundeter halbgeschlossener Vorderzungenvokal	publicar	u
Zusätzliche Symbole				
ˈ	"	Hauptakzent	Alabama	
ˌ	%	Nebenakzent	Alabama	
.	.	Silbengrenze	A.la.ba.ma	

Spanisch (mexikanisch) (es-MX)

In der folgenden Tabelle sind die Phoneme des Internationalen Phonetischen Alphabets (IPA), die X-SAMPA-Symbole (Extended Speech Assessment Methods Phonetic Alphabet) und die entsprechenden Viseme für die mexikanische spanische Stimme aufgeführt, die von Amazon Polly unterstützt wird.

Tabelle der Phoneme/Viseme

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
Konsonanten				
ɾ	4	alveolarer Tap	pero, bravo, amor, eterno	t
b	b	stimmhafter bilabialer Verschlus slaut	bestia	p
β	B	stimmhafter bilabialer Frikativ	bebé	B
d	d	stimmhaft er alveolarer Verschlusslaut	cuando	t
ð	D	stimmhafter dentaler Frikativ	arder	T
f	f	stimmloser labiodentaler Frikativ	fase, café	f
g	g	stimmhafter velarer Verschlusslaut	gato, lengua, guerra	k
ɣ	G	stimmhafter velarer Frikativ	trigo, Argos	k
j	j	palataler Approximant	hacia, tierra, radio, viuda	i
ʝ	j\	stimmhafter palataler Frikativ	enhielar, sayo, inyectado, desyerba	J

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
k	k	stimmloser velarer Verschlusslaut	caña, laca, quisimos	k
l	l	lateraler alveolarer Approximant	lino, calor, principal	t
m	m	bilabialer Nasal	madre, comer, anfibio	p
n	n	alveolarer Nasal	nido, anillo, sin	t
ɲ	J	palataler Nasal	cabaña, ñoquis	J
ŋ	N	velarer Nasal	angosto, increíble	k
p	p	stimmloser bilabialer Verschlus slaut	pozo, topo	p
r	r	alveolarer Vibrant	perro, enrachado	r
s	s	stimmlose r alveolarer Reibelaut	saco, casa, puertas	S
ʃ	S	stimmloser postalveolarer Reibelaut	show, flash	S
t	t	stimmlose r alveolarer Verschlusslaut	tamiz, átomo	t
͡tʃ	tS	stimmlose postalveolare Affrikate	chubasco	S

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
w	w	labiovelarer Approximant	fuego, fuimos, cuota, cuadro	u
x	x	stimmloser velarer Reibelaut	jamón, general, peaje, reloj	k
z	z	stimmhafter alveolarer Reibelaut	rasgo, mismo	S
Vokale				
a	a	ungerundeter offener Zentralvokal	tanque	a
e	e	ungerundeter halbgeschlossener Vorderzungenvokal	peso	e
i	i	ungerundeter geschlossener Vorderzungenvokal	cinco	i
o	o	gerundeter halbgeschlossener Hinterzungenvokal	bosque	o
u	u	gerundeter geschlossener Hinterzungenvokal	publicar	u
Zusätzliche Symbole				
ˈ	"	Hauptakzent	Alabama	
ˌ	%	Nebenakzent	Alabama	

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
.	.	Silbengrenze	A.la.ba.ma	

Spanisch (USA) (es-US)

In der folgenden Tabelle sind die Phoneme des Internationalen Phonetischen Alphabets (IPA), die X-SAMPA-Symbole (Extended Speech Assessment Methods Phonetic Alphabet) und die entsprechenden Viseme für die amerikanischen spanischen Stimmen aufgeführt, die von Amazon Polly unterstützt werden.

Tabelle der Phoneme/Viseme

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
Konsonanten				
ɾ	4	alveolarer Tap	pero, bravo, amor, eterno	t
b	b	stimmhafter bilabialer Verschlus slaut	bestia	p
β	B	stimmhafter bilabialer Frikativ	bebé	B
d	d	stimmhaft er alveolarer Verschlusslaut	cuando	t
ð	D	stimmhafter dentaler Frikativ	arder	T
f	f	stimmloser labiodentaler Frikativ	fase, café	f

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
g	g	stimmhafter velarer Verschlusslaut	gato, lengua, guerra	k
ɣ	G	stimmhafter velarer Frikativ	trigo, Argos	k
j	j	palataler Approximant	hacia, tierra, radio, viuda	i
ʝ	j\	stimmhafter palataler Frikativ	enhielar, sayo, inyectado, desyerba	J
k	k	stimmloser velarer Verschlusslaut	caña, laca, quisimos	k
l	l	lateraler alveolarer Approximant	lino, calor, principal	t
m	m	bilabialer Nasal	madre, comer, anfibio	p
n	n	alveolarer Nasal	nido, anillo, sin	t
ɲ	J	palataler Nasal	cabaña, ñoquis	J
ŋ	N	velarer Nasal	angosto, increíble	k
p	p	stimmloser bilabialer Verschlus slaut	pozo, topo	p
r	r	alveolarer Vibrant	perro, enrachado	r
s	s	stimmlose r alveolarer Reibelaut	saco, casa, puertas	S

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ʃ	S	stimmloser postalveolarer Reibelaut	show, flash	S
t	t	stimmlose r alveolarer Verschlusslaut	tamiz, átomo	t
͡tʃ	tS	stimmlose postalveolare Affrikate	chubasco	S
w	w	labiovelarer Approximant	fuego, fuimos, cuota, cuadro	u
x	x	stimmloser velarer Reibelaut	jamón, general, peaje, reloj	k
z	z	stimmhaft er alveolarer Reibelaut	rasgo, mismo	S
Vokale				
a	a	ungerundeter offener Zentralvo kal	tanque	a
e	e	ungerundeter halbgeschlossener Vorderzungenvokal	peso	e
i	i	ungerundeter geschlossener Vorderzungenvokal	cinco	i

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
o	o	gerundeter halbgeschlossener Hinterzungenvokal	bosque	o
u	u	gerundeter geschlossener Hinterzungenvokal	publicar	u
Zusätzliche Symbole				
ˈ	"	Hauptakzent	Alabama	
ˌ	%	Nebenakzent	Alabama	
.	.	Silbengrenze	A.la.ba.ma	

Schwedisch (sv-SE)

In der folgenden Tabelle sind die Phoneme des Internationalen Phonetischen Alphabets (IPA), die X-SAMPA-Symbole (Extended Speech Assessment Methods Phonetic Alphabet) und die entsprechenden Viseme für die schwedische Stimme aufgeführt, die von Amazon Polly unterstützt werden.

Tabelle der Phoneme/Viseme

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
Konsonanten				
b	b	stimmhafter bilabialer Verschlus slaut	bil	p
d	d	stimmhaft er alveolarer Verschlusslaut	dal	t

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ɖ	d`	stimmhafter retroflexer Verschlusslaut	bord	t
f	f	stimmloser labiodentaler Frikativ	fil	f
g	g	stimmhafter velarer Verschlusslaut	gås	k
h	h	stimmloser glottaler Frikativ	hal	k
j	j	palataler Approximant	jag	i
k	k	stimmloser velarer Verschlusslaut	kal	k
l	l	alveolarer lateraler Approximant	lös	t
ɭ	l`	lateraler retroflexer Approximant	härlig	t
m	m	bilabialer Nasal	mil	p
n	n	alveolarer Nasal	nålar	t
ɳ	n`	retroflexer Nasal	barn	t
ŋ	N	velarer Nasal	ring	k
p	p	stimmloser bilabialer Verschlus slaut	pil	p

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
r	r	alveolarer Vibrant	ris	r
s	s	stimmlose r alveolarer Reibelaut	sil	S
ɕ	S\	stimmloser alveolopalataler Frikativ	tjock	J
ʂ	S`	stimmloser retroflexer Frikativ	fors, schlager	S
t	t	stimmlose r alveolarer Verschlusslaut	tal	t
ʈ	t`	stimmloser retroflexer Plosiv	hjort	t
V	V	stimmhafter labiodentaler Reibelaut	vår	f
w	w	labiovelarer Approximant	aula, airways	u
ɧ	x\	stimmloser palatal-velarer Frikativ	sjuk	k
Vokale				
ø	2	gerundeter halbgeschlossener Vorderzungenvokal	föll, förr	o

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ø	2:	langer gerundeter halbgeschlossener Vorderzungenvokal	föl, nöt, för	o
ɵ	8	gerundeter halbgeschlossener Zentralvokal	buss, full	o
ə	@	Schwa	pojken	@
ʉː	}:	langer gerundete r geschlossener Zentralvokal	hus, ful	u
a	a	ungerundeter offener Vor	hall, matt	a
æ	{	ungerundeter fast offener Vorderzun genvokal	herr	a
ɑː	A:	langer ungerunde ter offener Hinterzungenvokal	hal, mat	a
eː	e:	langer ungerunde ter halbgesch lossener Vorderzun genvokal	vet, hel	e
ɛ	E	ungerundeter halboffener Vorderzungenvokal	vett, rätt, hetta, häll	E
ɛː	E:	langer ungerunde ter halboffener Vorderzungenvokal	säl, häl, här	E:

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
iː	i:	langer ungerunde ter geschlossener Vorderzungenvokal	vit, sil	i:
ɪ	I	ungerundeter zentralisierter fast geschlossener Vorderzungenvokal	vitt, sill	i
oː	o:	langer gerundeter halbgeschlossener Hinterzungenvokal	hål, mål	o
ɔ	O	gerundeter halboffener Hinterzungenvokal	håll, moll	O
uː	u:	langer gerundete r geschlossener hinterer Vokal	sol, bot	u
ʊ	U	gerundeter zentralisierter fast geschlossener Hinterzungenvokal	bott	u
y	y	geschlossener gerundeter Vorderzungenvokal	bytt	u
yː	y:	langer gerundete r geschlossener Vorderzungenvokal	syl, syl	u
Zusätzliche Symbole				
ˈ	"	Hauptakzent	Alabama	

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ˌ	%	Nebenakzent	Alabama	
.	.	Silbengrenze	A.la.ba.ma	

Türkisch (tr-TR)

In der folgenden Tabelle sind die Phoneme des Internationalen Phonetischen Alphabets (IPA), die X-SAMPA-Symbole (Extended Speech Assessment Methods Phonetic Alphabet) und die entsprechenden Viseme für die türkische Stimme aufgeführt, die von Amazon Polly unterstützt werden.

Tabelle der Phoneme/Viseme

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
Konsonanten				
ɾ	4	alveolarer Tap	durum	t
ɾ̥	4_0_r	stimmloser frikativer alveolarer Tap	bir	t
ɾ̞	4_r	frikativer alveolarer Tap	raf	t
b	b	stimmhafter bilabialer Verschlusslaut	raf	p
c	c	stimmloser palataler Plosiv	kedi	k
d	d	stimmhafter alveolarer Verschlusslaut	dede	t

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ʤ	dZ	stimmhafte postalveolare Affrikate	cam	S
f	f	stimmloser labiodentaler Frikativ	fare	f
g	g	stimmhafter velarer Plosiv	galibi	k
h	h	stimmloser glottaler Frikativ	hasta	k
j	j	palataler Approximant	yat	i
ɟ	J\	stimmhafter palataler Plosiv	genç	J
k	k	stimmloser velarer Verschlusslaut	akıl	k
l	l	alveolarer lateraler Approximant	lale	t
ɫ	5	velarisierter alveolarer lateraler Approximant	labirent	t
m	m	bilabialer Nasal	maaş	p
n	n	alveolarer Nasal	anı	t
p	p	stimmloser bilabialer Verschlus slaut	ip	p

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
s	s	stimmlose r alveolarer Reibelaut	ses	S
ʃ	S	stimmloser postalveolarer Reibelaut	aşı	S
t	t	stimmlose r alveolarer Verschlusslaut	ütü	t
t͡ʃ	tS	stimmlose postalveolare Affrikate	çaba	S
V	V	stimmhafter labiodentaler Reibelaut	ekvator, kahveci, akvaryum, isveçli, teşviki, cetvel	f
z	z	stimmhaft er alveolarer Reibelaut	ver	s
ʒ	Z	stimmhafter postalveolarer Frikativ	azık	S
Vokale				
ø	2	gerundeter halbgeschlossener Vorderzungenvokal	göl	0
œ	9	gerundeter halboffener Vorderzungenvokal	banliyö	O

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
a	a	ungerundeter offener Vor	kal	a
a:	a:	langer ungerunde ter offener Vorderzungenvokal	davacı	a
æ	{	ungerundeter fast offener Vorderzun genvokal	özlem, güvenlik, gürel, somersault	a
e	e	ungerundeter halbgeschlossener Vorderzungenvokal	keçi	e
ɛ	E	ungerundeter halboffener Vorderzungenvokal	dede	E
i	i	ungerundeter geschlossener Vorderzungenvokal	bir	i
i:	i:	langer ungerunde ter geschlossener Vorderzungenvokal	izah	i
ɪ	I	ungerundeter zentralisierter fast geschlossener Vorderzungenvokal	keçi	i
ɯ	Mio.	geschlossener ungerundeter Hinterzungenvokal	kıl	i

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
o	o	gerundeter halbgeschlossener Hinterzungenvokal	kol	o
oː	o:	langer gerundeter halbgeschlossener Hinterzungenvokal	dolar	o
u	u	gerundeter geschlossener Hinterzungenvokal	durum	u
uː	u:	langer gerundeter geschlossener hinterer Vokal	ruhum	u
ʊ	U	gerundeter zentralisierter fast geschlossener Hinterzungenvokal	dolu	u
y	y	geschlossener gerundeter Vorderzungenvokal	güvenlik	u
ʏ	Y	gerundeter zentralisierter fast geschlossener Vorderzungenvokal	aşı	u
Zusätzliche Symbole				
ˈ	"	Hauptakzent	Alabama	
ˌ	%	Nebenakzent	Alabama	
.	.	Silbengrenze	A.la.ba.ma	

Walisisch (cy-GB)

In der folgenden Tabelle sind die Phoneme des Internationalen Phonetischen Alphabets (IPA), die X-SAMPA-Symbole (Extended Speech Assessment Methods Phonetic Alphabet) und die entsprechenden Viseme für die walisische Stimme aufgeführt, die von Amazon Polly unterstützt werden.

Tabelle der Phoneme/Viseme

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
Konsonanten				
b	b	stimmhafter bilabialer Verschlus slaut	baban	p
d	d	stimmhaft er alveolarer Verschlusslaut	deg	t
͡dʒ	dZ	stimmhafte postalveolare Affrikate	garej	S
ð	D	stimmhafter dentaler Frikativ	deuddeg	T
f	f	stimmloser labiodentaler Frikativ	ffacs	f
g	g	stimmhafter velarer Verschlusslaut	gadael	k
h	h	stimmloser glottaler Frikativ	haearn	k
j	j	palataler Approximant	astudio	i

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
k	k	stimmloser velarer Verschlusslaut	cant	k
l	l	alveolarer lateraler Approximant	lan	t
ɬ	K	stimmloser alveolarer lateraler Frikativ	llan	t
m	m	bilabialer Nasal	mae	p
m̥	m_0	stimmloser bilabialer Nasal	ymhen	p
n	n	alveolarer Nasal	naw	t
n̥	n_0	stimmloser alveolarer Nasal	anhawster	t
ŋ	N	velarer Nasal	argyfwng	k
ŋ̊	N_0	stimmloser velarer Nasal	anghenion	k
p	p	stimmloser bilabialer Verschlus slaut	pump	p
r	r	alveolarer Vibrant	rhoi	r
r̥	r_0	stimmloser alveolarer Vibrant	garw	r
s	s	stimmlose r alveolarer Reibelaut	saith	S

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
ʃ	S	stimmloser postalveolarer Reibelaut	siawns	S
t	t	stimmlose r alveolarer Verschlusslaut	tegan	t
͡tʃ	tS	stimmlose postalveolare Affrikate	cytsain	S
θ	T	stimmloser dentaler Reibelaut	aberth	T
V	V	stimmhafter labiodentaler Reibelaut	prawf	f
w	w	labiovelarer Approximant	rhagweld	u
χ	X	stimmloser uvularer Frikativ	chwech	k
z	z	stimmhaft er alveolarer Reibelaut	aids	s
ʒ	Z	stimmhafter postalveolarer Frikativ	rouge	S
Vokale				
ə	@	Schwa	ychwanega	@

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
a	a	ungerundeter offener Vor	acen	a
ai	ai	Diphthong	dau	a
au	au	Diphthong	awdur	a
ɑː	A:	langer ungerunde ter offener Hinterzungenvokal	mab	a
ɑːɬ	A:1	Diphthong	aelod	a
eː	e:	langer ungerunde ter halbgesch lossener Vorderzun genvokal	peth	e
ɛ	E	ungerundeter halboffener Vorderzungenvokal	pedwar	E
ɛi	Ei	Diphthong	beic	E
iː	i:	langer ungerunde ter geschlossener Vorderzungenvokal	tri	i
ɪ	I	ungerundeter zentralisierter fast geschlossener Vorderzungenvokal	miliwn	i
ɬu	1u	Diphthong	unigryw	i

IPA	X-SAMPA	Beschreibung	Beispiel	Viseme
oː	o:	langer gerundeter halbgeschlossener Hinterzungenvokal	oddi	o
ɔ	O	gerundeter halboffener Hinterzungenvokal	oddieithr	O
ɔi	Oi	Diphthong	troi	O
ɔu	Ou	Diphthong	rownd	O
uː	u:	langer gerundeter geschlossener hinterer Vokal	cwch	u
ʊ	U	gerundeter zentralisierter fast geschlossener Hinterzungenvokal	acwstig	u
ʊi	Ui	Diphthong	wyth	u
Zusätzliche Symbole				
ˈ	"	Hauptakzent	Alabama	
ˌ	%	Nebenakzent	Alabama	
.	.	Silbengrenze	A.la.ba.ma	

Sprachmodule von Amazon Polly

Amazon Polly verfügt über vier Sprachmodule, die den eingegebenen Text in lebensechte Sprache umwandeln. Dazu gehören: Generative Long-form, Neural und Standard. Um eine Amazon Polly-Stimme zu verwenden, wählen Sie eine Engine und einen Sprachsynthese-API-Vorgang aus. Geben Sie dann den Eingangstext ein, den die Engine synthetisieren soll, und wählen Sie ein Audioausgabeformat aus. Anhand dieser Eingaben synthetisiert Amazon Polly den bereitgestellten Text zu einem hochwertigen Sprach-Audiostream.

Die folgenden Abschnitte enthalten Einzelheiten zu den von Amazon Polly angebotenen Sprachmodulen.

Topics

Generative Stimmen

Die generative Text-to-Speech-Engine (TTS) von Amazon Polly bietet die menschenähnlichsten, emotionalsten und anpassungsfähigsten Konversationsstimmen, die für die Nutzung über die Amazon Polly Polly-Konsole verfügbar sind.

Die Generative Engine ist das bisher größte Amazon Polly TTS-Modell. Sie verwendet einen Transformator mit Milliarden Parametern, der Rohtext in Sprachcodes umwandelt, gefolgt von einem auf Faltung basierenden Decoder, der diese Sprachcodes schrittweise und streamfähig in Wellenformen umwandelt. Diese Methode zeigt, welche Fähigkeiten Large Language Models (LLMs) häufig entwickeln, wenn sie mit immer größeren Mengen öffentlich verfügbarer und urheberrechtlich geschützter Daten trainiert werden, die eine Vielzahl von Stimmen, Sprachen und Stilen beinhalten.

Die Generative Engine erzeugt synthetische Sprache, die emotional engagiert, durchsetzungsfähig und sehr umgangssprachlich ist, und zwar auf eine Weise, die der menschlichen Stimme bemerkenswert ähnlich ist. Sie können diese Stimmen als sachkundigen Kundenassistenten,

virtuellen Trainer oder Werbetreibenden mit einer fast menschlichen synthetischen Sprache verwenden.

> **Note**
>
> Die hochmoderne Technologie, die diesen Stimmen zugrunde liegt, fällt in das Paradigma der generativen KI für Sprach- und Stimmmodellierung. Ein Nebeneffekt der Technologie besteht darin, dass jegliche Aktualisierung der Trainingsdaten und des Modells zu geringfügigen Abweichungen im Klang der Stimmen führen kann, selbst wenn sich ihre Gesamtqualität durch Modellaktualisierungen verbessert. Dies könnte sich auf Anwendungsfälle auswirken, bei denen verschiedene Inhaltsteile über einen langen Zeitraum synthetisiert wurden — beispielsweise eine Podcast-Saison.

Verfügbare generative Stimmen

Amazon Polly bietet derzeit 43 Stimmen in einer generativen Variante an.

	Sprache	Sprachcode	Name/ID	Gender
1	Englisch (australisch)	en-AU	Olivia	Weiblich
2	Englisch (britisch)	en-GB	Amy Brian	Weiblich Männlich
3	Englisch (indisch)	en-IN	Kajal	Weiblich
4	Englisch (Irland)	en-IE	Niamh	Weiblich
5	Englisch (Neuseeland)	en-NZ	Aria	Weiblich
6	Englisch (Singapurisch)	en-SG	Jasmin	Weiblich

	Sprache	Sprachcode	Name/ID	Gender
7	Englisch (Südafrikanisch)	en-ZA	Ayanda	Weiblich
8	Englisch (amerikanisch)	en-US	Danielle	Weiblich
			Joanna	Weiblich
			Matthew	Männlich
			Ruth	Weiblich
			Salli	Weiblich
			Stephen	Männlich
			Tiffany	Weiblich
9	Niederländisch (Belgien)	nl-BE	Lisa	Weiblich
10	Niederländisch (Niederlande)	nl-NL	Laura	Weiblich
11	Französisch (Belgisch)	fr-BE	Isabelle	Weiblich
12	Kanadisches Französisch	fr-CA	Gabrielle	Weiblich
			Liam	Männlich
13	Französisch (Frankreich)	fr-FR	Ambre	Weiblich
			Céline	Weiblich
			Florian	Männlich
			Léa	Weiblich
			Remi	Männlich

	Sprache	Sprachcode	Name/ID	Gender
14	Deutsch (Österreich)	de-AT	Hannah	Weiblich
15	Deutsch (Deutschland)	de-DE	Daniel	Männlich
			Lennart	Männlich
			Vicki	Weiblich
16	Deutsch (Schweizerisch)	de-CH	Sabrina	Weiblich
17	Italienisch (Italien)	it-IT	Beatrice	Weiblich
			Bianca	Weiblich
			Lorenzo	Männlich
18	Koreanisch (Korea)	ko-KR	Seoyeon	Weiblich
19	Polnisch (Polen)	pl-PL	Ewa	Weiblich
			Ola	Weiblich
20	Portugiesisch (brasilianisch)	pt-BR	Camila	Weiblich
21	Spanisch (Mexikanisch)	es-MX	Andrés	Männlich
			Mia	Weiblich
22	Spanisch (Spanien)	es-ES	Lucia	Weiblich
			Sergio	Männlich
23	Spanisch (USA)	es-US	Lupe	Weiblich
			Pedro	Männlich

> **Note**
>
> Die Kosten für Generative Stimmen sind auf der Preisinformationsseite von Amazon Polly angegeben.

Kompatibilität mit Funktionen und Regionen

Generative Stimmen von Amazon Polly sind in den folgenden Regionen verfügbar:

- USA Ost (Nord-Virginia): us-east-1
- Europa (Frankfurt): eu-central-1
- USA West (Oregon): us-west-2
- Asien-Pazifik (Tokio): ap-northeast-1
- Asien-Pazifik (Seoul): ap-northeast-2
- Asien-Pazifik (Singapur): ap-southeast-1
- Europa (London): eu-west-2
- Kanada (Zentral): ca-central-1
- Europa (Zürich): eu-central-2
- Andere Regionen sind nicht verfügbar

Die folgenden Funktionen werden für generative Stimmen unterstützt:

- Die bidirektionale Streaming-API wird jetzt in Generative Engine angeboten und ermöglicht das gleichzeitige Streamen von Eingabe und Ausgabe. Diese API ist in den folgenden AWS Regionen verfügbar: USA Ost (Nord-Virginia), Europa (Frankfurt), USA West (Oregon), Asien-Pazifik (Singapur), Europa (London), Kanada (Zentral) und Europa (Zürich). Weitere Informationen zur Verwendung finden Sie in der Dokumentation.
- Real-time und asynchrone Sprachsyntheseoperationen.
- Der Sprechstil von Newscaster wird von der Generative Engine nicht unterstützt.
- Viele (aber nicht alle) SSML-Tags werden von Amazon Polly unterstützt. Weitere Informationen zu NTTS-supported SSML-Tags finden Sie unter Unterstützte SSML-Tags
- Wie bei Standardstimmungen können Sie aus verschiedenen Samplingraten wählen, um die Bandbreite und Audioqualität für Ihre Anwendung zu optimieren. Gültige Samplingraten für

Standard- und neuronale Stimmen sind 8 kHz, 16 kHz, 22 kHz oder 24 kHz. Der Standardwert für Standardstimmen ist 22 kHz. Die Standardeinstellung für generative Stimmen ist 24 kHz. Amazon Polly unterstützt die Audiostreamformate MP3, OGG (Vorbis) und PCM-Rohdaten.

Support für die Generierung von Sprachmarken ist derzeit nicht verfügbar.

> **Note**
>
> Derzeit unterstützen die Regionen Europa (London), Kanada (Zentral) und Europa (Zürich) nur die folgenden generativen Stimmen: Joanna (en-US), Ruth (en-US), Salli (en-US), Stephen (en-US), Tiffany (en-US), Amy (en-GB), Brian (en-GB), Olivia (en-AU), Florian (fr-FR), Ambre (fr-FR), Lorenzo (it-IT)), Beatrice (it-IT), Jasmine (de-SG), Aria (de-DE), Lennart (de-DE), Vicki (de-DE), Sabrina (de-CH), Hannah (de-AT), Niamh (de-DE), Camila (de-DE), Lisa (de-DE) und Seoyeon (ko-KR)

> **Note**
>
> Im unwahrscheinlichen Fall einer Modellhalluzination (und aufgrund des Modellverhaltens der Generative Engine, die Sprache Zeichen für Zeichen wiederzugeben) ist ein erzwungener Not-Aus-Mechanismus vorhanden. Der eingebaute Mechanismus verhindert, dass das Modell Sprache weiter wiedergibt. Dieses Sicherheitsmerkmal basiert auf einer Datenanalyse, bei der das Modell halluzinieren kann, normalerweise am Ende des Satzes.
> Es kann Fälle geben, in denen das Modell denkt, es würde halluzinieren, und dann während eines Generationsschritts ein Wort herausschneiden und so die Hälfte des Wortes wiedergeben. Dies könnte möglicherweise zu unangemessenen Ergebnissen führen.

Long-form Stimmen

Amazon Polly verfügt über eine Long-form Engine, die menschenähnliche, ausdrucksstarke und emotional versierte Stimmen erzeugt. Long-form Stimmen sind so konzipiert, dass sie die Aufmerksamkeit der Zuhörer auf längere Inhalte wie Nachrichtenartikel, Schulungsmaterial oder Marketingvideos lenken.

Amazon Polly Long-form Polly-Stimmen wurden mit einer hochmodernen Deep-Learning-TTS-Technologie entwickelt. Das Modell lernt, Phoneme, Prosodie, Intonation und andere phonetische

und akustische Aspekte der menschlichen Sprache nachzubilden, was zu einer sehr natürlichen Sprachausgabe führt.

Die Long-form Engine verwendet Texteinbettungen, um die Bedeutung eines Textes zu interpretieren. Mithilfe von Texteinbettungen kann die Long-form Engine die richtige Betonung, die richtigen Pausen und den richtigen Ton einer natürlichen Stimme erzeugen. Das Ergebnis ist eine Stimme, die die gesamte Bandbreite der emotionalen Elemente der menschlichen Kommunikation kombiniert. Dazu gehört die Nachahmung überraschender Dialoge oder die Differenzierung von Erzählungen. Zusammen entsteht so ein erstklassiges Sprachprodukt, das wie ein lebender Mensch klingt.

> **Note**
>
> Die hochmoderne Technologie, die diesen Stimmen zugrunde liegt, fällt unter das Paradigma der generativen KI für Sprach- und Stimmmodellierung. Ein Nebeneffekt der Technologie besteht darin, dass jegliche Aktualisierung der Trainingsdaten und des Modells zu geringfügigen Abweichungen im Klang der Stimmen führen kann, selbst wenn sich ihre Gesamtqualität durch Modellaktualisierungen verbessert. Dies könnte sich auf Anwendungsfälle auswirken, bei denen verschiedene Inhaltsteile über einen langen Zeitraum synthetisiert wurden — beispielsweise eine Podcast-Saison.

Verfügbare Stimmen in Langform

Amazon Polly bietet derzeit vier Langform-Stimmen en-US und zwei es-ES an. In beiden Sprachen sind Frauen- und Männerstimmen verfügbar. Die englischen Langformstimmen Daniel, Gregory und Ruth sind auch in einer NTTS-Variante für Konversationen erhältlich.

	Sprache	Sprachcode	Name/ID	Gender
1	Englisch (amerikanisch)	en-US	Danielle	Weiblich
			Gregor	Männlich
			Ruth	Weiblich
			Patrick	Männlich
2	Spanisch (Spanien)	es-ES	Alba	Weiblich

	Sprache	Sprachcode	Name/ID	Gender
			Raul	Männlich

Kompatibilität mit Funktionen und Regionen

Langform-Stimmen von Amazon Polly sind in den folgenden Regionen erhältlich:

- USA Ost (Nord-Virginia): us-east-1
- Andere Regionen sind nicht verfügbar

Die Amazon Polly Long-form Polly-Engine unterstützt die folgenden Funktionen:

- Real-time und asynchrone Sprachsyntheseoperationen.
- Alle Sprachzeichen.
- Viele (aber nicht alle) SSML-Tags werden von Amazon Polly unterstützt. Weitere Informationen zu NTTS-supported SSML-Tags finden Sie unter Unterstützte SSML-Tags
- Wie bei Standardstimmungen können Sie aus verschiedenen Samplingraten wählen, um die Bandbreite und Audioqualität für Ihre Anwendung zu optimieren. Gültige Abtastraten für Standard-, Langform- und neuronale Stimmen sind: 8 kHz, 16 kHz, 22 kHz oder 24 kHz. Der Standardwert für Standardstimmen ist 22 kHz. Die Standardeinstellung für Langform- und neuronale Stimmen ist 24 kHz. Amazon Polly unterstützt die Audiostreamformate MP3, OGG (Vorbis) und PCM-Rohdaten.

> **Note**
>
> Long-form Die Kosten für Voices sind auf der Preisinformationsseite von Amazon Polly angegeben.

Neuronale Stimmen

Amazon Polly verfügt über eine neuronale Text-to-Speech-Engine (NTTS), die Stimmen in noch höherer Qualität erzeugen kann als ihre Standardstimmen. Standard-TTS-Stimmen verwenden eine verkettete Synthese. Die Standard-Engine verkettet Phoneme aufgezeichneter Sprache und erzeugt so eine sehr natürlich klingende synthetisierte Sprache. Die unvermeidlichen Variationen der Sprache und die Techniken, die zum Segmentieren der Wellenformen verwendet werden, beschränken

jedoch die Qualität der Sprache. Die Amazon Polly NTTS-Engine verwendet keine standardmäßige verkettete Synthese, um Sprache zu erzeugen. Es besteht aus zwei Teilen:

- Ein neuronales Netzwerk — das eine Folge von Phonemen (die grundlegendsten Spracheinheiten) in eine Folge von Spektrogrammen umwandelt. (Spektogramme sind Momentaufnahmen der Energieniveaus in verschiedenen Frequenzbändern.)
- Ein Vocoder — der Spektrogramme in ein nahezu kontinuierliches Audiosignal umwandelt.

Die erste Komponente des neuronalen TTS-Systems ist ein Sequenz-zu-Sequenz-Modell. Dieses Modell erstellt seine Ergebnisse nicht nur aus der entsprechenden Eingabe, sondern berücksichtigt auch, wie die Sequenz der Elemente der Eingabe zusammenarbeiten. Das Modell wählt die ausgegebenen Spektrogramme so aus, dass ihre Frequenzbänder akustische Merkmale betonen, die das menschliche Gehirn bei der Sprachverarbeitung verwendet.

Die Ausgabe dieses Modells wird dann an einen neuronalen Vocoder übergeben. Dadurch werden die Spektrogramme in Sprach-Wellenformen konvertiert. Wenn dieser Ansatz von Sequenz zu Sequenz mit den großen Datensätzen trainiert wird, die zum Aufbau von Allzwecksystemen für die verkettete Synthese verwendet werden, wird er zu qualitativ hochwertigeren und natürlicheren Stimmen führen.

Verfügbare neuronale Stimmen

Neuronale Stimmen sind in 36 Sprachen und Sprachvarianten verfügbar. In der folgenden Tabelle werden die Stimmen aufgelistet.

	Sprache und Sprachvarianten	Sprachcode	Name/ID	Gender
1	Arabisch (Golf)	ar-AE	Hala Zayd	Weiblich Männlich
2	Belgisches Niederländisch (Flämisch)	nl-BE	Lisa	Weiblich
3	katalanisch	ca-ES	Arlet	Weiblich

	Sprache und Sprachvarianten	Sprachcode	Name/ID	Gender
4	Tschechisch	cs-CZ	Jitka	Weiblich
5	Chinesisch (Kantonesisch)	Yue-CN	Huujin	Weiblich
6	Chinesisch (Mandarin)	cmn-CN	Zhiyu	Weiblich
7	Dänisch	da-DK	Sofie	Weiblich
8	Niederländisch	nl-NL	Laura	Weiblich
9	Englisch (australisch)	en-AU	Olivia	Weiblich
10	Englisch (britisch)	en-GB	Amy* Emma Brian Artur	Weiblich Weiblich Männlich Männlich
11	Englisch (indisch)	en-IN	Kajal	Weiblich
12	Englisch (Irisch)	en-IE	Niamh	Weiblich
13	Englisch (Neuseeland)	en-NZ	Aria	Weiblich
14	Englisch (Singapurisch)	en-SG	Jasmin	Weiblich
15	Englisch (Südafrikanisch)	en-ZA	Ayanda	Weiblich

	Sprache und Sprachvarianten	Sprachcode	Name/ID	Gender
16	Englisch (amerikanisch)	en-US	Danielle	Weiblich
			Gregor	Männlich
			Ivy	Weiblich (Kind)
			Joanna*	Weiblich
			Kendra	Weiblich
			Kimberly	Weiblich
			Salli	Weiblich
			Joey	Männlich
			Justin	Männlich (Kind)
			Kevin	Männlich (Kind)
			Matthew*	Männlich
			Ruth	Weiblich
			Stephen	Männlich
17	Finnisch	fi-FI	Suvi	Weiblich
18	Französisch (Belgisch)	fr-BE	Isabelle	Weiblich
19	Kanadisches Französisch	fr-CA	Gabrielle	Weiblich
			Liam	Männlich
20	Französisch	fr-FR	Léa	Weiblich
			Remi	Männlich

	Sprache und Sprachvarianten	Sprachcode	Name/ID	Gender
21	Deutsch	de-DE	Vicki	Weiblich
			Daniel	Männlich
22	Deutsch (Österreichisch)	de-AT	Hannah	Weiblich
23	Deutsch (Schweizerisch)	de-CH	Sabrina	Weiblich
24	Hindi	hi-IN	Kajal	Weiblich
25	Italienisch	it-IT	Bianca	Weiblich
			Adriano	Männlich
26	Japanisch	ja-JP	Takumi	Männlich
			Kazuha	Weiblich
			Tomoko	Weiblich
27	Koreanisch	ko-KR	Seoyeon	Weiblich
			Jihye	Weiblich
28	Norwegisch	nb-NO	Ida	Weiblich
29	Polnisch	pl-PL	Ola	Weiblich
30	Portugiesisch (brasilianisch)	pt-BR	Camila	Weiblich
			Vito ria/Vitoria	Weiblich
			Thiago	Männlich
31	Portugiesisch (europäisch)	pt-PT	Ini s/Ines	Weiblich

	Sprache und Sprachvarianten	Sprachcode	Name/ID	Gender
32	Spanisch (Spanien)	es-ES	Lucia Sergio	Weiblich Männlich
33	Spanisch (Mexikanisch)	es-MX	Mia Andrés	Weiblich Männlich
34	Spanisch (USA)	es-US	Lupe* Pedro	Weiblich Männlich
35	Schwedisch	sv-SE	Elin	Weiblich
36	Türkisch	tr-TR	Burcu	Weiblich

*Die Stimmen von Amy, Joanna, Lupe und Matthew können im Newscaster-Sprechstil verwendet werden. Weitere Informationen finden Sie unter Die Stimme des Nachrichtensprechers anwenden.

Kompatibilität mit Funktionen und Regionen

Neuronale Stimmen sind nicht in allen AWS Regionen verfügbar und unterstützen auch nicht alle Funktionen von Amazon Polly.

Neuronale Stimmen werden in den folgenden Regionen unterstützt:

- USA Ost (Nord-Virginia): us-east-1
- USA West (Oregon): us-west-2
- Afrika (Kapstadt): af-south-1
- Asien-Pazifik (Tokio): ap-northeast-1
- Asien-Pazifik (Seoul): ap-northeast-2
- Asien-Pazifik (Osaka): ap-northeast-3
- Asien-Pazifik (Mumbai): ap-south-1
- Asien-Pazifik (Singapur): ap-southeast-1
- Asien-Pazifik (Sydney): ap-southeast-2

- Asien-Pazifik (Malaysia): ap-southeast-5
- Asien-Pazifik (Thailand): ap-southeast-7
- Kanada (Zentral): ca-central-1
- Europa (Frankfurt): eu-central-1
- Europa (Irland): eu-west-1
- Europa (London): eu-west-2
- Europa (Paris): eu-west-3
- Europa (Spanien): eu-south-2
- Europa (Zürich): eu-central-2
- AWS GovCloud ()US-West: us-gov-west-1

Endpunkte und Protokolle für diese Regionen sind identisch mit denen für Standardstimmen. Weitere Informationen finden Sie unter Amazon Polly Polly-Endpunkte und Kontingente.

Die folgenden Funktionen werden für neuronale Stimmen unterstützt:

- Real-time und asynchrone Sprachsyntheseoperationen.
- Sprechstil von Newscaster. Weitere Informationen zu den Sprechstilen finden Sie unterDie Stimme des Nachrichtensprechers anwenden.
- Alle Sprachzeichen.
- Viele (aber nicht alle) SSML-Tags, die von Amazon Polly unterstützt werden. Weitere Informationen zu NTTS-supported SSML-Tags finden Sie unter Unterstützte Tags.

Wie bei Standardstimmungen können Sie aus verschiedenen Samplingraten wählen, um die Bandbreite und Audioqualität für Ihre Anwendung zu optimieren. Gültige Samplingraten für Standard- und neuronale Stimmen sind 8 kHz, 16 kHz, 22 kHz oder 24 kHz. Der Standardwert für Standardstimmen ist 22 kHz. Die Standardeinstellung für neuronale Stimmen ist 24 kHz. Amazon Polly unterstützt die Audiostreamformate MP3, OGG (Vorbis) und PCM-Rohdaten.

Standardstimmen

Amazon Polly verfügt über eine Standard-Engine, die die verkettete Synthese verwendet. Die Standard-Engine verkettet Phoneme aufgezeichneter Sprache und erzeugt so eine sehr natürlich klingende synthetisierte Sprache.

Verfügbare Standardstimmen

Amazon Polly bietet derzeit 40 weibliche und 20 männliche Standardstimmen in 29 Sprach- und Sprachvarianten an.

	Sprache	Sprachcode	Name/ID	Gender
1	Arabisch	arb	Zeina	Weiblich
2	Chinesisch (Mandarin)	cmn-CN	Zhiyu	Weiblich
3	Dänisch	da-DK	Naja	Weiblich
			Mads	Männlich
4	Niederländisch	nl-NL	Lotte	Weiblich
			Ruben	Männlich
5	Englisch (australisch)	en-AU	Nicole	Weiblich
			Russell	Männlich
6	Englisch (britisch)	en-GB	Amy	Weiblich
			Emma	Weiblich
			Brian	Männlich
7	Englisch (indisch)	en-IN	Aditi	Weiblich
			Raveena	Weiblich
8	Englisch (amerikanisch)	en-US	Ivy	Weiblich
			Joanna	Weiblich
			Kendra	Weiblich
			Kimberly	Weiblich

	Sprache	Sprachcode	Name/ID	Gender
			Salli	Weiblich
			Joey	Männlich
			Kevin	Männlich
9	Englisch (walisisch)	en- GB-WLS	Geraint	Männlich
10	Französisch	fr-FR	Cé line/Celine	Weiblich
			Léa	Weiblich
			Mathieu	Männlich
11	Kanadisches Französisch	fr-CA	Chantal	Weiblich
12	Deutsch	de-DE	Marlene	Weiblich
			Vicki	Weiblich
			Hans	Männlich
13	Hindi	hi-IN	Aditi	Weiblich
14	Isländisch	is-IS	Dó ra/Dora	Weiblich
			Karl	Männlich
15	Italienisch	it-IT	Carla	Weiblich
			Bianca	Weiblich
			Giorgio	Männlich
16	Japanisch	ja-JP	Mizuki	Weiblich
			Takumi	Männlich

	Sprache	Sprachcode	Name/ID	Gender
17	Koreanisch	ko-KR	Seoyeon	Weiblich
18	Norwegisch	nb-NO	Liv	Weiblich
19	Polnisch	pl-PL	Ewa	Weiblich
			Maja	Weiblich
			Jacek	Männlich
			. Jan.	Männlich
20	Portugiesisch (brasilianisch)	pt-BR	Camila	Weiblich
			Vito ria/Vitoria	Weiblich
			Ricardo	Männlich
21	Portugiesisch (europäisch)	pt-PT	Ini s/Ines	Weiblich
			Cristiano	Männlich
22	Rumänisch	ro-RO	Carmen	Weiblich
23	Russisch	ru-RU	Tatyana	Weiblich
			Maxim	Männlich
24	Spanisch (Spanien)	es-ES	Conchita	Weiblich
			Lucia	Weiblich
			Enrique	Männlich
25	Spanisch (Mexikanisch)	es-MX	Mia	Weiblich

	Sprache	Sprachcode	Name/ID	Gender
26	Spanisch (USA)	es-US	Lupe Pené lope/Pene lope Miguel	Weiblich Weiblich Männlich
27	Schwedisch	sv-SE	Astrid	Weiblich
28	Türkisch	tr-TR	Filiz	Männlich
29	Walisisch	cy-GB	Gwyneth	Weiblich

Kompatibilität mit Funktionen und Regionen

Standardstimmen von Amazon Polly sind in den folgenden Amazon Polly Polly-Regionen verfügbar:

- USA Ost (Nord-Virginia): us-east-1
- USA Ost (Ohio): us-east-2
- USA West (Nordkalifornien): us-west-1
- USA West (Oregon): us-west-2
- Afrika (Kapstadt): af-south-1
- Asien-Pazifik (Hongkong): ap-east-1
- Asien-Pazifik (Tokio): ap-northeast-1
- Asien-Pazifik (Seoul): ap-northeast-2
- Asien-Pazifik (Osaka): ap-northeast-3
- Asien-Pazifik (Mumbai): ap-south-1
- Asien-Pazifik (Singapur): ap-southeast-1
- Asien-Pazifik (Sydney): ap-southeast-2
- Asien-Pazifik (Malaysia): ap-southeast-5
- China (Ningxia): cn-northwest-1;
- Kanada (Zentral): ca-central-1

- Europa (Frankfurt): eu-central-1
- Europa (Irland): eu-west-1
- Europa (London): eu-west-2
- Europa (Paris): eu-west-3
- Europa (Spanien): eu-south-2
- Europa (Stockholm): eu-north-1
- Naher Osten (Bahrain): me-south-1
- Südamerika (São Paulo): sa-east-1
- AWS GovCloud (US-West): us-gov-west-1

Die Endpunkte und Protokolle für diese Regionen sind identisch mit denen, die für neuronale Stimmen verwendet werden. Weitere Informationen finden Sie unter Amazon Polly Polly-Endpunkte und Kontingente.

Die Amazon Polly Polly-Standard-Engine unterstützt die folgenden Funktionen (TBD):

- Real-time und asynchrone Sprachsyntheseoperationen.
- Alle Sprachzeichen.
- Viele (aber nicht alle) SSML-Tags werden von Amazon Polly unterstützt. Weitere Informationen zu NTTS-supported SSML-Tags finden Sie unter Unterstützte SSML-Tags.
- Sie können aus verschiedenen Abtastraten wählen, um die Bandbreite und Audioqualität für Ihre Anwendung zu optimieren. Die Standard-Abtastraten für Standardstimmen sind 22 kHz. Amazon Polly unterstützt die Audiostreamformate MP3, OGG (Vorbis) und PCM-Rohdaten.

> **Note**
>
> Die Standardkosten für Voices sind auf der Seite mit den Preisinformationen von Amazon Polly angegeben.

Auswahl einer Sprachengine

Sie können über die Amazon Polly Polly-Konsole oder auf Amazon Polly-Stimmen zugreifen. AWS CLI

Um eine Sprachengine auf der Konsole auszuwählen

1. Öffnen Sie die Amazon Polly Polly-Konsole unter https://console.aws.amazon.com/polly/.
2. Wählen Sie in der Amazon Polly Polly-Konsole die gewünschte Sprach-Engine aus.
3. Wählen Sie die gewünschte Stimme aus dem Drop-down-Menü „Stimme“ aus.
4. Generieren Sie TTS-Audio mit Text Ihrer Wahl.

Um eine Sprachengine in der auszuwählen AWS CLI, geben Sie die Operationen `Engine` und `VoiceId` in den `StartSpeechSynthesisTask` API-Operationen `SyntheszieSpeech` oder an. Einige Beispiele finden Sie in den Schnellstart-Codebeispielen und den Python-Beispielen.

Sprachzeichen

Sprachmarkierungen sind Metadaten, die eine Sprachausgabe beschreiben. Sie kennzeichnen beispielsweise Anfang und Ende von Sätzen und Wörtern im Audiostream. Wenn Sie Sprachzeichen für Ihren Text anfordern, gibt Amazon Polly diese Metadaten anstelle von synthetisierter Sprache zurück. Wenn Sie die Sprachmarkierungen zusammen mit dem Audiostream der Sprachausgabe verwenden, können Sie die visuelle Darstellung Ihrer Anwendungen verbessern.

Wenn Sie beispielsweise die Metadaten mit dem Audiostream aus Ihrem Text kombinieren, können Sie Sprache mit Gesichtsanimationen synchronisieren (Lippensynchronisierung) oder geschriebene Wörter hervorheben, während sie gesprochen werden.

Sprachzeichen sind verfügbar, wenn neuronale Engines, Langform- oder Standard-Engines verwendet werden. text-to-speech

Themen

Arten von Sprachzeichen

Sie fordern Sprachzeichen an, indem Sie entweder die SpeechMarkTypesOption für die StartSpeechSynthesisTaskBefehle SynthesizeSpeechoder verwenden. Dabei geben Sie an, welche Metadatenelemente für Ihren Eingabetext zurückgegeben werden sollen. Sie können bis zu vier verschiedene Metadatentypen anfordern, müssen pro Anforderung jedoch mindestens einen Typ angeben. Durch eine solche Anfrage wird keine Audioausgabe generiert.

In der AWS CLI, zum Beispiel:

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

Amazon Polly generiert Sprachmarken mithilfe der folgenden Elemente:

- sentence: Kennzeichnet ein Satzelement im Eingabetext.
- word: Kennzeichnet ein Wortelement im Text.
- Viseme — Beschreibt die Gesichts- und Mundbewegungen, die jedem gesprochenen Phonem entsprechen. Weitere Informationen finden Sie unter Visemes und Amazon Polly.
- ssml — Beschreibt ein <mark>Element aus dem SSML-Eingabetext. Weitere Informationen finden Sie unter Sprache aus SSML-Dokumenten generieren.

Visemes und Amazon Polly

Ein Mundbild steht für die Position von Gesicht und Mund bei der Aussprache eines Wortes. Es ist das visuelle Äquivalent eines Phonems, der akustischen Grundeinheit der Wortbildung. Damit sind Mundbilder die visuellen Grundbausteine der Sprache.

Jede Sprache hat eine Reihe von Visemen, die ihren spezifischen Phonemen entsprechen. Zu jedem Phonem einer Sprache gibt es ein Mundbild, das beschreibt, wie sich der Mund formt, wenn der entsprechende Laut gebildet wird. Allerdings hat nicht jedes Phonem ein einzigartiges Mundbild, denn viele Phoneme werden mit identischer Mund- und Gesichtsform ausgesprochen, auch wenn sie sich lautlich unterscheiden. Im Englischen beispielsweise unterscheiden sich die Worte "pet" und "bet" lautlich voneinander. Rein visuell betrachtet (ohne den zugehörigen Sprachlaut) sind Gesichts- und Mundform bei ihrer Aussprache jedoch jeweils identisch.

Die folgende Tabelle enthält eine Teilmenge der Phoneme des Internationalen Phonetischen Alphabets (IPA) und die Symbole aus dem Extended Speech Assessment Methods Phonetic Alphabet (X-SAMPA) sowie die zugehörigen Mundbilder für Stimmen mit der Sprache US-Englisch.

Eine vollständige Tabelle sowie Tabellen für alle verfügbaren Sprachen finden Sie unter Sprachen bei Amazon Polly.

IPA	X-SAMPA	Description	Beispiel	Viseme
Konsonanten				
b	b	stimmhafter bilabialer Verschlus slaut	bed	p

IPA	X-SAMPA	Description	Beispiel	Viseme
d	d	stimmhaft er alveolarer Verschlusslaut	dig	t
ʤ	dZ	stimmhafte postalveolare Affrikate	jump	S
ð	D	stimmhafter dentaler Frikativ	then	T
f	f	stimmloser labiodentaler Frikativ	five	f
g	g	stimmhafter velarer Plosiv	game	k
h	h	stimmloser glottaler Frikativ	house	k
...	...	...	...	...

Ausgabe von Sprachmarken

Amazon Polly gibt Sprachmarkenobjekte in einem zeilengetrennten JSON-Stream zurück. Ein Sprachmarkierungsobjekt enthält die folgenden Felder:

- time: der Zeitstempel in Millisekunden relativ zum Beginn des entsprechenden Audiostreams
- type — der Typ des Sprachzeichens (Satz, Wort, Visem oder SSML)
- start — der Offset in Byte (nicht in Zeichen) des Beginns des Objekts im Eingabetext (ohne Visem-Zeichen)
- end — der Offset in Byte (nicht in Zeichen) des Endes des Objekts im Eingabetext (ohne Visem-Markierungen)
- value: variabel je nach Sprachmarkierungstyp

- SSML: SSML-Tag des Typs <mark>
- viseme: der Name des Mundbilds
- word oder sentence: eine Teilzeichenfolge des Eingabetexts, gekennzeichnet durch die Felder "start" und "end"

Amazon Polly generiert beispielsweise das folgende `word` Sprachmarkenobjekt aus dem Text „Mary had a little lamb“:

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

Das beschriebene Wort ("had") beginnt 373 Millisekunden nach Start des Audiostreams. Sein Anfang liegt bei Byte 5, sein Ende bei Byte 8 des Eingabetexts.

> **Note**
>
> Diese Metadaten wurden für die Stimme `Joanna` generiert. Wenn Sie für denselben Eingabetext eine andere Stimme verwenden, sehen die Metadaten möglicherweise anders aus.

Sprachzeichen anfordern

Sie können die Konsole oder den `synthesize-speech` Befehl verwenden, um Sprachmarken von Amazon Polly anzufordern. Anschließend können Sie die Metadaten anzeigen oder in einer Datei speichern.

Console

Um Sprachmarken auf der Konsole zu generieren

1. Melden Sie sich bei der an AWS-Managementkonsole und öffnen Sie die Amazon Polly Polly-Konsole unter https://console.aws.amazon.com/polly/.
2. Wählen Sie die Registerkarte Text-to-Speech.
3. Aktivieren Sie SSML, um SSML zu verwenden.
4. Geben Sie Ihren Text in das Eingabefeld ein oder kopieren Sie ihn in das Eingabefeld.
5. Wählen Sie unter Sprache die Sprache Ihres Textes aus.

6. Wählen Sie für Voice die Stimme aus, die Sie verwenden möchten.
7. Um die Textaussprache zu ändern, erweitern Sie Zusätzliche Einstellungen, aktivieren Sie „Aussprache anpassen“ und wählen Sie unter „Lexikon anwenden“ das gewünschte Lexikon aus.
8. Um die Sprache zu überprüfen, wählen Sie „Zuhören“.
9. Aktivieren Sie die Einstellungen für das Sprachdateiformat.

   > **Note**
   >
   > Beim Herunterladen MP3 der Formate OGG, PCM, MU-Law oder A-Law werden keine Sprachmarken generiert.

10. Wählen Sie als Dateiformat die Option Sprachmarken aus.
11. Wählen Sie unter Sprachmarkentypen die Arten von Sprachmarken aus, die generiert werden sollen. Die Option zur Auswahl von SSML-Metadaten ist nur verfügbar, wenn SSML aktiviert ist. Weitere Informationen zur Verwendung von SSML mit Amazon Polly finden Sie unter. Sprache aus SSML-Dokumenten generieren
12. Wählen Sie Herunterladen aus.

AWS CLI

Zusätzlich zum Eingabetext sind die folgenden Elemente erforderlich, um diese Metadaten zurückzugeben:

- `output-format`

  Amazon Polly unterstützt bei der Rückgabe von Sprachzeichen nur das JSON-Format.

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

  Wenn Sie ein nicht unterstütztes Ausgabeformat verwenden, löst Amazon Polly eine Ausnahme aus.

- `voice-id`

  Geben Sie dieselbe Stimme an, die zur Generierung des Sprachausgabe-Audiostreams verwendet wird. Nur so ist sichergestellt, dass die Metadaten auch zu dem entsprechenden Audiostream passen. Die verfügbaren Stimmen haben nicht alle dieselbe

Sprechgeschwindigkeit. Wenn Sie nicht die Stimme verwenden, mit der die Sprachausgabe generiert wurde, werden die Metadaten nicht zum Audiostream passen.

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

- `speech-mark-types`

  Geben Sie an, welche Sprachmarkierungstypen zurückgegeben werden sollen. Sie können jeden beliebigen Sprachmarkierungstyp anfordern oder auch sämtliche Sprachmarkierungstypen. Sie müssen aber mindestens einen Typ angeben.

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

- `text-type`

  Klartext ist der Standardeingabetext für Amazon Polly. Sie müssen ihn also verwenden, `text-type ssml` wenn Sie SSML-Sprachzeichen zurückgeben möchten.

- `outfile`

  Geben Sie die Ausgabedatei an, in die die Metadaten geschrieben werden sollen.

```
MaryLamb.txt
```

Das folgende AWS CLI Beispiel ist für Unix, Linux und macOS formatiert. Ersetzen Sie unter Windows den Unix-Fortsetzungszeichen mit umgekehrtem Schrägstrich (\) am Ende jeder Zeile durch ein Caret (^) und setzen Sie den Eingabetext in vollständige Anführungszeichen („) und einfache Anführungszeichen (') für interne Tags.

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

Beispiel für Sprachzeichen ohne SSML

Im nachfolgenden Beispiel sehen Sie die Bildschirmausgabe der angeforderten Metadaten für den einfachen Satz "Mary had a little lamb". Um das Beispiel einfach zu halten, haben wir SSML-Sprachmarkierungen hier außen vor gelassen.

Das folgende AWS CLI Beispiel ist für Unix, Linux und macOS formatiert. Ersetzen Sie unter Windows den Unix-Fortsetzungszeichen mit umgekehrtem Schrägstrich (\) am Ende jeder Zeile durch ein Caret (^) und setzen Sie den Eingabetext in vollständige Anführungszeichen („) und einfache Anführungszeichen (') für interne Tags.

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

Wenn Sie diese Anfrage stellen, gibt Amazon Polly Folgendes in der TXT-Datei zurück:

```
{"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"}
```

In dieser Ausgabe wurden sämtliche Teile des Texts in Sprachmarkierungen überführt:

- Der Satz "Mary had a little lamb."
- Jedes Wort im Text: "Mary", "had", "a", "little" und "lamb"
- Das Mundbild für jeden Laut im zugehörigen Audiostream: "p", "E", "r", "i" usw. Weitere Informationen zu Mundbildern finden Sie unter Visemes und Amazon Polly.

Beispiel für Sprachzeichen mit SSML

Die Generierung von Sprachmarkierungen aus SSML-erweitertem Text funktioniert ähnlich wie die Generierung aus Text ohne SSML. Sie verwenden den Befehl `synthesize-speech` und geben wie im nachfolgenden Beispiel dargestellt den SSML-erweiterten Text und die gewünschten Sprachmarkierungstypen an. Um das Beispiel leichter lesbar zu machen, haben wir keine Viseme-Sprachzeichen aufgenommen, aber diese könnten ebenfalls enthalten sein.

Das folgende AWS CLI Beispiel ist für Unix, Linux und macOS formatiert. Ersetzen Sie unter Windows den Unix-Fortsetzungszeichen mit umgekehrtem Schrägstrich (\) am Ende jeder Zeile durch ein Caret (^) und setzen Sie den Eingabetext in vollständige Anführungszeichen („) und einfache Anführungszeichen (') für interne Tags.

```
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
```

Wenn Sie diese Anfrage stellen, gibt Amazon Polly Folgendes in der TXT-Datei zurück:

```
{"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"}
```

Sprache aus SSML-Dokumenten generieren

Sie können Amazon Polly verwenden, um Sprache entweder aus reinem Text oder aus Dokumenten zu generieren, die mit Speech Synthesis Markup Language (SSML) markiert wurden. Durch die Verwendung von durch SSL erweitertem Text haben Sie zusätzliche Kontrolle darüber, wie Amazon Polly Sprache aus dem von Ihnen bereitgestellten Text generiert.

Mit SSML-Tags können Sie Sprachaspekte wie Aussprache, Lautstärke und Sprechgeschwindigkeit anpassen und steuern. In der wird der durch SSL erweiterte Text AWS-Managementkonsole, den Sie in Audio konvertieren möchten, auf der Registerkarte SSML der Seite eingegeben. Text-to-Speech Text, der im Klartext eingegeben wird, hängt zwar von den Standardeinstellungen für die Sprache und Stimme ab, die Sie ausgewählt haben, aber mit SSML erweiterter Text teilt Amazon Polly nicht nur mit, was Sie sagen möchten, sondern auch, wie Sie es sagen möchten. Mit Ausnahme der hinzugefügten SSML-Tags synthetisiert Amazon Polly durch SSL erweiterten Text auf die gleiche Weise wie Klartext. Weitere Informationen finden Sie unter Beispiel für Sprachsynthese mit Amazon Polly.

Wenn Sie SSML verwenden, schließen Sie den gesamten Text in ein `<speak>` Tag ein, damit Amazon Polly weiß, dass Sie SSML verwenden. Beispiel:

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

Anschließend verwenden Sie spezifische SSML-Tags für den Text innerhalb der `<speak>`-Tags, um die Art und Weise anzupassen, wie der Text klingen soll. Sie können eine Pause hinzufügen, die Geschwindigkeit der Sprachausgabe ändern, die Lautstärke der Stimme erhöhen oder senken oder viele weitere Anpassungen hinzufügen, sodass der Text für Sie richtig klingt. Eine vollständige Liste der SSML-Tags, die Sie verwenden können, finden Sie unter Unterstützte SSML-Tags.

Sie können beispielsweise eine lange Pause in den Text einfügen oder Sprechgeschwindigkeit oder Tonlage ändern. Weitere Optionen sind:

- Betonung bestimmter Wörter oder Phrasen
- Verwendung der phonetischen Aussprache
- einschließlich Atemgeräusche
- Whispering
- unter Verwendung des Newscaster-Sprechstils.

Vollständige Informationen zu den von Amazon Polly unterstützten SSML-Tags und deren Verwendung finden Sie unter Unterstützte SSML-Tags

Bei der Verwendung von SSML gibt es mehrere reservierte Zeichen, die eine spezielle Behandlung erfordern. Der Grund hierfür ist, dass SSML diese Zeichen als Teil des Codes verwendet. Verwenden Sie zu ihrem Einsatz eine bestimmte Entität, um sie durch Escapezeichen zu schützen. Weitere Informationen finden Sie unter Reservierte Zeichen in SSML.

Amazon Polly bietet diese Steuerungstypen mit einer Teilmenge der SSML-Markup-Tags, die durch die Speech Synthesis Markup Language (SSML) Version 1.1, W3C-Empfehlung, definiert sind.

Sie können SSML in der Amazon Polly Polly-Konsole oder mithilfe von verwenden. AWS CLI In den folgenden Themen wird beschrieben, wie Sie mit SSML Sprachausgaben generieren und die Ausgabe kontrollieren können, damit sie exakt Ihren Anforderungen entspricht.

Themen

Reservierte Zeichen in SSML

Es gibt fünf vordefinierte Zeichen, die normalerweise nicht innerhalb einer SSML-Anweisung verwendet werden können. Diese Entitäten sind durch die Sprachspezifikation reserviert. Diese Zeichen lauten wie folgt:

Zeichen	Name	Escape-Code
&	Kaufmännisches Und	&
"	Anführungszeichen (doppelte Anführung	"

Zeichen	Escape-Code	Name
		szeichen)
	&	kaufmännisches Und-Zeichen
	'	Apostroph oder einfaches Anführungszeichen
	<	Kleiner-als-Zeichen
	>	Größeres-als-Zeichen

Da SSML diese Zeichen als Teil des Codes verwendet, müssen Sie für die Verwendung dieser Symbole in SSML ein Escape-Zeichen für das Zeichen verwenden, wenn Sie es verwenden. Sie verwenden anstelle des tatsächlichen Zeichens den Escape-Code, damit er beim Erstellen eines gültigen SSML-Dokuments korrekt angezeigt wird. Beispiel: Der folgende Satz

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

würde in SSML als gerendert werden als

```
<speak>
```

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

In diesem Fall werden die Sonderzeichen für den Apostroph und das kaufmännische Und-Zeichen durch Escape-Zeichen geschützt, sodass das SSML-Dokument gültig bleibt.

Für die Symbole &, < und > sind Escape-Codes immer erforderlich, wenn Sie SSML verwenden. Wenn Sie das Apostroph/einfache Anführungszeichen (') als Apostroph verwenden, müssen Sie außerdem den Escape-Code verwenden.

Wenn Sie jedoch das doppelte Anführungszeichen (") oder das apostrophe/single Anführungszeichen (') als Anführungszeichen verwenden, hängt es vom Kontext ab, ob Sie den Escape-Code verwenden oder nicht.

Doppelte Anführungszeichen

- Muss durch Escape-Zeichen geschützt werden, wenn ein Attributwert durch doppelte Anführungszeichen getrennt ist. Zum Beispiel im folgenden AWS CLI Code

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

- Muss im Textkontext nicht durch Escape-Zeichen geschützt werden. Beispiel: Im folgenden

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

- Muss nicht durch Escape-Zeichen geschützt werden, wenn als Trennzeichen für ein Codeattribut einfache Anführungszeichen verwendet werden. Zum Beispiel im folgenden AWS CLI Code

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

Einfache Anführungszeichen

- Muss durch Escape-Zeichen geschützt werden, wenn es als Apostroph verwendet wird. Beispiel: Im folgenden

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

- Muss im Textkontext nicht durch Escape-Zeichen geschützt werden. Beispiel: Im folgenden

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

- Muss nicht durch Escape-Zeichen geschützt werden, wenn als Trennzeichen für ein Codeattribut doppelte Anführungszeichen verwendet werden. Zum Beispiel im folgenden AWS CLI Code

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

SSML auf der Konsole verwenden

Im folgenden Beispiel verwenden Sie ein SSML-Tag, um Amazon Polly anzuweisen, „W3C“ durch „World Wide Web Consortium“ zu ersetzen, wenn es einen kurzen Absatz liest. Sie können Tags auch verwenden, um eine Pause einzufügen oder ein Wort flüstern zu lassen. Vergleichen Sie das Ergebnis dieser Übung mit dem von Lexika anwenden (Sprachsynthese).

Weitere Informationen und Beispiele zu SSML finden Sie unter Unterstützte SSML-Tags.

So generieren Sie Sprachausgaben aus Text mit SSML-Tags (Konsole)

1. Melden Sie sich bei der an AWS-Managementkonsole und öffnen Sie die Amazon Polly Polly-Konsole unter https://console.aws.amazon.com/polly/.
2. Wählen Sie – falls erforderlich – die Registerkarte Text-to-Speech (Text in Sprache).
3. Schalten Sie SSML ein.
4. Geben Sie den folgenden Text in das Textfeld ein:

```
<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>
```

   Die SSML-Tags teilen Amazon Polly mit, wie der Text gerendert werden soll:

   - `<break time="1s"/>`weist Amazon Polly an, zwischen den ersten beiden Sätzen eine Sekunde zu pausieren.

- `<sub alias="World Wide Web Consortium">W3C</sub>`fordert Amazon Polly auf, das Akronym W3C durch World Wide Web Consortium zu ersetzen.
- `<amazon:effect name="whispered">Nice job</amazon:effect>`fordert Amazon Polly auf, die zweite Instanz von „Gute Arbeit“ zu flüstern.

> **Note**
>
> Wenn Sie den verwenden AWS CLI, setzen Sie den eingegebenen Text in Anführungszeichen, um ihn vom umgebenden Code zu unterscheiden. Die Amazon Polly Polly-Konsole zeigt Ihnen keinen Code an, sodass Sie den eingegebenen Text nicht in Anführungszeichen setzen, wenn Sie ihn verwenden.

5. Wählen Sie als Sprache Englisch, USA und dann eine Stimme aus.
6. Um sich die Rede anzuhören, wählen Sie „Zuhören“.
7. Um die Sprachdatei zu speichern, wählen Sie „Herunterladen“. Wenn Sie die Datei in einem anderen Format speichern möchten, erweitern Sie Zusätzliche Einstellungen, aktivieren Sie Einstellungen für das Sprachdateiformat, wählen Sie das gewünschte Format aus und wählen Sie dann Herunterladen aus.

Verwenden von SSML mit dem Befehl Synthesize-Speech

In diesem Beispiel wird gezeigt, wie der Befehl `synthesize-speech` mit einer SSML-Zeichenfolge verwendet wird. Wenn Sie den Befehl `synthesize-speech` verwenden, geben Sie üblicherweise Folgendes an:

- Eingabetext (erforderlich)
- Öffnendes und schließendes Tag (erforderlich)
- Das Ausgabeformat
- Eine Stimme

In diesem Beispiel geben Sie eine einfache Textzeichenfolge in Anführungszeichen sowie die erforderlichen öffnenden und schließenden `<speak></speak>`-Tags an.

Important

Obwohl Sie in der Amazon Polly Polly-Konsole keine Anführungszeichen für den Eingabetext verwenden, müssen Sie sie bei der Verwendung von verwenden. AWS CLI Es ist auch wichtig, dass Sie zwischen den Anführungszeichen rund um den Eingabetext und den für einzelne Tags erforderlichen Anführungszeichen unterscheiden.
Sie können beispielsweise den Eingabetext in Standardanführungszeichen (") einschließen und einfache Anführungszeichen (') für eingebettete Tags verwenden – oder umgekehrt. Beide Varianten funktionieren für Unix, Linux und macOS. Für Windows müssen Sie den Eingabetext dagegen in Standardanführungszeichen einschließen und für die Tags einfache Anführungszeichen verwenden.
Unter allen Betriebssystemen können Sie den Eingabetext in Standardanführungszeichen (") einschließen und einfache Anführungszeichen (') für eingebettete Tags verwenden. Beispiel:

```
--text "<speak>Hello <break time='300ms'/> World</speak>"
```

Unter Unix, Linux und macOS können Sie auch umgekehrt vorgehen, also den Eingabetext in einfache Anführungszeichen (') einschließen und Standardanführungszeichen (") für eingebettete Tags verwenden:

```
--text '<speak>Hello <break time="300ms"/> World</speak>'
```

Das folgende AWS CLI Beispiel ist für Unix, Linux und macOS formatiert. Ersetzen Sie unter Windows den Unix-Fortsetzungszeichen mit umgekehrtem Schrägstrich (\) am Ende jeder Zeile durch ein Caret (^) und setzen Sie den Eingabetext in vollständige Anführungszeichen („) und einfache Anführungszeichen (') für interne Tags.

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

Sie können die Sprachausgabe anhören, indem Sie die resultierende `speech.mp3`-Datei mit einem Audio-Player abspielen.

Synthetisieren eines SSL-erweiterten Dokuments

Bei längeren Eingabedaten ist es wahrscheinlich komfortabler, den SSML-Inhalt in einer Datei zu speichern und den Dateinamen dann im `synthesize-speech`-Befehl anzugeben. Sie können beispielsweise Folgendes in einer Datei namens `example.xml` speichern:

```
<?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>
```

Das Attribut `xml:lang` gibt `en-US` (Englisch (USA)) als Sprache für den Eingabetext an. Weitere Informationen dazu, wie sich die Sprache des Eingabetexts und die Sprache der gewählten Stimme auf die `SynthesizeSpeech`-Operation auswirken, finden Sie unter Angabe einer anderen Sprache für bestimmte Wörter.

So führen Sie eine Datei mit SSML-Tags aus

1. Speichern Sie die SSML-Daten in einer Datei (zum Beispiel als `example.xml`).
2. Führen Sie den folgenden `synthesize-speech`-Befehl in dem Pfad aus, in dem die XML-Datei gespeichert ist. Verwenden Sie die SSML-Datei als Eingabe, indem Sie anstelle des Eingabetexts `file:\\example.xml` angeben. Weil dieser Befehl auf eine Datei verweist und nicht den eigentlichen Eingabetext enthält, werden keine Anführungszeichen verwendet.

   > **Note**
   >
   > Das folgende AWS CLI Beispiel ist für Unix, Linux und macOS formatiert. Ersetzen Sie unter Windows den umgekehrten Schrägstrich (\), das Unix-Fortsetzungszeichen, am Ende jeder Zeile durch ein Caret-Zeichen oder Zirkumflex (^).

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

```
speech.mp3
```

3. Sie können die Sprachausgabe anhören, indem Sie die resultierende `speech.mp3`-Datei mit einem Audio-Player abspielen.

Unterstützte SSML-Tags

Alle Tags mit Ausnahme von `<amazon:domain name="news">` werden für Standardstimmen unterstützt. Die Verfügbarkeit von Tags für andere Stimmen ist in der folgenden Tabelle aufgeführt.

Amazon Polly unterstützt die folgenden SSML-Tags:

Action	SSML-Tag	Verfügbarkeit neuronaler Stimmen	Verfügbarkeit von Sprachsignalen in Langform	Generative Sprachverfügbarkeit
Eine Pause hinzufügen	<break>	Vollständige Verfügbarkeit	Vollständige Verfügbarkeit	Vollständige Verfügbarkeit
Wörter hervorheben	<emphasis>	Nicht verfügbar	Nicht verfügbar	Nicht verfügbar
Angabe einer anderen Sprache für bestimmte Wörter	<lang>	Vollständige Verfügbarkeit	Vollständige Verfügbarkeit	Vollständige Verfügbarkeit
Platzieren Sie ein benutzerdefiniertes Tag in Ihrem Text	<mark>	Vollständige Verfügbarkeit	Vollständige Verfügbarkeit	Teilweise Verfügbarkeit
Eine Pause zwischen Absätzen hinzufügen	<p>	Vollständige	Vollständige	Vollständige

Action	SSML-Tag	Verfügbarkeit neuronaler Stimmen	Verfügbarkeit von Sprachsignalen in Langform	Generative Sprachverfügbarkeit
		Verfügbarkeit	Verfügbarkeit	Verfügbarkeit
Verwendung der phonetischen Aussprache	<phoneme>	Vollständige Verfügbarkeit	Vollständige Verfügbarkeit	Teilweise Verfügbarkeit
Steuerung von Lautstärke, Sprechgeschwindigkeit und Tonhöhe	<prosody>	Teilweise Verfügbarkeit	Teilweise Verfügbarkeit	Teilweise Verfügbarkeit
Einstellung einer maximalen Dauer für synthetisierte Sprache	<prosody amazon:max-duration>	Nicht verfügbar	Nicht verfügbar	Nicht verfügbar
Eine Pause zwischen Sätzen hinzufügen	<s>	Vollständige Verfügbarkeit	Vollständige Verfügbarkeit	Vollständige Verfügbarkeit
Steuert, wie bestimmte Arten von Wörtern gesprochen werden	<say-as>	Teilweise Verfügbarkeit	Vollständige Verfügbarkeit	Vollständige Verfügbarkeit
Identifizieren von durch SSL erweitertem Text	<speak>	Vollständige Verfügbarkeit	Vollständige Verfügbarkeit	Vollständige Verfügbarkeit

Action	SSML-Tag	Verfügbarkeit neuronaler Stimmen	Verfügbarkeit von Sprachsignalen in Langform	Generative Sprachverfügbarkeit
Aussprache von Akronymen und Abkürzungen	<sub>	Vollständige Verfügbarkeit	Vollständige Verfügbarkeit	Vollständige Verfügbarkeit
Verbesserung der Aussprache durch Spezifizierung von Wortarten	<w>	Vollständige Verfügbarkeit	Vollständige Verfügbarkeit	Vollständige Verfügbarkeit
Das Geräusch des Atems hinzufügen	<amazon:auto-breaths>	Nicht verfügbar	Nicht verfügbar	Nicht verfügbar
Sprechstil von Newscaster	<amazon:domain name="news">	Nur neuronale Stimmen auswählen	Nicht verfügbar	Nicht verfügbar
Komprimierung des Dynamikbereichs wird hinzugefügt	<amazon:effect name="drc">	Vollständige Verfügbarkeit	Vollständige Verfügbarkeit	Nicht verfügbar
Leise sprechen	<amazon:effect phonation="soft">	Nicht verfügbar	Nicht verfügbar	Nicht verfügbar
Steuerung der Klangfarbe	<amazon:effect > vocal-tract-length	Nicht verfügbar	Nicht verfügbar	Nicht verfügbar
Flüstern	<amazon:effect name="whispered">	Nicht verfügbar	Nicht verfügbar	Nicht verfügbar

> **Note**
>
> Wenn Sie SSML-Tags im Standard-, Neural- oder Langformformat verwenden, die nicht unterstützt werden, wird eine Fehlermeldung angezeigt.

Identifizieren von durch SSL erweitertem Text

<speak>

Dieses Tag wird von generativen, langformatigen, neuronalen und standardmäßigen TTS-Formaten unterstützt.

Das `<speak>` Tag ist das Stammelement des gesamten Amazon Polly SSML-Textes. Der gesamte Text mit SSML-Tags muss in ein Paar <speak>-Tags eingeschlossen werden.

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

Eine Pause hinzufügen

<break>

Dieses Tag wird von generativen, langformatigen, neuronalen und standardmäßigen TTS-Formaten unterstützt.

Verwenden Sie das Tag <break>, um Ihrem Text eine Pause hinzuzufügen. Sie können eine Pause auf Grundlage der Stärke (entspricht der Pause nach einem Komma, Satz oder Absatz) oder den Wert auf eine bestimmte Dauer in Sekunden oder Millisekunden festlegen. Wenn Sie kein Attribut zur Bestimmung der Pausenlänge angeben, verwendet Amazon Polly die Standardeinstellung, d. h.`<break strength="medium"/>`, die nach einem Komma eine Pause um die Länge einer Pause hinzufügt.

Werte des Attributs `strength`:

- `none`: Keine Pause. Verwenden Sie `none`, um standardmäßig auftretende Pausen – z. B. nach einem Punkt – zu entfernen.
- `x-weak`: Hat die gleiche Wirkung wie `none`, keine Pause.

- `weak`: Legt eine Pause derselben Dauer wie die Pause nach einem Komma fest.
- `medium`: Hat die gleiche Wirkung wie `weak`.
- `strong`: Legt eine Pause derselben Dauer wie die Pause nach einem Satz fest.
- `x-strong`: Legt eine Pause derselben Dauer wie die Pause nach einem Absatz fest.

Werte des Attributs `time`:

- *`[number]`*`s`: Dauer der Pause in Sekunden. Die maximale Dauer ist `10s`.
- *`[number]`*`ms`: Dauer der Pause in Millisekunden. Die maximale Dauer ist `10000ms`.

Beispiel:

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

Wenn Sie kein Attribut mit dem `break`-Tag verwenden, variiert das Ergebnis je nach Text:

- Wenn sich neben dem `break`-Tag keine anderen Satzzeichen befinden, wird eine `<break strength="medium"/>` (Pause in Komma-Länge) erstellt.
- Wenn sich das Tag neben einem Komma befindet, wird es zu einer `<break strength="strong"/>` (Pause in Satz-Länge).
- Wenn sich das Tag neben einem Punkt befindet, wird es zu einer `<break strength="x-strong"/>` (Pause in Absatz-Länge).

Wörter hervorheben

<emphasis>

Dieses Tag wird nur vom Standard-TTS-Format unterstützt.

Verwenden Sie das Tag <emphasis>, um Wörter zu betonen. Die Betonung von Wörtern wirkt sich auf Sprechgeschwindigkeit und -lautstärke aus. Durch mehr Betonung spricht Amazon Polly den Text lauter und langsamer. Bei weniger Betonung wird leiser und schneller gesprochen. Die Stärke der Betonung geben Sie mit dem Attribut `level` an.

Werte des Attributs `level`:

- `Strong`: Erhöht die Lautstärke und verlangsamt die Sprechgeschwindigkeit, sodass die Sprachausgabe lauter und langsamer erfolgt.
- `Moderate`: Erhöht die Lautstärke und verlangsamt die Sprechgeschwindigkeit in geringerem Umfang als `strong`. `Moderate` ist die Standardeinstellung.
- `Reduced`: Verringert die Lautstärke und beschleunigt die Sprechgeschwindigkeit. Die Sprachausgabe ist weicher und schneller.

> **Note**
>
> Die normale Sprechgeschwindigkeit und -lautstärke liegen zwischen `moderate` und `reduced`.

Beispiel:

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

Angabe einer anderen Sprache für bestimmte Wörter

<lang>

Dieses Tag wird von generativen, langformatigen, neuronalen und standardmäßigen TTS-Formaten unterstützt. Bei generativen Stimmen kann das `<lang>` Tag nur bei ganzen Sätzen verwendet werden.

Mit dem Tag **<lang>** können Sie eine andere Sprache für ein Wort, eine Wendung oder einen Satz angeben. Fremdsprachige Wörter und Wendungen werden in der Regel besser gesprochen, wenn sie in ein Paar `<lang>`-Tags eingeschlossen werden. Verwenden Sie zum Angeben der Sprache das Attribut `xml:lang`. Eine vollständige Liste der verfügbaren Sprachen finden Sie unter Sprachen bei Amazon Polly.

Sofern Sie nicht das Tag `<lang>` anwenden, werden alle Wörter im Eingabetext in der Sprache der Stimme gesprochen, die mit `voice-id` angegeben wurde. Wenn Sie das Tag `<lang>` anwenden, werden die Wörter in jener Sprache gesprochen.

Wenn es sich beispielsweise um Joanna `voice-id` handelt (die US-Englisch spricht), spricht Amazon Polly Folgendes mit der Stimme von Joanna ohne französischen Akzent:

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

Wenn Sie die Joanna-Stimme mit dem `<lang>` Tag verwenden, spricht Amazon Polly den Satz mit Joanna-Stimme in Französisch mit amerikanischem Akzent:

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

Da Joanna keine französische Muttersprachlerin ist, basiert die Aussprache auf ihrer Muttersprache, also US-Englisch. Eine Person mit perfekter französischer Aussprache würde beispielsweise das Wort français mit einem uvularen Vibrant (/R/) sprechen. Joannas Stimme (US-Englisch) spricht dieses Phonem dagegen wie /r/.

Wenn Sie das `voice-id` von Giorgio, der Italienisch spricht, mit dem folgenden Text verwenden, spricht Amazon Polly den Satz in Giorgios Stimme mit italienischer Aussprache:

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

Wenn Sie dieselbe Stimme mit dem folgenden `<lang>` Tag verwenden, spricht Amazon Polly Bruce Springsteen auf Englisch mit italienischem Akzent aus:

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

Dieses Tag kann auch als Ersatz für die optionale Option bei der Sprachsynthese verwendet werden. DefaultLangCode In diesem Fall ist es jedoch erforderlich, dass Sie Ihren Text mit SSML formatieren.

Platzieren Sie ein benutzerdefiniertes Tag in Ihrem Text

<mark>

Dieses Tag wird von Langform-, neuronalen und Standard-TTS-Formaten unterstützt. Dieses Tag hat bei generativen Stimmen keine Wirkung, da Sprachzeichen für generative Stimmen nicht verfügbar sind.

<mark>Um ein benutzerdefiniertes Tag in den Text einzufügen, verwenden Sie das Tag. Amazon Polly ergreift keine Aktion für das Tag, gibt jedoch die Position des Tags in den SSML-Metadaten zurück. Bei diesem Tag kann es sich um eine beliebige hervorzuhebende Information handeln, sofern das folgende Format eingehalten wird:

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

Beispiel: Der Tag-Name lautet "animal" und der Eingabetext:

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

Amazon Polly gibt möglicherweise die folgenden SSML-Metadaten zurück:

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

Eine Pause zwischen Absätzen hinzufügen

<p>

Dieses Tag wird von generativen, langformatigen, neuronalen und standardmäßigen TTS-Formaten unterstützt.

Mit dem Tag <p> können Sie eine Pause zwischen Absätzen im Text einfügen. Mit diesem Tag wird eine längere Pause als die eingefügt, die Muttersprachler üblicherweise nach Kommas oder am Satzende einfügen. Schließen Sie den Absatz in das Tag <p> ein:

```
<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>
```

Dies entspricht der Angabe einer Pause mit <break strength="x-strong"/>.

Verwendung der phonetischen Aussprache

<phoneme>

Dieses Tag wird von Langform-, neuronalen und Standard-TTS-Formaten unterstützt.

<phoneme>Verwenden Sie das Tag, damit Amazon Polly die phonetische Aussprache für einen bestimmten Text verwendet.

Für das Tag <phoneme> sind zwei Attribute erforderlich. Sie zeigen das von Amazon Polly verwendete phonetische Alphabet und die phonetischen Symbole der korrigierten Aussprache an:

- `alphabet`
  - `ipa`: Gibt an, dass das IPA (International Phonetic Alphabet) verwendet wird.
  - `x-sampa`: Gibt an, dass X-SAMPA (Extended Speech Assessment Methods Phonetic Alphabet) verwendet wird.
- ph
  - Gibt die phonetischen Symbole für die Aussprache an. Weitere Informationen finden Sie unter Sprachen bei Amazon Polly.

Mit dem <phoneme> Tag verwendet Amazon Polly die durch das ph Attribut angegebene Aussprache anstelle der Standardaussprache, die standardmäßig der Sprache zugeordnet ist, die von der ausgewählten Stimme verwendet wird.

Das Wort „pecan“ kann beispielsweise auf zwei Arten ausgesprochen werden. Im folgenden Beispiel wird „pecan“ in jeder Zeile eine andere Aussprache zugewiesen. Amazon Polly spricht Pekannuss wie in den ph Attributen angegeben aus, anstatt die Standardaussprache zu verwenden.

International Phonetic Alphabet (IPA)

```
<speak>
```

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

Extended Speech Assessment Methods Phonetic Alphabet (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>
```

Mandarin-Chinesisch verwendet Pinyin für die phonetische Aussprache..

Pinyin

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

Japanisch verwendet Yomigana und Aussprache Kana.

Yomigana

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

Aussprache Kana

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

Steuerung von Lautstärke, Sprechgeschwindigkeit und Tonhöhe

<prosody>

Die Attribute des Prosody-Tags werden von den standardmäßigen TTS-Stimmen vollständig unterstützt. Generative, neuronale und Langform-Stimmen unterstützen die `rate` Attribute `volume` und, aber nicht das Attribut. `pitch` Bei generativen Stimmen kann das Prosodie-Tag nur für ganze Sätze verwendet werden.

Mit dem `prosody`-Tag können Sie Lautstärke, Geschwindigkeit und Tonlage Ihrer gewählten Stimme steuern.

Lautstärke, Sprechgeschwindigkeit und Tonlage sind von der jeweils gewählten Stimme abhängig. Neben den Unterschieden der Stimmen für verschiedene Sprachen gibt es auch Unterschiede zwischen Stimmen, die dieselbe Sprache sprechen. Hieraus erklärt sich, dass es trotz zwischen den Sprachen ähnlicher Attribute klare Unterschiede von Sprache zu Sprache gibt. Absolute Werte existieren nicht.

Das Tag `prosody` hat drei Attribute, für die jeweils unterschiedliche Werte verfügbar sind. Jedes Attribut verwendet die gleiche Syntax:

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

- `volume`
  - `default`: Setzt die Lautstärke auf den Standardwert für die aktuelle Stimme zurück.
  - `silent`, `x-soft`, `soft`, `medium`, `loud`, `x-loud`: Legt die Lautstärke auf einen vordefinierten Wert für die aktuelle Stimme fest.
  - `+ndB`, `-ndB`: Ändert die Lautstärke relativ zum aktuellen Pegel. Ein Wert von `+0dB` bedeutet keine Änderung, `+6dB` bedeutet in etwa eine Verdoppelung der aktuellen Lautstärke und `-6dB` bedeutet ungefähr eine Halbierung der aktuellen Lautstärke.

  Sie können die Lautstärke für eine Passage beispielsweise folgendermaßen einstellen:

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

  Sie können die Lautstärke auch folgendermaßen festlegen:

  ```
  <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` Stellt die Tonhöhe auf einen vordefinierten Wert für die ausgewählte Stimme ein.
  - n%: Eine Änderung der Sprechgeschwindigkeit um einen Prozentsatz (nicht negativ). Beispiel: Ein Wert von 100 % bedeutet, dass die Sprechgeschwindigkeit unverändert bleibt. Ein Wert von 200 % bedeutet, dass die Sprechgeschwindigkeit verdoppelt, und ein Wert von 50 %, dass die Sprechgeschwindigkeit halbiert wird. Der Wert kann zwischen 20 und 200 % liegen.

  Sie können die Sprechgeschwindigkeit für eine Passage beispielsweise folgendermaßen einstellen:

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

  Sie können die Lautstärke auch folgendermaßen festlegen:

```
<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`: Setzt die Tonlage auf den Standardwert für die aktuelle Stimme zurück.
  - `x-low`, `low`, `medium`, `high`, `x-high`: Legt die Tonlage auf einen vordefinierten Wert für die aktuelle Stimme fest.
  - `+n%` oder `-n%`: passt die Tonhöhe um einen relativen Prozentsatz an. Beispiel: Ein Wert von `+0%` bedeutet keine Änderung der Baseline-Tonhöhe, `+5%` führt zu einer etwas höheren Baseline-Tonhöhe und `-5%` führt zu einer etwas niedrigeren Baseline-Tonhöhe.

  Sie können die Tonlage für eine Passage beispielsweise folgendermaßen einstellen:

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

```
</speak>
```

Sie können die Lautstärke auch folgendermaßen festlegen:

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

Das Tag **<prosody>** muss mindestens ein Attribut, kann aber auch mehrere enthalten.

```
<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>
```

Es kann zudem folgendermaßen mit verschachtelten Tags kombiniert werden:

```
<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>
```

> **Note**
>
> Derzeit `<prosody>` ist es teilweise für die generativen Stimmen verfügbar.

Einstellung einer maximalen Dauer für synthetisierte Sprache

<prosody amazon:max-duration>

Dieses Tag wird derzeit nur vom Standard-TTS-Format unterstützt.

Um zu steuern, wie lange eine Sprachausgabe dauern soll, wenn sie generiert wird, verwenden Sie das `<prosody>`-Tag mit dem `amazon:max-duration`-Attribut.

Die Dauer der Sprachsynthese variiert je nach gewählter Stimme geringfügig. Dies erschwert die Abstimmung von generierter Sprache mit Visualisierungen oder anderen Aktivitäten, die ein präzises Timing erfordern. Dieses Problem tritt verstärkt bei Übersetzungsanwendungen auf, da die Zeit, die benötigt wird, um bestimmte Phrasen zu sagen, je nach Sprache stark variieren kann.

Das `<prosody amazon:max-duration>`-Tag passt die synthetisierte Sprache an die gewünschte Dauer an.

Dieses Tag verwendet folgende Syntax:

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

Mit dem `<prosody amazon:max-duration>`-Tag können Sie die Dauer in Sekunden oder Millisekunden festlegen:

- *n*`s`: maximale Dauer in Sekunden.
- *n*`ms`: maximale Dauer in Millisekunden.

Beispiel: Der folgende gesprochene Text hat eine maximale Dauer von 2 Sekunden:

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

Wenn Text innerhalb des Tags platziert wird, überschreitet er die angegebene Dauer nicht. Wenn die gewählte Stimme oder Sprache normalerweise länger als diese Dauer dauern würde, beschleunigt Amazon Polly die Sprache, sodass sie in die angegebene Dauer passt.

Wenn die angegebene Dauer länger ist, als es dauert, den Text mit normaler Geschwindigkeit zu lesen, liest Amazon Polly die Sprache normal vor. Es verlangsamt weder die Sprachausgabe noch werden Stilleperioden hinzugefügt. Die resultierende Audioausgabe ist also kürzer als angefordert.

> **Note**
>
> Amazon Polly erhöht die Geschwindigkeit nicht mehr als das Fünffache der normalen Geschwindigkeit. Wenn Text schneller gesprochen wird, ergibt er in der Regel keinen Sinn. Wenn eine Sprachausgabe auch bei maximaler Beschleunigung nicht in die angegebene

Dauer passt, wird das Audiomaterial beschleunigt, ist dann jedoch länger als die angegebene Dauer.

Sie können einen einzelnen Satz oder mehrere Sätze innerhalb eines `<prosody amazon:max-duration>`-Tags und mehrere `<prosody amazon:max-duration>`-Tags in Ihrem Text verwenden.

Beispiel:

```
<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>
```

Die Verwendung des `<prosody amazon:max-duration>` Tags kann die Latenz erhöhen, wenn Amazon Polly synthetisierte Sprache zurückgibt. Der Grad der Latenz hängt von der Passage und ihrer Länge ab. Wir empfehlen die Verwendung von Text aus relativ kurzen Textpassagen.

Einschränkungen

Es gibt Einschränkungen sowohl bei der Verwendung des `<prosody amazon:max-duration>`-Tags als auch bei der Funktion des Tags mit anderen SSML-Tags:

- Der Text innerhalb eines `<prosody amazon:max-duration>`-Tags kann nicht mehr als 1 500 Zeichen betragen.

- Sie können keine <prosody amazon:max-duration>-Tags verschachteln. Wenn Sie ein <prosody amazon:max-duration> Etikett in ein anderes einfügen, ignoriert Amazon Polly das innere Etikett.

  Im folgenden Beispiel wird das <prosody amazon:max-duration="5s">-Tag ignoriert:

```
<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>
```

- Es ist nicht möglich, die <prosody>-Tags mit dem rate-Attribut innerhalb eines <prosody amazon:max-duration>-Tags zu verwenden. Denn beide beeinflussen die Geschwindigkeit, mit der der Text gesprochen wird.

  Im folgenden Beispiel ignoriert Amazon Polly das <prosody rate="2"> Tag:

```
<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>
```

Pausiert und **max-duration**

Bei der Verwendung Ihres `max-duration`-Tags können Sie weiterhin Pausen in Ihren Text einfügen. Amazon Polly berücksichtigt jedoch die Länge der Pause bei der Berechnung der maximalen Sprachdauer. Darüber hinaus behält Amazon Polly die kurzen Pausen bei, die auftreten, wenn Kommas und Punkte innerhalb einer Passage stehen, und schließt die maximale Dauer ein.

Beispiel: Im folgenden Block kommen Pausen von 600 Millisekunden und die durch Kommata und Punkte verursachten Pausen innerhalb der 8-Sekunden-Sprachausgabe vor:

```
<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>
```

Eine Pause zwischen Sätzen hinzufügen

<s>

Dieses Tag wird von generativen, langformatigen, neuronalen und standardmäßigen TTS-Formaten unterstützt.

Mit dem Tag <s> können Sie eine Pause zwischen Zeilen oder Sätzen im Text einfügen. Die Verwendung dieses Tags hat die gleiche Wirkung wie:

- Beenden eines Satzes mit einem Punkt (.)
- Angeben einer Pause mit `<break strength="strong"/>`

Im Unterschied zum Tag `<break>` schließt das Tag **<s>** den Satz ein. Das ist beim Generieren von Sprachausgabe nützlich, deren Eingabetext zeilen- statt satzweise angeordnet ist, also beispielsweise bei Gedichten.

Im folgenden Beispiel sorgt das Tag <s> für eine kurze Pause nach dem ersten und zweiten Satz. Der letzte Satz hat kein <s>-Tag. Es folgt aber trotzdem eine kurze Pause, weil er mit einem Punkt endet.

```
<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>
```

Steuert, wie bestimmte Arten von Wörtern gesprochen werden

<say-as>

Das `<say-as>` Tag wird von generativen, langformatigen, neuronalen und standardmäßigen TTS-Engines unterstützt. Beachten Sie jedoch, dass, wenn Amazon Polly eine neuronale Stimme verwendet und zur Laufzeit auf das `<say-as>` Tag mit der `characters` Option trifft, der betroffene Satz mit der entsprechenden Standardstimme synthetisiert wird. Der betroffene Satz wird jedoch weiterhin so abgerechnet, als ob er eine neuronale Stimme verwendet.

Verwenden Sie das `<say-as>` Tag mit dem `interpret-as` Attribut, um Amazon Polly mitzuteilen, wie bestimmte Zeichen, Wörter und Zahlen ausgesprochen werden sollen. Auf diese Weise können Sie zusätzlichen Kontext angeben, um Unklarheiten darüber zu vermeiden, wie Amazon Polly den Text wiedergeben soll.

Das `<say-as>` Tag verwendet ein Attribut`interpret-as`, das eine Reihe möglicher verfügbarer Werte verwendet. Jeder dieser Werte verwendet die gleiche Syntax:

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

Die folgenden Werte können mit `interpret-as` verwendet werden:

- `characters`oder`spell-out`: Buchstabiert jeden Buchstaben des Textes wie in a-b-c.

  > **Note**
  >
  > Diese Option wird derzeit für neuronale Stimmen nicht unterstützt. Wenn Sie eine neuronale Stimme verwenden und dieser SSML-Code von Amazon Polly zur Laufzeit erkannt wird, wird der betroffene Satz mit der entsprechenden Standardstimme synthetisiert. Bitte beachten Sie jedoch, dass dieser Satz weiterhin so abgerechnet wird, als ob er eine neuronale Stimme verwendet.

- `cardinal` oder `number`: Interpretiert den numerischen Text als Kardinalzahl (z. B. 1.234).

- `ordinal`: Interpretiert den numerischen Text als Ordnungszahl (z. B. 1.234).
- `digits`: Spricht jede Ziffer einzeln (wie in 1-2-3-4).
- `fraction`: Interpretiert numerischen Text als Bruch. Dies funktioniert sowohl für gemeine Brüche wie 3/20 als auch für gemischte Brüche wie 2 ½. Weitere Informationen hierzu finden Sie unten.
- `unit`: Interpretiert einen numerischen Text als Messwert. Der Wert sollte eine Zahl oder ein Bruch gefolgt von einer Einheit ohne Leerstelle wie in `1/2inch` oder nur eine Einheit wie in `1meter` sein.
- `date`: Interpretiert den Text als Datum. Das Datumsformat muss durch das Formatattribut festgelegt werden. Weitere Informationen hierzu finden Sie unten.
- `time`: interpretiert den numerischen Text als Dauer in Minuten und Sekunden (z. B. `1'21"`).
- `address`: Interpretiert den Text als Teil einer Angabe von Straße und Hausnummer.
- `expletive`: Der im Tag eingeschlossene Inhalt wird durch einen Piepton überdeckt.
- `telephone`: Interpretiert den numerischen Text als sieben- oder zehnstellige Telefonnummer, z. B. `2025551212`. Sie können diesen Wert auch für Nebenstellen wie in `2025551212x345` verwenden. Weitere Informationen hierzu finden Sie unten.

> **Note**
>
> Derzeit ist die Option `telephone` nicht für alle Sprachen verfügbar. Es ist jedoch für Stimmen verfügbar, die englische Sprachvarianten (en-AU, en-GB, en-IN, en-US und en-GB-WLS), spanische Sprachvarianten (es-ES, es-MX und es-US), französische Sprachvarianten (fr-FR und fr-CA) und portugiesische Varianten (pt-BR und pt-PT) sowie Deutsch (de-DE), Italienisch (it-IT), Japanisch (ja-JP) und Russisch (ru-RU) sprechen). Es sollte auch beachtet werden, dass Sprachen wie Arabisch (arb) in einigen Fällen die als Telefonnummer eingestellte Nummer automatisch verarbeiten und das `telephone` SSML-Tag daher nicht wirklich implementieren.

Bruchzahlen

Amazon Polly interpretiert Werte innerhalb des `say-as` Tags, die das `interpret-as="fraction"` Attribut enthalten, als gemeinsame Brüche. Im Folgenden wird die Syntax für Bruchzahlen beschrieben.

- Bruchzahlen

  Syntax:*`cardinal number`*/*`cardinal number`*, z. B. 2/9.

Beispiel: `<say-as interpret-as="fraction">2/9</say-as>` wird ausgesprochen als "two ninth".

- Nicht negative gemischte Nummer

  Syntax: *`cardinal number`* +*`cardinal number`*/*`cardinal number`*, z. B. 3+1/2.

  Beispiel: `<say-as interpret-as="fraction">3+1/2</say-as>` wird ausgesprochen als "three and a half".

> **Note**
>
> +Zwischen „3" und „1/2" muss ein Wert stehen. Amazon Polly unterstützt keine gemischten Zahlen ohne das+, z. B. „3 1/2“.

Datumsangaben

Wenn `interpret-as` auf `date` gesetzt ist, müssen Sie auch das Datumsformat angeben.

Für dieses Tag gilt folgende Syntax:

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

Beispiel:

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

Die folgenden Formate können für das Attribut `date` angegeben werden.

- `mdy`: Month-day-year.
- `dmy`: Day-month-year.
- `ymd`: Year-month-day.
- `md`: Monat-Tag.
- `dm`: Tag-Monat.
- `ym`: Jahr-Monat.

- `my`: Monat-Jahr.
- `d`: Tag.
- `m`: Monat.
- `y`: Jahr.
- `yyyymmdd`:. Year-month-day Wenn Sie dieses Format verwenden, können Sie Amazon Polly mithilfe von Fragezeichen veranlassen, Teile des Datums zu überspringen.

  Amazon Polly gibt beispielsweise Folgendes als „22. September“ wieder:

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

  `Format` ist nicht erforderlich.

Telefonnummer

Amazon Polly versucht, den von Ihnen bereitgestellten Text anhand der Textformatierung auch ohne das `<say-as>` Tag korrekt zu interpretieren. Wenn Ihr Text beispielsweise „202-555-1212“ enthält, interpretiert Amazon Polly ihn als 10-stellige Telefonnummer und sagt jede Ziffer einzeln, mit einer kurzen Pause für jeden Gedankenstrich. In diesem Fall müssen Sie `<say-as interpret-as="telephone">` nicht verwenden. Wenn Sie jedoch den Text „2025551212“ angeben und möchten, dass Amazon Polly ihn als Telefonnummer sagt, geben Sie an. `<say-as interpret-as="telephone">`

Die Logik zur Interpretation der einzelnen Elemente ist sprachspezifisch. Die Aussprache von Telefonnummern unterscheidet sich beispielsweise zwischen US-amerikanischem und britischem Englisch (in Großbritannien werden aufeinanderfolgende gleiche Ziffern zusammengefasst, z. B. "double five" oder "triple four"). Sie können das folgende Beispiel mit einer US-amerikanischen und einer britischen Stimme testen, um den Unterschied zu hören:

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

Aussprache von Akronymen und Abkürzungen

<sub>

Dieses Tag wird von generativen, langformatigen, neuronalen und standardmäßigen TTS-Formaten unterstützt.

Verwenden Sie das `<sub>`-Tag mit dem `alias`-Attribut, um gewählten Text – z. B. ein Akronym oder eine Abkürzung – durch ein anderes Wort (oder eine andere Aussprache) zu ersetzen.

Es gilt folgende Syntax:

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

Im folgenden Beispiel wird der Name "Mercury" anstelle des chemischen Symbols für das Element gesprochen, um den Audioinhalt verständlicher zu machen.

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

Verbesserung der Aussprache durch Spezifizierung von Wortarten

<w>

Dieses Tag wird von generativen, langformatigen, neuronalen und standardmäßigen TTS-Formaten unterstützt.

Sie können das Tag **<w>** verwenden, um die Aussprache von Wörtern anzupassen, indem Sie die Wortart oder eine alternative Bedeutung angeben. Dies erfolgt mithilfe des Attributs `role`.

Dieses Tag verwendet folgende Syntax:

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

Folgende Werte können für das Attribut `role` angegeben werden:

So geben Sie die Wortart an:

- `amazon:VB`: Das Wort wird als Verb (in der Gegenwartsform) interpretiert.
- `amazon:VBD`: interpretiert das Wort als Verb in der Vergangenheitsform.
- `amazon:DT`: interpretiert das Wort als Determinator.

- `amazon:IN`: interpretiert das Wort als Präposition.
- `amazon:JJ`: interpretiert das Wort als Adjektiv.
- `amazon:NN`: interpretiert das Wort als Substantiv.

Beispiel: Je nach Wortart variiert die Aussprache des Wortes „read“ im US-Englischen in Abhängigkeit vom Tag:

```
<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>
```

Um eine bestimmte Bedeutung zu spezifizieren:

- `amazon:DEFAULT`: verwendet die Standardbedeutung des Wortes.
- `amazon:SENSE_1`: Der nicht standardmäßige Wortsinn wird verwendet (sofern vorhanden). Beispiel: Das Substantiv „bass“ wird je nach Bedeutung anders ausgesprochen. Die Standardbedeutung ist die tiefste Tonlage in der Musik. Die alternative Bedeutung ist eine Spezies von Süßwasserfischen, die auch als „bass“ bezeichnet, aber anders ausgesprochen wird. Durch `<w role="amazon:SENSE_1">bass</w>` wird in der Sprachausgabe die nichtstandardmäßige Aussprache (für den Süßwasserfisch) verwendet.

Dieser Unterschied in Aussprache und Bedeutung ist hörbar, wenn Sie Folgendes zusammenfassen:

```
<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**
>
> Einige Sprachen weisen möglicherweise eine andere Auswahl unterstützter Sprachelemente auf.

Das Geräusch des Atems hinzufügen

<amazon:breath> und <amazon:auto-breaths>

Dieses Tag wird nur vom Standard-TTS-Format unterstützt.

Natürlich klingende Sprache besteht aus richtig gesprochenen Wörtern und Atemgeräuschen. Wenn Sie der synthetisierten Sprachausgabe Atemgeräusche hinzufügen, klingt sie natürlicher. Die Tags `<amazon:breath>` und `<amazon:auto-breaths>` stellen Atemgeräusche bereit. Ihnen stehen folgende Optionen zur Verfügung:

- Manueller Modus: Sie legen Position, Dauer und Lautstärke des Atemgeräusches im Text fest
- Automatisierter Modus: Amazon Polly fügt automatisch Atemgeräusche in die Sprachausgabe ein
- Gemischter Modus: Sowohl Sie als auch Amazon Polly fügen Atemgeräusche hinzu

Manueller Modus

Im manuellen Modus platzieren Sie das Tag `<amazon:breath/>` im Eingabetext an der Stelle, an der das Atemgeräusch hörbar werden soll. Sie können Dauer und Lautstärke des Atemgeräusches mit den Attributen `duration` und `volume` festlegen:

- `duration`: Legt die Dauer des Atemgeräusches fest. Folgende Werte sind zulässig: `default`, `x-short`, `short`, `medium`, `long`, `x-long`. Der Standardwert ist `medium`.
- `volume`: Legt die Lautstärke des Atemgeräusches fest. Folgende Werte sind zulässig: `default`, `x-soft`, `soft`, `medium`, `loud`, `x-loud`. Der Standardwert ist `medium`.

> **Note**
>
> Die genaue Länge und Lautstärke der einzelnen Attributwerte hängt von der jeweils verwendeten Amazon Polly Polly-Stimme ab.

Sie können ein Atemgeräusch mit Standardwerten festlegen, indem Sie `<amazon:breath/>` ohne Attribute verwenden.

Um beispielsweise Dauer und Lautstärke eines Atemgeräusches mit Attributen festzulegen, verwenden Sie folgende Attributwerte:

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

Für ein Atemgeräusch mit Standardwerten verwenden Sie einfach das Tag:

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

Sie können folgendermaßen Atemgeräusche in eine Textpassage einfügen:

```
<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>
```

Automatischer Modus

Im automatisierten Modus verwenden Sie das `<amazon:auto-breaths>` Tag, um Amazon Polly anzuweisen, in geeigneten Intervallen automatisch Atemgeräusche zu erzeugen. Sie können die Häufigkeit der Intervalle sowie Lautstärke und Dauer einstellen. Platzieren Sie das Tag `</amazon:auto-breaths>` am Anfang und das entsprechende schließende Tag am Ende des Textes, für den Sie automatisierte Atemgeräusche generieren möchten.

> **Note**
>
> Im Unterschied zum Tag `<amazon:breath/>` für den manuellen Modus ist für `<amazon:auto-breaths>` ein schließendes Tag (`</amazon:auto-breaths>`) erforderlich.

Sie können die folgenden optionalen Attribute mit dem Tag `<amazon:auto-breaths>` verwenden:

- `volume`: Legt die Lautstärke der Atemgeräusche fest. Folgende Werte sind zulässig: `default`, `x-soft`, `soft`, `medium`, `loud`, `x-loud`. Der Standardwert ist `medium`.
- `frequency`: Steuert, wie oft Atemgeräusche im Text generiert werden. Folgende Werte sind zulässig: `default`, `x-low`, `low`, `medium`, `high`, `x-high`. Der Standardwert ist `medium`.
- `duration`: Legt die Dauer des Atemgeräusches fest. Folgende Werte sind zulässig: `default`, `x-short`, `short`, `medium`, `long`, `x-long`. Der Standardwert ist `medium`.

Standardmäßig hängt die Häufigkeit der Atemgeräusche vom Eingabetext ab. Atemgeräusche treten häufig nach Kommas und Punkten auf.

Die folgenden Beispiele demonstrieren die Verwendung des Tags `<amazon:auto-breaths>`. Um zu entscheiden, welche Optionen Sie für Ihre Inhalte verwenden möchten, kopieren Sie die entsprechenden Beispiele auf die Amazon Polly Polly-Konsole und hören Sie sich die Unterschiede an.

- Automatischer Modus ohne optionale Parameter

```
<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>
```

- Automatischer Modus mit Lautstärkeregelung: Für nicht angegebene Parameter (`duration` und `frequency`) werden die Standardwerte (`medium`) verwendet.

```
<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>
```

- Automatischer Modus mit Häufigkeitsregelung: Für nicht angegebene Parameter (`duration` und `volume`) werden die Standardwerte (`medium`) verwendet.

```
<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>
```

- Automatischer Modus mit mehreren Parametern: Für den nicht spezifizierten `Duration` Parameter verwendet Amazon Polly den Standardwert ()`medium`.

```
<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>
```

Sprechstil von Newscaster

<amazon:domain name="news">

Der Newscaster-Stil ist nur für die Stimmen von Matthew oder Joanna verfügbar, die nur in amerikanischem Englisch (en-US), Lupe in US-Spanisch (es-US) und Amy in britischem Englisch (en-GB) verfügbar sind. Er wird nur für das Format `Neural` unterstützt.

Um den Newscaster-Stil zu verwenden, verwenden Sie SSML-Tags und die folgende Syntax:

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

Beispielsweise könnten Sie den Newscaster-Stil mit der Stimme von Amy wie folgt verwenden:

```
<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>
```

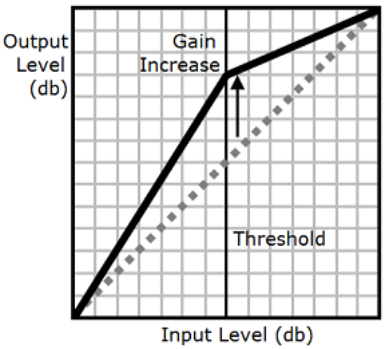
Komprimierung des Dynamikbereichs wird hinzugefügt

<amazon:effect name="drc">

Dieses Tag wird von Langform-, neuronalen und standardmäßigen TTS-Formaten unterstützt.

Je nach dem in einer Audiodatei verwendeten Text, der Sprache und der Stimme reichen die Töne von leise bis laut. Umgebungsgeräusche, wie z. B. der Klang eines sich bewegenden Fahrzeugs, können oft die leisen Töne überdecken, wodurch die Audiospur schwer zu hören ist. Um die Lautstärke bestimmter Sounds in Ihrer Audiodatei zu erhöhen, verwenden Sie den Tag für die Dynamikbereichkomprimierung (`drc`).

Das `drc`-Tag stellt einen mittleren „Lautstärke“-Schwellenwert für Ihr Audiomaterial ein und erhöht die Lautstärke (die Verstärkung) der Sounds um diesen Schwellenwert. Es wendet die größte Verstärkungszunahme an, die dem Schwellenwert am nächsten ist, und die Verstärkungszunahme wird weiter weg vom Schwellenwert verringert.



Dadurch werden die Klänge des mittleren Bereichs in einer geräuschvollen Umgebung besser hörbar, wodurch die gesamte Audiodatei klarer wird.

Der `drc`-Tag ist ein boolescher Parameter (entweder vorhanden oder nicht). Es verwendet die Syntax: `<amazon:effect name="drc">` und wird mit `</amazon:effect>` geschlossen.

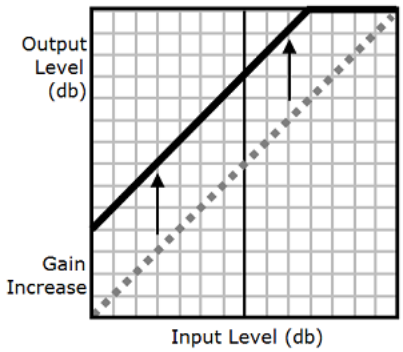
Sie können das `drc` Tag mit jeder Stimme oder Sprache verwenden, die von Amazon Polly unterstützt wird. Sie können es auf einen ganzen Abschnitt der Aufnahme oder nur für einige Wörter anwenden. Beispiel:

```
<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>
```

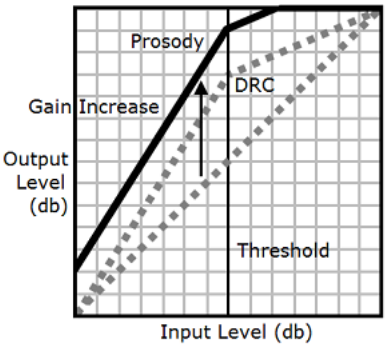
> **Note**
>
> Wenn Sie „`drc`“ in der `amazon:effect` -Syntax verwenden, wird die Groß-/Kleinschreibung beachtet.

Verwenden von **`drc`** mit dem **`prosody volume`**-Tag

Wie die folgende Grafik zeigt, erhöht der Tag `prosody volume` die Lautstärke einer gesamten Audiodatei gleichmäßig vom ursprünglichen Level (gepunktete Linie) auf einen angepassten Level (durchgezogene Linie). Um die Lautstärke bestimmter Teile der Datei weiter erhöhen, verwenden Sie den `drc`-Tag mit dem `prosody volume`-Tag. Die Kombination von Tags hat keine Auswirkungen auf die Einstellungen des Tags `prosody volume`.



Wenn Sie die `prosody volume` Tags `drc` und zusammen verwenden, wendet Amazon Polly das `drc` Tag zuerst an, wodurch die Geräusche im mittleren Bereich (diejenigen, die sich in der Nähe des Schwellenwerts befinden) verstärkt werden. Dann wendet es den Tag `prosody volume` an und erhöht die Lautstärke der gesamten Audiospur weiter gleichmäßig.



Um die Tags zusammen zu verwenden, verschachteln Sie sie ineinander. Beispiel:

```
<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>
```

In diesem Text erhöht der Tag `prosody volume` die Lautstärke der gesamten Passage auf „laut“- Der Tag `drc` erhöht die Lautstärke der Mittelwerte im zweiten Satz.

> **Note**
>
> Wenn Sie die Tags `drc` und `prosody volume` zusammen verwenden, verwenden Sie XML-Standardpraktiken zum Verschachteln von Tags.

Leise sprechen

<amazon:effect phonation="soft">

Dieses Tag wird derzeit nur vom Standard-TTS-Format unterstützt.

Verwenden Sie das <amazon:effect phonation="soft">Tag, um anzugeben, dass der eingegebene Text mit einer softer-than-normal Stimme gesprochen werden soll.

Es gilt folgende Syntax:

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

Sie können dieses Tag beispielsweise folgendermaßen mit der Stimme Matthew verwenden:

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

Steuerung der Klangfarbe

<amazon:effect > vocal-tract-length

Dieses Tag wird derzeit nur vom Standard-TTS-Format unterstützt.

Timbre ist die Klangqualität einer Stimme, mit der Sie den Unterschied zwischen Stimmen erkennen können, selbst wenn sie die gleiche Tonhöhe und Lautstärke haben. Eine der wichtigsten

physiologischen Eigenschaften, die zur Sprachtimbre beiträgt, ist die Länge des Vokaltraktes. Der Vokaltrakt ist eine Lufthöhle, die sich von der Oberseite der Stimmfalten bis zum Rand der Lippen erstreckt.

Verwenden Sie das Tag, um die Klangfarbe der ausgegebenen Sprache in Amazon Polly zu steuern. `vocal-tract-length` Dieser Tag hat die Wirkung, die Länge des Vokaltrakts des Sprechers zu verändern, was wie eine Änderung der Sprechergröße klingt. Wenn Sie die `vocal-tract-length`erhöhen, klingt der Sprecher physikalisch größer. Wenn Sie es verringern, klingt der Sprecher kleiner. Sie können dieses Tag mit allen Stimmen im Amazon Text-to-Speech Polly-Portfolio verwenden.

Verwenden Sie die folgenden Werte, um das Timbre zu ändern:

- `+n%` oder `-n%`: Passt die Vokaltraktlänge um einen relativen Prozentsatz der derzeit verwendeten Stimme an. Beispiel: +4 % oder -2 %. Gültige Werte liegen zwischen 100 % und -50 %. Werte außerhalb dieses Bereichs werden abgeschnitten. Zum Beispiel klingt +111 % wie +100 % und -60 % klingt wie -50 %.
- `n%`: Ändert die Länge des Vokaltrakts auf einen absoluten Prozentsatz der Länge der aktuellen Stimme. Zum Beispiel 110 % oder 75 %. Ein absoluter Wert von 110 % entspricht einem relativen Wert von +10 %. Ein absoluter Wert von 100 % entspricht dem Standardwert für die aktuelle Stimme.

Das folgende Beispiel zeigt, wie die Länge des Vokaltrakts geändert wird, um das Timbre zu ändern:

```
<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>
```

Kombinieren von mehreren Tags

Sie können das `vocal-tract-length` Tag mit jedem anderen SSML-Tag kombinieren, das von Amazon Polly unterstützt wird. Da Timbre (Vokaltraktlänge) und Tonhöhe eng miteinander verbunden sind, können Sie die besten Ergebnisse erzielen, wenn Sie sowohl den `vocal-tract-length` als auch den `<prosody pitch>`-Tag verwenden. Um die realistischste Stimme zu erzeugen, empfehlen wir Ihnen, unterschiedliche Prozentsätze der Änderungen für die beiden Tags zu verwenden. Experimentieren Sie mit verschiedenen Kombinationen, um die gewünschten Ergebnisse zu erzielen.

Das folgende Beispiel zeigt, wie Tags kombiniert werden.

```
<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>
```

Flüstern

<amazon:effect name="whispered">

Dieses Tag wird derzeit nur vom Standard-TTS-Format unterstützt.

Dieses Tag gibt an, dass der Eingabetext nicht normal gesprochen, sondern geflüstert werden soll. Dies kann mit allen Stimmen im Amazon Text-to-Speech Polly-Portfolio verwendet werden.

Für dieses Tag gilt folgende Syntax:

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

Beispiel:

```
<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>
```

In diesem Fall wird die von der Figur gesprochene synthetisierte Sprache geflüstert, aber der Ausdruck „sie sagte“ wird in der normalen synthetisierten Sprache der ausgewählten Amazon Polly-Stimme gesprochen.

Sie können den „Flüstereffekt“ noch verstärken, indem Sie den Satzrhythmus je nach Belieben um bis zu 10 % verlangsamen.

Beispiel:

```
<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>
```

Beim Erstellen der Sprachmarkierungen für eine Flüsterstimme muss der Audiostream diese ebenfalls enthalten, um sicherzustellen, dass die Sprachmarkierungen zum Audiostream passen.

Verwaltung von Lexika

Mit Aussprachelexika können Sie die Aussprache von Wörtern individuell anpassen. Amazon Polly bietet API-Operationen, mit denen Sie Lexika in einer AWS Region speichern können. Diese Lexika gelten dann speziell für diese bestimmte Region. Sie können eine oder mehrere der Lexika aus dieser Region verwenden, wenn Sie die Textsynthese mithilfe des `SynthesizeSpeech`-Vorgangs vornehmen. Dies gilt für das angegebene Lexikon des Eingabetexts, bevor die Synthese beginnt. Weitere Informationen finden Sie unter SynthesizeSpeech.

> **Note**
>
> Diese Lexika müssen mit der Angabe zur W3C-Empfehlung des Aussprachelexikons übereinstimmen. Weitere Informationen dazu finden Sie auf der W3C-Website unter Angaben zum Aussprachelexikon (PLS) Version 1.0.

Es folgen Beispiele für die Verwendung von Lexika mit Sprachsynthese-Engines:

- Häufige Wörter werden manchmal mit Zahlen anstelle von Buchstaben stilisiert, wie z. B. „g3t sm4rt“ (get smart). Menschen können diese Wörter korrekt lesen. Eine Text-to-Speech (TTS-) Engine liest den Text jedoch wörtlich und spricht den Namen genau so aus, wie er geschrieben ist. Hier können Sie Lexika nutzen, um die synthetisierte Sprache mithilfe von Amazon Polly anzupassen. In diesem Beispiel können Sie einen Alias (get smart) für das Wort „g3t sm4rt“ im Lexikon angeben.
- Der Text kann ein Akronym enthalten, z. B. W3C. Sie können mit einem Lexikon einen Alias für das Wort W3C definieren, sodass er vollständig in erweiterter Form gelesen wird (World Wide Web Consortium).

Lexika geben Ihnen zusätzliche Kontrolle darüber, wie Amazon Polly Wörter ausspricht, die in der ausgewählten Sprache ungewöhnlich sind. Beispielsweise können Sie die Aussprache mit einem phonetischen Alphabet angeben. Weitere Informationen dazu finden Sie auf der W3C-Website unter Angaben zum Aussprachelexikon (PLS) Version 1.0.

Themen

- Verwendung mehrerer Lexika
- Ein Lexikon hochladen

- Lexika anwenden (Sprachsynthese)
- Filterung der Lexikonliste auf der Konsole
- Lexika werden auf die Konsole heruntergeladen
- Löschen eines Lexikons

Verwendung mehrerer Lexika

Sie können bis zu fünf Lexika auf Ihren Text anwenden. Wenn dasselbe Graphem in mehr als einem Lexikon angezeigt wird, das Sie auf Ihren Text anwenden, kann die Reihenfolge, in der sie angewendet werden, eine unterschiedliche Sprachausgabe zur Folge haben. Nehmen wir zum Beispiel den folgenden Text: „Hallo, ich heiße Bob.“ und zwei Lexeme in verschiedenen Lexika, in denen jeweils das Graphem Bob.

LexA

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

LexB verwendet wird.

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

Wenn die Lexika in der Reihenfolge LexA und LexB aufgelistet werden, lautet die synthetische Sprache „Hallo, ich heiße Robert.“ Wenn sie in der Reihenfolge LexB und LexA aufgeführt werden, lautet die synthetische Sprache „Hallo, ich heiße Roland.“

Example– Anwenden von LexA vor LexB

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

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

Speech Ausgabe: „Hallo, ich heiße Robert.“

Example– Anwenden von LexB vor LexA

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

Speech Ausgabe: „Hallo, ich heiße Bobby.“

Informationen zum Anwenden von Lexika mithilfe der Amazon Polly Polly-Konsole finden Sie unter. Lexika anwenden (Sprachsynthese)

Ein Lexikon hochladen

Die von Ihnen verwendeten Lexika müssen der W3C-Empfehlung der Pronunciation Lexicon Specification (PLS) entsprechen. Weitere Informationen dazu finden Sie auf der W3C-Website unter Angaben zum Aussprachelexikon (PLS) Version 1.0.

Console - Lexicons tab

Zum Verwenden eines Aussprachelexikons müssen Sie es zunächst hochladen. Es gibt zwei Positionen auf der Konsole, von denen Sie ein Lexikon hochladen können, die Registerkarte Text-to-Speech und die Registerkarte Lexicons.

In den folgenden Prozessen wird beschrieben, wie Sie Lexika hinzufügen, die Sie verwenden können, um festzulegen, wie Wörter und Sätze ausgesprochen werden, die in der gewählten Sprache seltener vorkommen.

Um ein Lexikon über die Registerkarte Lexika hinzuzufügen

1. Melden Sie sich bei der an AWS-Managementkonsole und öffnen Sie die Amazon Polly Polly-Konsole unter https://console.aws.amazon.com/polly/.

2. Wählen Sie die Registerkarte Lexicons aus.
3. Wählen Sie Lexikon hochladen.
4. Geben Sie einen Namen für das Lexikon ein und klicken Sie dann auf Lexikondatei auswählen, um das hochzuladende Lexikon zu suchen. Sie können nur PLS-Dateien mit den Erweiterungen.pls oder .xml hochladen.
5. Wählen Sie Lexikon hochladen. Existiert bereits ein Lexikon mit demselben Namen (unabhängig davon, ob es sich um eine .pls- oder .xml-Datei handelt), wird durch das Hochladen des Lexikons das bestehende Lexikon überschrieben.

Console - TTS tab

Um text-to-Speech ein Lexikon aus der Registerkarte hinzuzufügen

1. Melden Sie sich bei der an AWS-Managementkonsole und öffnen Sie die Amazon Polly Polly-Konsole unter https://console.aws.amazon.com/polly/.
2. Wählen Sie die Registerkarte Text-to-Speech.
3. Erweitern Sie Zusätzliche Einstellungen, aktivieren Sie Aussprache anpassen und wählen Sie dann Lexikon hochladen aus.
4. Geben Sie einen Namen für das Lexikon ein und klicken Sie dann auf Lexikondatei auswählen, um das hochzuladende Lexikon zu suchen. Sie können PLS-Dateien nur mit den Erweiterungen.pls oder .xml verwenden.
5. Wählen Sie Lexikon hochladen. Existiert bereits ein Lexikon mit demselben Namen (unabhängig davon, ob es sich um eine .pls- oder .xml-Datei handelt), wird durch das Hochladen des Lexikons das bestehende Lexikon überschrieben.

AWS CLI - one lexeme

Mit Amazon Polly können Sie PutLexicon Aussprachelexika in einer bestimmten AWS Region für Ihr Konto speichern. Anschließend können Sie eine oder mehrere dieser gespeicherten Lexika in Ihrer SynthesizeSpeech-Anforderung angeben, die Sie anwenden möchten, bevor der Dienst mit der synthetischen Sprache beginnt. Weitere Informationen finden Sie unter Verwaltung von Lexika.

Erwägen Sie das folgende W3C-PLS-konforme Lexikon.

```
<?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>
```

Beachten Sie Folgendes:

- Im `<lexicon>`-Element werden die folgenden zwei Attribute angegeben:
  - Das `xml:lang` Attribut gibt den Sprachcode an`en-US`, für den das Lexikon gilt. Amazon Polly kann dieses Beispiellexikon verwenden, wenn die Stimme, die Sie im `SynthesizeSpeech` Anruf angeben, denselben Sprachcode hat (en-US).

    > **Note**
    >
    > Sie können den `DescribeVoices`-Vorgang verwenden, um nach dem mit einer Stimme verknüpften Sprachcode zu suchen.

  - Das `alphabet`-Attribut gibt `IPA` an, das heißt, das international phonetische Alphabet (IPA) wird für die Aussprache verwendet. IPA ist eines der Alphabete zum Schreiben von Aussprachen. Amazon Polly unterstützt auch das X-SAMPA (Extended Speech Assessment Methods Phonetic Alphabet).

- Das `<lexeme>`-Element beschreibt die Zuordnung zwischen `<grapheme>` (d. h. Textdarstellung des Wortes) und `<alias>`.

Führen Sie zum Testen dieses Lexikons folgende Schritte aus:

1. Speichern Sie das Lexikon unter dem Namen `example.pls`.

2. Führen Sie den `put-lexicon` AWS CLI Befehl aus, um das Lexikon (mit dem Name`w3c`) in der Region us-east-2 zu speichern.

```
aws polly put-lexicon \
--name w3c \
--content file://example.pls
```

3. Führen Sie den `synthesize-speech`-Befehl aus, um den Beispieltext synthetisch in einem Audio-Stream (`speech.mp3`) zu bilden, und geben Sie den optionalen`lexicon-name`-Parameter an.

```
aws polly synthesize-speech \
--text 'W3C is a Consortium' \
--voice-id Joanna \
--output-format mp3 \
--lexicon-names="w3c" \
speech.mp3
```

4. Geben Sie die resultierende `speech.mp3`-Datei wieder und beachten Sie, dass das Wort W3C in dem Text durch World Wide Web Consortium ersetzt wird.

Im vorherigen Beispiel-Lexikon wird ein Alias verwendet. Das im Lexikon erwähnte IPA-Alphabet wird nicht verwendet. Das folgende Lexikon gibt eine phonetische Aussprache über das <phoneme>-Element mit dem IPA-Alphabet an.

```
<?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>
```

Führen Sie zum Testen dieses Lexikons die gleichen Schritte aus. Stellen Sie sicher, dass Sie einen Eingabetext angeben, der das Wort „Pekannuss“ enthält (z. B. „Pekannusskuchen ist köstlich“).

In den folgenden Ressourcen finden Sie zusätzliche Codebeispiele für den PutLexicon API-Vorgang:

- Java-Beispiele: PutLexicon
- Python (Boto3)-Beispiel: PutLexicon

AWS CLI - multiple lexemes

Mit Amazon Polly können Sie PutLexicon Aussprachelexika in einer bestimmten AWS Region für Ihr Konto speichern. Anschließend können Sie eine oder mehrere dieser gespeicherten Lexika in Ihrer SynthesizeSpeech-Anforderung angeben, die Sie anwenden möchten, bevor der Dienst mit der synthetischen Sprache beginnt. Weitere Informationen finden Sie unter Verwaltung von Lexika.

In diesem Beispiel wird das im Lexikon angegebene Lexem nur auf den synthetischen Eingabetext angewendet. Erwägen Sie das folgende Lexikon:

```
<?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>
```

Das Lexikon gibt drei Lexeme an, von denen zwei einen Alias für das Graphem W3C folgendermaßen definieren:

- Die erste <lexeme>-Element definiert einen Alias (World Wide Web Consortium).
- Das zweite <lexeme> definierte einen alternativen Alias (WWW Consortium).

Amazon Polly verwendet den ersten Ersatz für ein beliebiges Graphem in einem Lexikon.

Das dritte <lexeme> definiert eine Ersetzung (Community) für das Wort Consortium.

Testen wir zunächst dieses Lexikon. Nehmen wir an, Sie synthetisieren den folgenden Beispieltext in eine Audiodatei (speech.mp3) und geben das Lexikon in einem Aufruf an SynthesizeSpeech an.

```
The W3C is a Consortium
```

SynthesizeSpeech wendet das Lexikon zunächst folgendermaßen an:

- Wie bei dem ersten Lexem wird das Wort W3C in World Wide Web Consortium geändert. Der geänderte Text wird wie folgt angezeigt:

  ```
  The World Wide Web Consortium is a Consortium
  ```

- Der im dritten Lexem angegebene Alias wird nur auf das Wort Consortium angewendet, das Teil des ursprünglichen Texts war, sodass sich folgender Text ergibt:

  ```
  The World Wide Web Consortium is a Community.
  ```

Sie können dies wie folgt testen AWS CLI :

1. Speichern Sie das Lexikon unter dem Namen example.pls.
2. Führen Sie den put-lexicon-Befehl aus, um das Lexikon mit dem Namen w3c in der Region us-east-2 zu speichern.

   ```
   aws polly put-lexicon \
   ```

```
--name w3c \
--content file://example.pls
```

3. Führen Sie den `list-lexicons`-Befehl aus, um sicherzustellen, dass das w3c-Lexikon in der Liste der Lexika zurückgegeben wird.

```
aws polly list-lexicons
```

4. Führen Sie den `synthesize-speech`-Befehl aus, um den Beispieltext synthetisch in einer Audiodatei (`speech.mp3`) zu bilden, und geben Sie den optionalen `lexicon-name`-Parameter an.

```
aws polly synthesize-speech \
--text 'W3C is a Consortium' \
--voice-id Joanna \
--output-format mp3 \
--lexicon-names="w3c" \
speech.mp3
```

5. Geben Sie die `speech.mp3`-Datei wieder, um sicherzustellen, dass die synthetische Sprache die Textänderungen widerspiegelt.

In den folgenden Ressourcen finden Sie zusätzliche Codebeispiele für den PutLexicon API-Vorgang:

- Java-Beispiele: PutLexicon
- Python (Boto3)-Beispiel: PutLexicon

Lexika anwenden (Sprachsynthese)

Die von Ihnen verwendeten Lexika müssen der W3C-Empfehlung der Pronunciation Lexicon Specification (PLS) entsprechen. Weitere Informationen dazu finden Sie auf der W3C-Website unter Angaben zum Aussprachelexikon (PLS) Version 1.0.

Console

Im folgenden Verfahren wird gezeigt, wie Sie ein Lexikon auf Ihren Eingabetext anwenden können, indem Sie das `W3c.pls`-Lexikon so anwenden, dass es „World Wide Web Consortium“ durch „W3C“ ersetzt. Wenn Sie mehrere Lexika auf Ihren Text anwenden, Ihrem Text werden

sie in der Reihenfolge von oben nach unten angewendet, dabei hat die erste Übereinstimmung Vorrang vor nachfolgenden Übereinstimmungen. Ein Lexikon wird nur auf den Text angewendet, wenn die die im Lexikon angegebene Sprache mit der ausgewählten Sprache übereinstimmt.

Sie können ein Lexikon auf Klartext oder SSML-Eingaben anwenden.

Example– Anwenden des W3C.pls-Lexikons

Weitere Informationen zum Erstellen des Lexikons, das Sie für diese Übung benötigen, finden Sie unter Ein Lexikon hochladen. Verwenden Sie einen Texteditor zum Erstellen des W3C.pls Lexikon, das oben im Thema angezeigt wird. Merken Sie sich, wo Sie diese Datei speichern.

So wenden Sie das W3C.pls-Lexikon auf Ihre Eingabe an

In diesem Beispiel setzen wir Lexika ein, um „World Wide Web Consortium“ durch „W3C“ zu ersetzen. Vergleichen Sie das Ergebnis dieser Übung mit dem von SSML auf der Konsole verwenden für Englisch (USA) und eine weitere Sprache.

1. Melden Sie sich bei der an AWS-Managementkonsole und öffnen Sie die Amazon Polly Polly-Konsole unter https://console.aws.amazon.com/polly/.
2. Führen Sie eine der folgenden Aktionen aus:

   - Schalten Sie SSML aus und geben Sie dann diesen Text ein oder fügen Sie ihn in das Texteingabefeld ein.

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

   - Aktivieren Sie SSML und geben Sie dann diesen Text ein oder fügen Sie ihn in das Texteingabefeld ein.

     ```
     <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. Wählen Sie in der Liste Sprache die Option Englisch, USA und dann die Stimme aus, die Sie für diesen Text verwenden möchten.
4. Erweitern Sie Zusätzliche Einstellungen und aktivieren Sie „Aussprache anpassen“.
5. Wählen Sie aus der Liste der Lexika `W3C (English, US)` aus.

Wenn das `W3C (English, US)`-Lexikon nicht aufgeführt ist, wählen Sie Upload lexicon und laden Sie es hoch, anschließend können Sie es aus der Liste wählen. Informationen zum Erstellen dieses Lexikons finden Sie unter Ein Lexikon hochladen.

6. Um sich die Rede sofort anzuhören, wählen Sie „Zuhören“.
7. So speichern Sie die Sprachausgabe in einer Datei

   a. Wählen Sie Herunterladen aus.

   b. Um zu einem anderen Dateiformat zu wechseln, aktivieren Sie die Einstellungen für das Sprachdateiformat, wählen Sie das gewünschte Dateiformat aus und klicken Sie dann auf Herunterladen.

Wiederholen Sie den vorherigen Schritten, wählen Sie jedoch eine andere Sprache, und beachten Sie den Unterschied in der Ausgabe.

AWS CLI

Bei einem Aufruf an `SynthesizeSpeech` können Sie mehrere Lexika angeben. In diesem Fall überschreibt das erste angegeben Lexikon (von links nach rechts) alle vorausgehenden Lexika.

Erwägen Sie die folgenden zwei Lexika. Beachten Sie, dass jedes Lexikon verschiedene Aliase für dasselbe Graphem W3C beschreibt.

- Lexikon 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>
```

- Lexikon 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>
```

Nehmen wir an, Sie speichern diese Lexika als `w3c` und `w3cAlternate`. Wenn Sie Lexika in der Reihenfolge (`w3c` gefolgt von `w3cAlternate`) in einem `SynthesizeSpeech`-Aufruf angeben, hat der im ersten Lexikon für W3C angegebene Alias Vorrang vor dem im zweiten Lexikon angegebenen Alias. Führen Sie zum Testen der Lexika folgende Schritte aus:

1. Speichern Sie die Lexika lokal als `w3c.pls` und `w3cAlternate.pls`..
2. Laden Sie diese Lexika mit dem `put-lexicon` AWS CLI Befehl hoch.

   - Laden Sie das `w3c.pls`-Lexikon hoch und speichern Sie es als `w3c`.

     ```
     aws polly put-lexicon \
     --name w3c \
     --content file://w3c.pls
     ```

   - Laden Sie das `w3cAlternate.pls`-Lexikon als Dienst als `w3cAlternate` hoch.

     ```
     aws polly put-lexicon \
     --name w3cAlternate \
     --content file://w3cAlternate.pls
     ```

3. Führen Sie den `synthesize-speech`-Befehl aus, um den Beispieltext synthetisch in einem Audio-Stream (`speech.mp3`) zu bilden, und geben Sie beide Lexika mithilfe des `lexicon-name`-Parameters an.

   ```
   aws polly synthesize-speech \
   --text 'PLS is a W3C recommendation' \
   --voice-id Joanna \
   ```

```
--output-format mp3 \
--lexicon-names '["w3c","w3cAlternative"]' \
speech.mp3
```

4. Testen der `speech.mp3` Ergebnisse Sie sollte wie folgt gelesen werden:

```
PLS is a World Wide Web Consortium recommendation
```

Filterung der Lexikonliste auf der Konsole

Im folgenden Verfahren wird beschrieben, wie Sie die Lexikonliste filtern können, damit nur Lexika einer gewählten Sprache angezeigt werden.

Console

So filtern Sie nach Sprache aufgelistete Lexika

1. Melden Sie sich bei der an AWS-Managementkonsole und öffnen Sie die Amazon Polly Polly-Konsole unter https://console.aws.amazon.com/polly/.
2. Wählen Sie die Registerkarte Lexicons aus.
3. Wählen Sie eine beliebige Sprache.
4. Wählen Sie aus der Liste der Sprachen die Sprache, nach der Sie filtern möchten.

   Die Liste zeigt nur die Lexika für die gewählte Sprache.

AWS CLI

Amazon Polly bietet den ListLexicons API-Vorgang, mit dem Sie die Liste der Aussprachelexika in Ihrem Konto in einer bestimmten AWS Region abrufen können. Der folgende AWS CLI Aufruf listet die Lexika in Ihrem Konto in der Region US-East-2 auf.

```
aws polly list-lexicons
```

Es folgt ein Beispiel für eine Antwort mit zwei Lexika mit den Namen `w3c` und `tomato`. Für jedes Lexikon gibt die Antwort gibt Metadaten zurück, wie z. B. den Sprachcode, auf das das Lexikon angewendet wird, die Anzahl der im Lexikon definierten Lexeme, die Größe in Byte usw. Der

Sprachcode beschreibt eine Sprache und ein Gebietsschema, auf die die im Lexikon definierten Lexeme angewendet werden.

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

Die folgenden Ressourcen enthalten zusätzliche Informationen für den Vorgang: ListLexicons

- Java-Beispiele: ListLexicons
- Python (Boto3)-Beispiel: ListLexicon

Lexika werden auf die Konsole heruntergeladen

Im folgenden Verfahren wird beschrieben, wie Sie ein Lexikon oder mehrere Lexika herunterladen können. Sie können Lexikoneinträge in der Datei hinzufügen, entfernen oder ändern und sie dann erneut hochladen, um Ihr Lexikon beizubehalten. up-to-date

Console

So laden Sie ein Lexikon oder mehrere Lexika herunter

1. Melden Sie sich bei der an AWS-Managementkonsole und öffnen Sie die Amazon Polly Polly-Konsole unter https://console.aws.amazon.com/polly/.
2. Wählen Sie die Registerkarte Lexicons aus.
3. Wählen Sie das Lexikon oder die Lexika, die Sie herunterladen möchten.
   a. Wählen Sie den Namen aus der Liste, um ein einzelnes Lexikon herunterzuladen.
   b. Um mehrere Lexika als einzelne komprimierte Archivdatei herunterzuladen, aktivieren Sie das Kontrollkästchen neben den Einträgen in der Liste, die Sie herunterladen möchten.
4. Wählen Sie Herunterladen aus.
5. Öffnen Sie den Ordner, aus dem Sie das Lexikon herunterladen möchten.
6. Wählen Sie Speichern.

AWS CLI

Amazon Polly bietet die GetLexicon API-Operation zum Abrufen des Inhalts eines Aussprachelexikons, das Sie in Ihrem Konto in einer bestimmten Region gespeichert haben.

Der folgende `get-lexicon` AWS CLI Befehl ruft den Inhalt des Lexikons ab. `example`

```
aws polly get-lexicon \
--name example
```

Falls Sie noch kein Lexikon in Ihrem Konto gespeichert haben, können Sie den `PutLexicon`-Vorgang verwenden, um eines zu speichern. Weitere Informationen finden Sie unter Ein Lexikon hochladen.

Im Folgenden wird eine Beispielantwort dargestellt: Zusätzlich zum Lexikoninhalt gibt die Antwort den Code der Metadaten zurück, z. B. den Sprachcode, auf den das Lexikon angewendet wird, die Anzahl der im Lexikon definierten Lexeme, den Amazon Resource Name (ARN) der Ressource und die Größe des Lexikons in Byte. Der `LastModified`-Wert ist ein Unix-Zeitstempel.

```
{
```

```
    "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
    }
}
```

Die folgenden Ressourcen enthalten zusätzliche Codebeispiele für den GetLexicon Vorgang:

- Java-Beispiele: GetLexicon
- Python (Boto3)-Beispiel: GetLexicon

Löschen eines Lexikons

Im folgenden Verfahren wird beschrieben, wie Sie ein Lexikon löschen können. Nach dem Löschen des Lexikons müssen Sie es wieder hinzufügen, bevor Sie es erneut verwenden können. Sie können ein Lexikon oder mehrere Lexika gleichzeitig löschen, indem Sie die Kontrollkästchen neben den einzelnen Lexika aktivieren.

Console

So löschen Sie ein Lexikon

1. Melden Sie sich bei der an AWS-Managementkonsole und öffnen Sie die Amazon Polly Polly-Konsole unter https://console.aws.amazon.com/polly/.
2. Wählen Sie die Registerkarte Lexicons aus.
3. Wählen Sie ein Lexikon oder mehrere zu löschende Lexika aus der Liste, die Sie löschen möchten.
4. Wählen Sie Löschen aus.
5. Geben Sie den Bestätigungstext ein und wählen Sie dann Löschen, um das Lexikon aus der Region zu entfernen, oder Abbrechen, um es beizubehalten.

AWS CLI

Amazon Polly bietet die DeleteLexicon API-Operation zum Löschen eines Aussprachelexikons aus einer bestimmten AWS Region in Ihrem Konto. Im Folgenden wird das AWS CLI angegebene Lexikon gelöscht.

Das folgende AWS CLI Beispiel ist für Unix, Linux und macOS formatiert. Ersetzen Sie unter Windows den Unix-Fortsetzungszeichen mit umgekehrtem Schrägstrich (\) am Ende jeder Zeile durch ein Caret (^) und setzen Sie den Eingabetext in vollständige Anführungszeichen („) und einfache Anführungszeichen (') für interne Tags.

```
aws polly delete-lexicon \
--name example
```

Die folgenden Ressourcen enthalten zusätzliche Informationen für den Vorgang: DeleteLexicon

- Java-Beispiele: DeleteLexicon
- Python (Boto3)-Beispiel: DeleteLexicon

Lange Audiodateien

Verwenden Sie die asynchrone Synthesefunktion von Amazon Polly, um TTS-Dateien für große Textpassagen zu erstellen. Dabei werden die drei verwendet: `SpeechSynthesisTask` APIs

- `StartSpeechSynthesisTask`: beginnt eine neue Syntheseaufgabe.
- `GetSpeechSynthesisTask`: gibt Details zurück zu einer zuvor übermittelten Syntheseaufgabe.
- `ListSpeechSynthesisTasks`: listet alle übermittelten Syntheseaufgaben auf.

Die `SynthesizeSpeech`-Operation erzeugt nahezu in Echtzeit Audiomaterial mit meist relativ geringer Latenz. Aus diesem Grund kann die Operation nur 3000 Zeichen generieren.

Die asynchrone Synthese-Funktion von Amazon Polly bewältigt die Herausforderung, ein größeres Textdokument zu verarbeiten, indem sie die Art und Weise ändert, wie das Dokument sowohl synthetisiert als auch zurückgegeben wird. Wenn eine Syntheseanfrage gestellt wird, indem der Eingabetext mit dem gesendet wird`StartSpeechSynthesisTask`, stellt Amazon Polly die Anfragen in eine Warteschlange und verarbeitet sie dann asynchron im Hintergrund, sobald die Systemressourcen verfügbar sind. Amazon Polly lädt dann den resultierenden Sprach- oder Sprachmarken-Stream direkt in Ihren (erforderlichen) Amazon Simple Storage Service (Amazon S3) -Bucket hoch und benachrichtigt Sie über Ihr (optionales) SNS-Thema über die Verfügbarkeit der fertigen Datei.

Auf diese Weise steht die gesamte Funktionalität mit Ausnahme der Verarbeitung in Echtzeit für Texte von bis zu 100 000 kostenpflichtigen Zeichen (bzw. 200 000 Zeichen insgesamt) zur Verfügung.

Um ein Dokument mit dieser Methode zu synthetisieren, benötigen Sie einen beschreibbaren Amazon S3 S3-Bucket, in dem die Audiodatei gespeichert werden kann. Sie können benachrichtigt werden, wenn das generierte Audiomaterial bereit ist, indem Sie eine optionale SNS-Themenkennung angeben. Wenn die Syntheseaufgabe abgeschlossen ist, veröffentlicht Amazon Polly eine Nachricht zu diesem Thema. Diese Meldung kann auch nützliche Fehlerinformationen enthalten, wenn die Syntheseaufgabe nicht erfolgreich war. Stellen Sie zu diesem Zweck sicher, dass der Benutzer, der die Syntheseaufgabe erstellt, auch im SNS-Thema veröffentlichen kann. Weitere Informationen zum Erstellen und Abonnieren eines SNS-Themas finden Sie in der Amazon SNS-Dokumentation.

Verschlüsselung

Sie können die Ausgabedatei in verschlüsselter Form in Ihrem S3-Bucket speichern. Dazu aktivieren Sie die Verschlüsselung des Amazon S3-Buckets, die eine der stärksten Blockverschlüsselungen verwendet, die zur Verfügung stehen, nämlich 256-Bit Advanced Encryption Standard (AES-256).

Themen

Einrichtung der IAM-Richtlinie für asynchrone Synthese

Damit Sie die Funktionalität der asynchronen Synthese nutzen können, benötigen Sie eine IAM-Richtlinie, die Folgendes zulässt:

- Nutzung neuer Amazon Polly Polly-Operationen
- Schreiben in den ausgegebenen S3-Bucket
- Veröffentlichung im Status des SNS-Themas [optional]

Mit der folgenden Richtlinie werden nur die für die asynchrone Synthese erforderlichen Berechtigungen erteilt. Die Richtlinie kann mit dem IAM-Benutzer verknüpft werden.

Lange Audiodateien erstellen

Sie können die Amazon Polly Polly-Konsole verwenden, um lange Reden mithilfe der asynchronen Synthese mit derselben Funktionalität wie mit der zu erstellen. AWS CLI Dies erfolgt wie jede andere Synthese über die Registerkarte Text-to-Speech.

Console

Die weiteren Funktionen der asynchronen Synthese sind ebenfalls über die Konsole verfügbar. Die Registerkarte S3 synthesis tasks (S3-Syntheseaufgabe) zeigt die `ListSpeechSynthesisTasks`-Funktionalität, die alle im S3-Bucket gespeicherten Aufgaben anzeigt und es Ihnen ermöglicht, diese wenn nötig zu filtern. Durch Klicken auf eine bestimmte einzelne Aufgabe werden Details angezeigt, die die `GetSpeechSynthesisTask`-Funktionalität abbilden.

Um einen großen Text mit der Amazon Polly Polly-Konsole zu synthetisieren

1. Melden Sie sich bei der an AWS-Managementkonsole und öffnen Sie die Amazon Polly Polly-Konsole unter https://console.aws.amazon.com/polly/.
2. Wählen Sie die Registerkarte Text-to-Speech. Wählen Sie gegebenenfalls Long Form als Engine aus.
3. Wenn SSML aktiviert oder deaktiviert ist, geben Sie Ihren Text ein oder fügen Sie ihn in das Eingabefeld ein.
4. Wählen Sie Sprache, Region und Stimme für Ihren Text.
5. Wählen Sie In S3 speichern.

   > **Note**
   >
   > Sowohl die Optionen „Herunterladen“ als auch „Abhören“ sind ausgegraut, wenn die Textlänge für den `SynthesizeSpeech` Echtzeitvorgang die Obergrenze von 3.000 Zeichen überschreitet.

6. Die Konsole öffnet ein Formular, sodass Sie auswählen können, wo die Ausgabedatei gespeichert werden soll.

   a. Geben Sie den Namen des Amazon S3-Ziel-Buckets ein.

   b. Geben Sie optional den Präfixschlüssel der Ausgabe ein.

      > **Note**
      >
      > Der ausgegebene S3-Bucket muss beschreibbar sein.

   c. Wenn Sie benachrichtigt werden möchten, wenn die Syntheseaufgabe abgeschlossen ist, geben Sie optional eine SNS-Themen-ID an.

      > **Note**
      >
      > Der SNS muss für die Veröffentlichung durch den aktuellen Konsolenbenutzer geöffnet sein, um diese Option nutzen zu können. Weitere Informationen finden Sie unter Amazon Simple Notification Service (SNS).

   d. Wählen Sie In S3 speichern.

So rufen Sie Informationen zu Ihren Sprachsyntheseaufgaben ab

1. Wählen Sie in der Konsole die Registerkarte S3 Synthesis Tasks (S3-Syntheseaufgaben) aus.
2. Die Aufgaben werden nach Datum sortiert angezeigt. Um die Aufgaben nach Status zu filtern, wählen Sie Alle Status und dann den Status aus, den Sie verwenden möchten.
3. Um die Details einer bestimmten Aufgabe anzuzeigen, wählen Sie die verknüpfte Task ID (Aufgaben-ID).

AWS CLI

Die asynchrone Synthese-Funktionalität von Amazon Polly verwendet drei Funktionen `SpeechSynthesisTask` APIs , um mit großen Textmengen zu arbeiten:

- `StartSpeechSynthesisTask`: beginnt eine neue Syntheseaufgabe.
- `GetSpeechSynthesisTask`: gibt Details zurück zu einer zuvor übermittelten Syntheseaufgabe.
- `ListSpeechSynthesisTasks`: listet alle übermittelten Syntheseaufgaben auf.

Generieren von großen Textmengen (**`StartSpeechSynthesisTask`**)

Wenn Sie eine Audiodatei erstellen möchten, die größer ist als eine, die Sie mit der Echtzeitfunktion `SynthesizeSpeech` erstellen können, verwenden Sie die `StartSpeechSynthesisTask`-Operation. Zusätzlich zu den für den `SynthesizeSpeech` Vorgang benötigten Argumenten ist `StartSpeechSynthesisTask` auch der Name eines Amazon S3 S3-Buckets erforderlich. Zwei weitere optionale Argumente sind ebenfalls verfügbar: ein Schlüsselpräfix für die Ausgabedatei und der ARN für ein SNS-Thema, wenn Sie eine Statusbenachrichtigung über die Aufgabe erhalten möchten.

- `OutputS3BucketName`: Der Name des Amazon S3 S3-Buckets, in den die Synthese hochgeladen werden soll. Dieser Bucket sollte sich in derselben Region befinden wie der Amazon Polly Polly-Service. Darüber hinaus sollte der IAM-Benutzer, der für den Anruf verwendet wird, Zugriff auf den Bucket haben. [Erforderlich]
- `OutputS3KeyPrefix`: Schlüsselpräfix für die Ausgabedatei. Verwenden Sie diesen Parameter, wenn Sie die Sprachausgabedatei in einem benutzerdefinierten, verzeichnisähnlichen Schlüssel in Ihrem Bucket speichern möchten. [Optional]

- `SnsTopicArn`: Der ARN für das SNS-Thema, den Sie verwenden können, wenn Sie Benachrichtigungen über den Status der Aufgabe erhalten möchten. Dieses SNS-Thema sollte sich in derselben Region wie der Amazon Polly befinden. Darüber hinaus sollte der IAM-Benutzer, der für den Anruf verwendet wird, Zugriff auf das Thema haben. [Optional]

Das folgende Beispiel kann beispielsweise verwendet werden, um den `start-speech-synthesis-task` AWS CLI Befehl in der Region USA Ost (Ohio) auszuführen:

Das folgende AWS CLI Beispiel ist für Unix, Linux und macOS formatiert. Ersetzen Sie unter Windows den Unix-Fortsetzungszeichen mit umgekehrtem Schrägstrich (\) am Ende jeder Zeile durch ein Caret (^) und setzen Sie den Eingabetext in vollständige Anführungszeichen („) und einfache Anführungszeichen (') für interne Tags.

```
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
```

Dies führt zu einer Antwort, die ähnlich aussieht wie diese:

```
"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"
 }
```

Die `start-speech-synthesis-task`-Operation gibt mehrere neue Felder zurück:

- `OutputUri`: der Speicherort Ihrer Ausgabesprachdatei.

- `TaskId`: eine eindeutige Kennung für die von Amazon Polly generierte Sprachsyntheseaufgabe.
- `CreationTime`: Zeitstempel für den Zeitpunkt, zu dem die Aufgabe ursprünglich übermittelt wurde.
- `RequestCharacters`: die Anzahl kostenpflichtiger Zeichen in der Aufgabe.
- `TaskStatus`: gibt Auskunft über den Status der übermittelten Aufgabe.

  Sobald Ihre Aufgabe übermittelt wurde, zeigt der ursprüngliche Status `scheduled` an. Wenn Amazon Polly mit der Bearbeitung der Aufgabe beginnt, ändert sich der Status zu `inProgress` und später zu `completed` oder`failed`. Schlägt die Aufgabe fehl, wird beim Aufrufen der ListSpeechSynthesisTasks Operation GetSpeechSynthesisTask oder eine Fehlermeldung zurückgegeben.

Wenn die Aufgabe abgeschlossen ist, wird die Sprachdatei am angegebenen Speicherort in `OutputUri` verfügbar.

Informationen zu Ihren Sprachsyntheseaufgaben abrufen

Informationen zu einer Aufgabe, wie z. B. Fehler, Status usw., erhalten Sie mithilfe der Verwendung der `GetSpeechSynthesisTask`-Operation. Zu diesem Zweck benötigen Sie die `task-id`, die von der `StartSpeechSynthesisTask` zurückgegeben wird.

Das folgende Beispiel kann beispielsweise verwendet werden, um den `get-speech-synthesis-task` AWS CLI Befehl auszuführen:

```
aws polly get-speech-synthesis-task \
--region us-east-2 \
--endpoint-url "https:// polly.us-east-2.amazonaws.com/" \
--task-id task identifier
```

Sie können auch alle Sprachsynthese-Aufgaben auflisten, die Sie in der aktuellen Region ausgeführt haben. Verwenden Sie dazu die `ListSpeechSynthesisTasks`-Operation.

Das folgende Beispiel kann beispielsweise verwendet werden, um den `list-speech-synthesis-tasks` AWS CLI Befehl auszuführen:

```
aws polly list-speech-synthesis-tasks \
--region us-east-2 \
```

```
--endpoint-url "https:// polly.us-east-2.amazonaws.com/"
```

Kontingente in Amazon Polly

Amazon Polly wendet Kontingente auf den Kundenverkehr an, indem es übermäßige Anfragen ablehnt. Das Standardkontingent für `SynthesizeSpeech` Anfragen mit Standardstimmen beträgt 80 Transaktionen pro Sekunde (tps) in einer einzigen Region für eine einzelne. AWS-Konto Wenn die Grenzwerte nicht erhöht würden und Sie 100 `SynthesizeSpeech` Anfragen pro Sekunde mit einer Standardsprache generieren würden, wären 80 Anfragen pro Sekunde erfolgreich, und 20 Anfragen pro Sekunde würden von Amazon Polly gedrosselt. Diese Anfragen würden eine Antwort mit dem HTTP-Status 400 und einem entsprechenden Antwort-Header zurückgeben. `ThrottlingException` Amazon Polly drosselt außerdem den Datenverkehr für alle Operationen auf der Grundlage der Anforderungsrate.

Beispiele für Einschränkungen bei der Sprachsynthese

- Synthetisieren Sie die ersten 24 Buchstaben des englischen Alphabets Buchstabe für Buchstabe. Wenn die Synthese jedes Buchstabens weniger als 50 Millisekunden dauern würde, würde bei einem Operationslimit von acht Tps die Synthese von 24 Buchstaben mindestens drei Sekunden dauern. Während dieser Zeit könnten Sie bis zu acht Buchstaben pro Sekunde synthetisieren. Alle weiteren Anfragen würden gedrosselt. Da die Anfragen nur eine kurze Zeit dauern, würden sie ohne Überschneidung seriell synthetisiert.
- Synthetisieren Sie 16 Textabsätze. Wenn jeder Absatz innerhalb von zwei Sekunden oder weniger synthetisiert und auf Kundenseite vollständig empfangen würde, würde die Zusammenfassung aller 16 Artikel bei einem Bearbeitungslimit von acht gleichzeitigen Anfragen mindestens vier Sekunden dauern. In der ersten Sekunde könnten Sie bis zu acht Anfragen starten. Bei gleichzeitigen Anfragen würde jeder Versuch, eine neue Synthese zu starten, aufgrund der Parallelitätsbegrenzung gedrosselt. Sie könnten die verbleibenden acht Absätze nach den ersten zwei Sekunden zusammenfassen, nachdem der erste Stapel von Anfragen abgeschlossen ist.

Beachten Sie bei der Verwendung von Amazon Polly die folgenden Beschränkungen.

Themen

- SpeechSynthesisTask API-Operationen
- Speech Synthesis Markup Language (SSML)

Unterstützte -Regionen

Eine Liste der AWS Regionen, in denen Amazon Polly verfügbar ist, finden Sie unter Amazon Polly Endpoints and Quotas in der. Allgemeine Amazon Web Services-Referenz

- Informationen zu Regionen, die generative Stimmen unterstützen, finden Sie unter Generative Stimmen.
- Informationen zu Regionen, die Stimmen in langer Form unterstützen, finden Sie unter Long-form Stimmen.
- Informationen zu Regionen, die neuronale Stimmen unterstützen, finden Sie unter the section called “Kompatibilität mit Funktionen und Regionen” neuronale TTS.

Kontingente und Drosselungsraten

In der folgenden Tabelle sind die Drosselungsraten pro Amazon Polly Polly-Vorgang definiert. Sie können den verwenden AWS-Managementkonsole , um bei Bedarf Kontingenterhöhungen für die einstellbaren Kontingente zu beantragen.

Operation	Limit
Lexikon	

`DeleteLexicon`
`PutLexicon`
`GetLexicon`
`ListLexicons`
	Beliebige 2 Transaktionen pro Sekunde (TPS) kombiniert von diesen Operationen.
	Maximal zulässige Steigerung von 4 TPS.

Sprache	
`DescribeVoices`	80 TPS, bei einem Steigerungsmaximum von 100 TPS
`SynthesizeSpeech`	Generative Stimme: 8 Tips

Operation	Limit
	Long-form Stimme: 8 Tps mit einem Burst-Limit von 10 Tps Neuronale Stimme: 8 TPS mit einem Burst-Limit von 10 TPS Standardstimme: 80 TPS mit einem Burst-Limit von 100 TPS
`StartSpeechSynthesisTask`	Generative Stimme: 1 Tps Long-form Stimme: 1 Tps Neuronale Stimme: 10 tps Standardstimme: 10 TPS mit einem Burst-Limit von 12 TPS
`StartSpeechSynthesisStream`	Generative Stimme: 8 tps
`GetSynthesizeSpeechTask` und `ListSynthesizeSpeechTask`	Zulässiges Maximum: 10 TPS insgesamt

Gleichzeitige Anforderungen

Für generatives Sprechen unterstützt Amazon Polly bis zu 26 gleichzeitige Anfragen. Für Sprachanrufe in Langform unterstützt Amazon Polly bis zu 26 gleichzeitige Anfragen. Für neuronale Sprache unterstützt Amazon Polly 8 Tps mit einem Burst-Limit von 10 Tps für bis zu 18 gleichzeitige Anfragen. Amazon Polly unterstützt auch Limits für gleichzeitige Anfragen. Für Standardsprache unterstützt Amazon Polly 80 Tps für bis zu 80 gleichzeitige Anfragen.

Denn StartSpeechSynthesisStreamAmazon Polly unterstützt bis zu 8 gleichzeitige Anfragen.

Bewährte Methoden zur Minderung der Drosselung

- Versuchen Sie, Drosselungen mit Backoff und Jitter erneut durchzuführen, damit Sie die Last über einen kurzen Zeitraum verteilen und unerwartete Auslastungsspitzen bewältigen können, ohne die Verfügbarkeit zu beeinträchtigen. AWS-Codebeispiel-Katalog ist in vielen Programmiersprachen bereits standardmäßig dafür konfiguriert. Einzelheiten finden Sie unter Verhalten bei Wiederholungsversuchen von Funktionen.

- Verwenden Sie Amazon Polly Polly-Metriken. Amazon Polly veröffentlicht automatisch, um Ihre aktuelle Nutzung CloudWatch zu analysieren und das Nutzungswachstum zu prognostizieren.

> **Note**
>
> Bevor Sie eine Erhöhung des Kontingents beantragen (falls zutreffend), berechnen Sie Ihren TPS-Bedarf anhand der Richtlinien auf dieser Seite. Amazon Polly sichert nur die benötigten Rechenressourcen entsprechend der Kundennachfrage, um Ihre Kosten niedrig zu halten.

Lexika für die Aussprache

- Pro Konto können maximal 100 Lexika gespeichert werden.
- Als Lexikonname dürfen alphanumerische Zeichenfolgen mit maximal 20 Zeichen verwendet werden.
- Jedes Lexikon kann bis zu 40.000 Zeichen groß sein. (Beachten Sie, dass die Größe des Lexikons die Latenz des SynthesizeSpeech Vorgangs beeinflusst.)
- Sie dürfen je <phoneme>- oder <alias>-Ersatz in einem Lexikon maximal 100 Zeichen angeben.

Weitere Informationen zur Verwendung von Lexika finden Sie unter Verwaltung von Lexika.

SynthesizeSpeech API-Operationen

Beachten Sie bei der Schätzung der Nutzung von`SynthesizeSpeech`, dass die Wiedergabe des von Amazon Polly produzierten Audios, insbesondere für interaktive Anwendungen, in der Regel mindestens einige Sekunden dauert. Dadurch wird die Anzahl der Anfragen reduziert`SynthesizeSpeech`, selbst bei einer großen Anzahl gleichzeitiger Nutzer. Darüber hinaus drosselt Amazon Polly `SynthesizeSpeech` Anfragen nach der Anzahl der gleichzeitigen Anfragen, die es synthetisiert. Es gibt keine separate Einstellung für gleichzeitige Anfragen. Das Limit für gleichzeitige Anfragen entspricht immer der Anzahl der zulässigen TPs und wird entsprechend skaliert.

Beispielanwendung mit Kurzgeschichte. Sie können Amazon Polly verwenden, um eine Anwendung zu erstellen, die eine Reihe von Kurzgeschichten abspielt. Bei dieser Art von App würde die erste Geschichte abgespielt werden, dann die nächste usw., bis ein Benutzer die Anwendung beendet. Die Synthese jeder Geschichte würde etwa 0,5 Sekunden und das Abspielen 10 Sekunden dauern. In

diesem Szenario könnten Sie damit rechnen, dass alle 10 Sekunden, `SynthesizeSpeech` die der Kunde mit der Anwendung verbracht hat, ein Anruf eingeht. Dies würde einem Anruf pro Sekunde für jeweils 10 Kunden entsprechen, die die Anwendung gleichzeitig verwenden. Wenn Sie 1000 Kunden hätten, die die Anwendung gleichzeitig nutzen, könnten Sie mit einer durchschnittlichen Anrufrate `SynthesizeSpeech` von nur 100 Transaktionen pro Sekunde rechnen.

Bei der Verwendung der API-Operation `SynthesizeSpeech` gelten folgende Einschränkungen:

- Der Eingabetext darf maximal 3 500 berechnete Zeichen enthalten (6 000 Zeichen insgesamt). SSML-Tags werden nicht als berechnete Zeichen gezählt.
- Sie dürfen maximal fünf Lexika angeben, die auf den Eingabetext angewendet werden sollen.
- Der Ausgabe-Audiostream (Synthese) ist auf 10 Minuten beschränkt. Danach wird jeglicher verbleibende Sprechtext abgeschnitten.

Weitere Informationen finden Sie unter SynthesizeSpeech.

> **Note**
>
> Einige Einschränkungen der API-Operation `SynthesizeSpeech` können mit der API-Operation `StartSythensizeSpeechTask` umgangen werden. Weitere Informationen finden Sie unter Lange Audiodateien.

SpeechSynthesisTask API-Operationen

Folgende Einschränkungen gelten bei der Verwendung der API-Operationen `StartSpeechSynthesisTask`, `GetSpeechSynthesisTask` und `ListSpeechSynthesisTasks`:

- Der Eingabetext darf maximal 100 000 kostenpflichtige Zeichen enthalten (200 000 Zeichen insgesamt). SSML-Tags werden nicht als berechnete Zeichen gezählt.
- Sie dürfen maximal fünf Lexika angeben, die auf den Eingabetext angewendet werden sollen.

Speech Synthesis Markup Language (SSML)

Bei der Verwendung von SSML sind folgende Einschränkungen zu beachten:

- Die Tags `<audio>`, `<lexicon>`, `<lookup>` und `<voice>` werden nicht unterstützt.
- Elemente des Typs `<break>` dürfen jeweils maximal 10 Sekunden angeben.
- Das Tag `<prosody>` unterstützt für das Attribut "rate" keine niedrigeren Werte als –80 %.

Weitere Informationen finden Sie unter Sprache aus SSML-Dokumenten generieren.

Beispielcode und Anwendungen für Amazon Polly

Dieser Abschnitt enthält Codebeispiele und Beispielanwendungen, mit denen Sie Amazon Polly erkunden können.

Das Thema Beispielcode enthält Codefragmente, die nach Programmiersprachen geordnet und in Beispiele für verschiedene Funktionen von Amazon Polly unterteilt sind. Das Thema Beispielanwendung enthält nach Programmiersprachen geordnete Anwendungen, die unabhängig voneinander verwendet werden können, um Amazon Polly zu erkunden.

Wir empfehlen Ihnen, vor der Verwendung dieser Beispiele zunächst den Abschnitt So funktioniert Amazon Polly zu lesen und die in Erste Schritte mit Amazon Polly beschriebenen Schritte durchzuführen.

Themen

- Java-Beispiele
- Python-Beispiele
- Java-Beispiel
- Python-Beispiel (HTML5 Client und Python-Server)
- iOS-Beispiel
- Android-Beispiel

Java-Beispiele

Die folgenden Codebeispiele zeigen, wie Sie Java-basierte Anwendungen verwenden, um verschiedene Aufgaben mit Amazon Polly zu erledigen. Bei diesen Beispielen handelt es sich nicht um vollständige Beispiele, sie können jedoch in größere Java-Anwendungen integriert werden, die den verwenden. AWS SDK für Java

Codeausschnitte

- DeleteLexicon
- DescribeVoices
- GetLexicon
- ListLexicons
- PutLexicon

- StartSpeechSynthesisTask
- Sprachmarkierungen
- SynthesizeSpeech

DeleteLexicon

Das folgende Java-Codebeispiel zeigt, wie Java-basierte Anwendungen verwendet werden, um ein bestimmtes Lexikon zu löschen, das in einer Region gespeichert ist. AWS Ein gelöschtes Lexikon ist nicht für die Sprachsynthese verfügbar und kann auch nicht mit dem Oder abgerufen werden. `GetLexicon` `ListLexicon` APIs

Weitere Informationen zu diesem Vorgang finden Sie in der Referenz für die API `DeleteLexicon`.

SDK v2

```
/*
   Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
   SPDX-License-Identifier: Apache-2.0
*/

package com.example.polly;

import software.amazon.awssdk.auth.credentials.ProfileCredentialsProvider;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.polly.PollyClient;
import software.amazon.awssdk.services.polly.model.DeleteLexiconRequest;
import software.amazon.awssdk.services.polly.model.DeleteLexiconResponse;
import software.amazon.awssdk.services.polly.model.PollyException ;

/**
 * Before running this Java V2 code example, set up your development environment,
 including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class DeleteLexiconSample {

    public static void main(String args[]) {
```

```
        PollyClient polly = PollyClient.builder()
                .region(Region.US_WEST_2)
                .credentialsProvider(ProfileCredentialsProvider.create())
                .build();

        deleteLexicon(polly) ;
        polly.close();
    }

    private static String LEXICON_NAME = "SampleLexicon";
    public static void deleteLexicon(PollyClient client) {

        try {
            DeleteLexiconRequest deleteLexiconRequest = DeleteLexiconRequest.builder()
                    .name(LEXICON_NAME).build();

            DeleteLexiconResponse deleteLexiconResult =
 client.deleteLexicon(deleteLexiconRequest);

        } catch (PollyException e) {
            System.err.println("Exception caught: " + e);
            System.exit(1);
        }
    }

}
```

SDK Version 1

```
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

Das folgende Java-Codebeispiel zeigt, wie mithilfe von Java-basierten Anwendungen eine Liste der Stimmen erstellt wird, die für die Anforderung der Sprachsynthese verfügbar sind. Sie können optional einen Sprachcode angeben, um die verfügbaren Stimmen zu filtern. Wenn Sie beispielsweise en-US angeben, gibt der Vorgang eine Liste aller verfügbaren US-englischen Stimmen zurück.

Weitere Informationen zu diesem Vorgang finden Sie in der Referenz für die API `DescribeVoices`.

```
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

Das folgende Java-Codebeispiel zeigt, wie Java-basierte Anwendungen verwendet werden, um den Inhalt eines bestimmten Aussprachelexikons zu erzeugen, das in einer Region gespeichert ist. AWS

Weitere Informationen zu diesem Vorgang finden Sie in der Referenz für die API GetLexicon.

SDK v2

```
/*
   Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
   SPDX-License-Identifier: Apache-2.0
*/

package com.example.polly;

import software.amazon.awssdk.auth.credentials.ProfileCredentialsProvider;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.polly.PollyClient;
import software.amazon.awssdk.services.polly.model.GetLexiconRequest;
import software.amazon.awssdk.services.polly.model.GetLexiconResponse;
import software.amazon.awssdk.services.polly.model.PollyException ;

/**
 * Before running this Java V2 code example, set up your development environment,
 including your credentials.
```

```
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class GetLexiconSample {

    public static void main(String args[]) {

        PollyClient polly = PollyClient.builder()
                .region(Region.US_WEST_2)
                .credentialsProvider(ProfileCredentialsProvider.create())
                .build();

        getLexicon(polly) ;
        polly.close();
    }

    private static String LEXICON_NAME = "SampleLexicon";
    public static void getLexicon(PollyClient client) {

        try {
            GetLexiconRequest getLexiconRequest = GetLexiconRequest.builder()
                    .name(LEXICON_NAME).build();

            GetLexiconResponse getLexiconResult = client.getLexicon(getLexiconRequest);
            System.out.println("The name of the Lexicon is " +
 getLexiconResult.lexicon().name());

        } catch (PollyException e) {
            System.err.println("Exception caught: " + e);
            System.exit(1);
        }
    }

}
```

SDK Version 1

```
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

Das folgende Java-Codebeispiel zeigt, wie Java-basierte Anwendungen verwendet werden, um eine Liste von Aussprachelexika zu erstellen, die in einer Region gespeichert sind. AWS

Weitere Informationen zu diesem Vorgang finden Sie in der Referenz für die API `ListLexicons`.

```
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

Das folgende Java-Codebeispiel zeigt, wie Java-basierte Anwendungen verwendet werden, um ein Aussprachelexikon in einer Region zu speichern. AWS

Weitere Informationen zu diesem Vorgang finden Sie in der Referenz für die API [PutLexicon](#).

SDK v2

```
/*
   Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
   SPDX-License-Identifier: Apache-2.0
*/

package com.example.polly;

import software.amazon.awssdk.auth.credentials.ProfileCredentialsProvider;
```

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.polly.PollyClient;
import software.amazon.awssdk.services.polly.model.PutLexiconRequest;
import software.amazon.awssdk.services.polly.model.PutLexiconResponse;
import software.amazon.awssdk.services.polly.model.PollyException ;

/**
 * Before running this Java V2 code example, set up your development environment,
 including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class PutLexiconSample {

    public static void main(String args[]) {

        PollyClient polly = PollyClient.builder()
                .region(Region.US_WEST_2)
                .credentialsProvider(ProfileCredentialsProvider.create())
                .build();

        putLexicon(polly) ;
        polly.close();
    }

    private static 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 static String LEXICON_NAME = "SampleLexicon";
    public static void putLexicon(PollyClient client) {

        try {
            PutLexiconRequest putLexiconRequest = PutLexiconRequest.builder()
                    .name(LEXICON_NAME).content(LEXICON_CONTENT).build();

            PutLexiconResponse putLexiconResult = client.putLexicon(putLexiconRequest);
```

```
        } catch (PollyException e) {
            System.err.println("Exception caught: " + e);
            System.exit(1);
        }
    }

}
```

SDK Version 1

```
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

Das folgende Java-Codebeispiel zeigt, wie Sie mithilfe von Java-basierten Anwendungen eine lange Sprache (bis zu 100.000 fakturierte Zeichen) synthetisieren und direkt in einem Amazon S3 S3-Bucket speichern können.

Weitere Informationen finden Sie in der Referenz für API `StartSpeechSynthesisTask`.

SDK v2

```
/*
   Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
   SPDX-License-Identifier: Apache-2.0
*/

package com.example.polly;

import software.amazon.awssdk.auth.credentials.ProfileCredentialsProvider;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.polly.PollyClient;
import software.amazon.awssdk.services.polly.model.*;

import java.time.Duration;
import org.awaitility.Durations;
import org.awaitility.Awaitility;

/**
 * Before running this Java V2 code example, set up your development environment,
 including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class StartSpeechSynthesisTaskSample {

    public static void main(String args[]) {

        PollyClient polly = PollyClient.builder()
                .region(Region.US_WEST_2)
                .credentialsProvider(ProfileCredentialsProvider.create())
                .build();
```

```
        startSpeechSynthesisTask(polly) ;
        polly.close();
    }

    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 =
 Durations.FIVE_SECONDS;
    private static final Duration SYNTHESIS_TASK_POLL_DELAY = Durations.TEN_SECONDS;
    private static final Duration SYNTHESIS_TASK_TIMEOUT = Durations.FIVE_MINUTES;
    public static void startSpeechSynthesisTask(PollyClient client) {

        try {
            StartSpeechSynthesisTaskRequest startSpeechSynthesisTaskRequest =
 StartSpeechSynthesisTaskRequest.builder()

 .outputFormat(OUTPUT_FORMAT_MP3).text(PLAIN_TEXT).textType(TextType.TEXT)

 .voiceId(VoiceId.AMY).outputS3BucketName(OUTPUT_BUCKET).snsTopicArn(SNS_TOPIC_ARN)
                    .engine("neural").build();

            StartSpeechSynthesisTaskResponse startSpeechSynthesisTaskResponse =
                    client.startSpeechSynthesisTask(startSpeechSynthesisTaskRequest);
            String taskId = startSpeechSynthesisTaskResponse.synthesisTask().taskId();

            Awaitility.await().with()
                    .pollInterval(SYNTHESIS_TASK_POLL_INTERVAL)
                    .pollDelay(SYNTHESIS_TASK_POLL_DELAY)
                    .atMost(SYNTHESIS_TASK_TIMEOUT)
                    .until(
                            () -> getSynthesisTaskStatus(client,
 taskId).equals(TaskStatus.COMPLETED.toString())
                    );

        } catch (PollyException e) {
            System.err.println("Exception caught: " + e);
            System.exit(1);
        }
    }
```

```
    private static String getSynthesisTaskStatus(PollyClient client, String taskId) {
        GetSpeechSynthesisTaskRequest getSpeechSynthesisTaskRequest =
 GetSpeechSynthesisTaskRequest.builder()
                .taskId(taskId).build();
        GetSpeechSynthesisTaskResponse result =
 client.getSpeechSynthesisTask(getSpeechSynthesisTaskRequest);
        return result.synthesisTask().taskStatusAsString();
    }

}
```

SDK Version 1

```
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();
    }

}
```

Sprachmarkierungen

Das folgende Codebeispiel zeigt, wie Java-basierte Anwendungen verwendet werden, um Sprachmarkierungen für eingegebenen Text zu synthetisieren. Diese Funktionalität verwendet die SynthesizeSpeech API.

Weitere Informationen zur Funktionalität finden Sie unter Sprachzeichen.

Weitere Informationen zur API finden Sie in der Referenz für die API `SynthesizeSpeech`.

SDK v2

```
/*
   Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
   SPDX-License-Identifier: Apache-2.0
*/

package com.example.polly;

import software.amazon.awssdk.auth.credentials.ProfileCredentialsProvider;
import software.amazon.awssdk.core.ResponseInputStream;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.polly.PollyClient;
import software.amazon.awssdk.services.polly.model.*;

import java.io.File;
import java.io.FileOutputStream;
import java.io.InputStream;

/**
 * Before running this Java V2 code example, set up your development environment,
 including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class SpeechMarksSample {

    public static void main(String args[]) {

        PollyClient polly = PollyClient.builder()
                .region(Region.US_WEST_2)
```

```
                .credentialsProvider(ProfileCredentialsProvider.create())
                .build();

        speechMarksSample(polly) ;
        polly.close();
    }

    private static final String OUTPUT_FILE = "./speechMarks.json";
    public static void speechMarksSample(PollyClient client) {

        try {
            SynthesizeSpeechRequest speechMarksSampleRequest =
 SynthesizeSpeechRequest.builder()
                    .outputFormat(OutputFormat.JSON)
                    .speechMarkTypes(SpeechMarkType.VISEME, SpeechMarkType.WORD)
                    .voiceId(VoiceId.JOANNA)
                    .text("This is a sample text to be synthesized")
                    .build();
            try (FileOutputStream outputStream = new FileOutputStream(new
 File(OUTPUT_FILE))) {
                ResponseInputStream<SynthesizeSpeechResponse> synthesizeSpeechResponse
 = client
                        .synthesizeSpeech(speechMarksSampleRequest);
                byte[] buffer = new byte[2 * 1024];
                int readBytes;

                try (InputStream in = synthesizeSpeechResponse){
                    while ((readBytes = in.read(buffer)) > 0) {
                        outputStream.write(buffer, 0, readBytes);
                    }
                }
            } catch (Exception e) {
                System.err.println("Exception caught: " + e);
            }

        } catch (PollyException e) {
            System.err.println("Exception caught: " + e);
            System.exit(1);
        }
    }

}
```

SDK Version 1

```
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

Das folgende Java-Codebeispiel zeigt, wie man mit auf Java basierenden Anwendungen Sprachausgaben mit kürzeren Texten für eine echtzeitnahe Verarbeitung generiert.

Weitere Informationen finden Sie in der Referenz für API SynthesizeSpeech.

```
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-Beispiele

Die folgenden Codebeispiele zeigen, wie Sie Python-basierte (Boto3) -basierte Anwendungen verwenden, um verschiedene Aufgaben mit Amazon Polly zu erledigen. Diese Beispiele sind nicht als vollständige Beispiele gedacht, können aber in größeren Python-Anwendungen enthalten sein, die die AWS SDK für Python (Boto) verwenden.

Codeausschnitte

- DeleteLexicon
- GetLexicon
- ListLexicon
- PutLexicon
- StartSpeechSynthesisTask
- SynthesizeSpeech

DeleteLexicon

Das folgende Python-Codebeispiel verwendet das AWS SDK für Python (Boto) , um ein Lexikon in der Region zu löschen, die in Ihrer lokalen AWS Konfiguration angegeben ist. In dem Beispiel wird nur das angegebene Lexikon gelöscht. Dabei werden Sie dazu aufgefordert, zu bestätigen, dass Sie fortfahren möchten, bevor das Lexikon tatsächlich gelöscht wird.

Das folgende Codebeispiel verwendet Standardanmeldedaten, die in der AWS SDK-Konfigurationsdatei gespeichert sind. Weitere Informationen zum Erstellen der Konfigurationsdatei finden Sie unter Einrichtung des AWS CLI.

Weitere Informationen zu diesem Vorgang finden Sie in der Referenz für die API `DeleteLexicon`.

```
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

Der folgende Python-Code verwendet die AWS SDK für Python (Boto) , um alle in einer AWS Region gespeicherten Lexika abzurufen. In dem Beispiel wird ein Lexikon-Name als Befehlszeilenparameter akzeptiert, er ruft nur das Lexikon ab und gibt dabei den tmp-Pfad an, unter dem es lokal gespeichert wurde.

Das folgende Codebeispiel verwendet Standardanmeldedaten, die in der AWS SDK-Konfigurationsdatei gespeichert sind. Weitere Informationen zum Erstellen der Konfigurationsdatei finden Sie unter Einrichtung des AWS CLI.

Weitere Informationen zu diesem Vorgang finden Sie in der Referenz für die API `GetLexicon`.

```
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

Das folgende Python-Codebeispiel verwendet die AWS SDK für Python (Boto) , um die Lexika in Ihrem Konto in der Region aufzulisten, die in Ihrer lokalen AWS Konfiguration angegeben ist. Weitere Informationen zum Erstellen der Konfigurationsdatei finden Sie unter Einrichtung des AWS CLI.

Weitere Informationen zu diesem Vorgang finden Sie in der Referenz für die API `ListLexicons`.

```
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

Das folgende Codebeispiel zeigt, wie Python-basierte (Boto3) -basierte Anwendungen verwendet werden, um ein Aussprachelexikon in einer Region zu speichern. AWS

Weitere Informationen zu diesem Vorgang finden Sie in der Referenz für die API `PutLexicon`.

Beachten Sie Folgendes:

- Sie müssen den Code aktualisieren, indem Sie einen lokalen Lexikon-Dateinamen und einen gespeicherten Lexikon-Namen angeben.
- In diesem Beispiel wird davon ausgegangen, dass Sie in einem Unterverzeichnis mit der Bezeichnung `pls` Lexikon-Dateien erstellt haben. Sie müssen den Pfad ggf. aktualisieren.

Das folgende Codebeispiel verwendet Standardanmeldedaten, die in der AWS SDK-Konfigurationsdatei gespeichert sind. Weitere Informationen zum Erstellen der Konfigurationsdatei finden Sie unter Einrichtung des AWS CLI.

Weitere Informationen zu diesem Vorgang finden Sie in der Referenz für die API `PutLexicon`.

```
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

Das folgende Python-Codebeispiel verwendet die AWS SDK für Python (Boto) , um die Lexika in Ihrem Konto in der Region aufzulisten, die in Ihrer lokalen AWS Konfiguration angegeben ist. Weitere Informationen zum Erstellen der Konfigurationsdatei finden Sie unter Einrichtung des AWS CLI.

Weitere Informationen finden Sie in der Referenz für API `StartSpeechSynthesisTask`.

```
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

Das folgende Python-Codebeispiel verwendet die AWS SDK für Python (Boto) Synthesize-Sprache mit kürzeren Texten für die Verarbeitung nahezu in Echtzeit. Weitere Informationen finden Sie in der Referenz für den SynthesizeSpeech Vorgang.

In diesem Beispiel wird eine kurze Zeichenfolge mit Klartext verwendet. Sie können SSML-Text verwenden, um mehr Kontrolle über die Ausgabe zu haben. Weitere Informationen finden Sie unter Sprache aus SSML-Dokumenten generieren.

```
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()
```

Java-Beispiel

Dieses Beispiel zeigt, wie Amazon Polly verwendet wird, um Sprache aus einer Java-basierten Anwendung zu streamen. Das Beispiel verwendet die AWS SDK für Java, um den angegebenen Text mit einer Stimme zu lesen, die aus einer Liste ausgewählt wurde.

Der abgebildete Code deckt die wichtigsten Tasks ab, führt aber nur eine minimale Fehlerprüfung durch. Wenn Amazon Polly auf einen Fehler stößt, wird die Anwendung beendet.

Zur Ausführung dieser Beispielanwendung benötigen Sie Folgendes:

- Java 8 Java Development Kit (JDK)
- AWS SDK für Java
- Apache Maven

So testen Sie die Anwendung

1. Stellen Sie sicher, dass für das JDK die Umgebungsvariable "JAVA_HOME" gesetzt ist.

   Haben Sie beispielsweise JDK 1.8.0_121 unter Windows unter dem Pfad `C:\Program Files \Java\jdk1.8.0_121` installiert, geben Sie Folgendes in die Eingabeaufforderung ein:

   ```
   set JAVA_HOME=""C:\Program Files\Java\jdk1.8.0_121""
   ```

   Wenn Sie JDK 1.8.0_121 unter Linux unter dem Pfad `/usr/lib/jvm/java8-openjdk-amd64` installiert haben, geben Sie Folgendes in die Eingabeaufforderung ein:

   ```
   export JAVA_HOME=/usr/lib/jvm/java8-openjdk-amd64
   ```

2. Legen Sie die Maven-Umgebungsvariablen so fest, dass Maven über die Befehlszeile ausgeführt wird.

   Haben Sie beispielsweise Maven 3.3.9 unter Windows unter dem Pfad `C:\Program Files \apache-maven-3.3.9` installiert, geben Sie Folgendes ein:

   ```
   set M2_HOME=""C:\Program Files\apache-maven-3.3.9""
   set M2=%M2_HOME%\bin
   set PATH=%M2%;%PATH%
   ```

   Haben Sie Maven 3.3.9 unter Linux unter dem Pfad `/home/ec2-user/opt/apache-maven-3.3.9` installiert, geben Sie Folgendes ein:

   ```
   export M2_HOME=/home/ec2-user/opt/apache-maven-3.3.9
   export M2=$M2_HOME/bin
   export PATH=$M2:$PATH
   ```

3. Erstellen Sie ein neues Verzeichnis mit dem Namen `polly-java-demo`.
4. Erstellen Sie im Verzeichnis `polly-java-demo` eine neue Datei mit dem Namen `pom.xml` und fügen Sie den folgenden Code in diese Datei ein:

```
<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. Erstellen Sie ein neues Verzeichnis mit dem Namen `polly` unter `src/main/java/com/amazonaws/demos`.
6. Erstellen Sie im Verzeichnis `polly` eine neue Java-Quelldatei mit dem Namen `PollyDemo.java` und fügen Sie den folgenden Code in diese Datei ein:

```
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();

 }
}
```

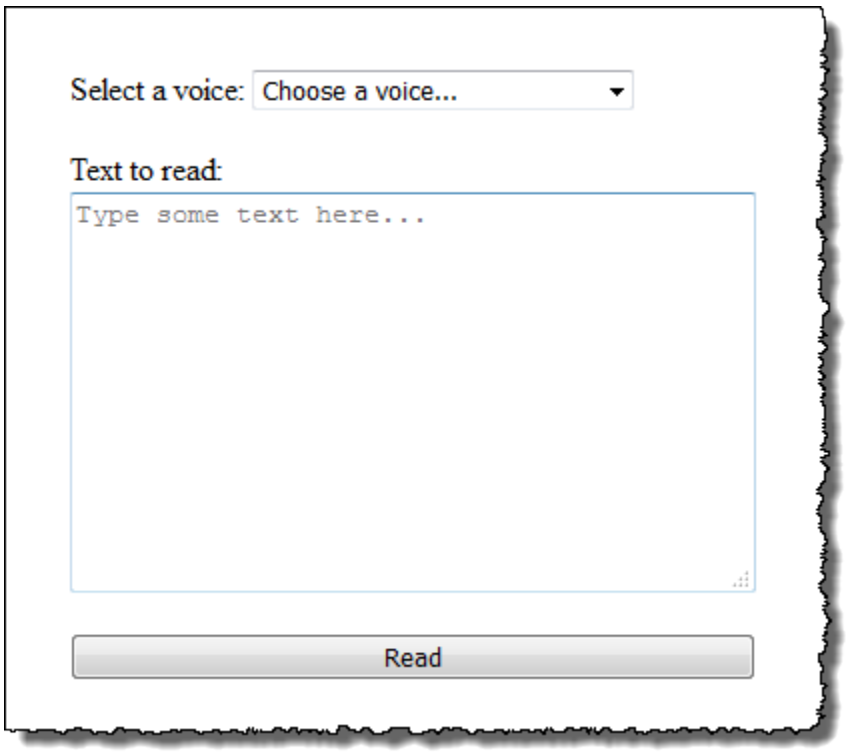
7. Wechseln Sie zurück in das Verzeichnis `polly-java-demo`, um die Demo zu bereinigen, zu kompilieren und auszuführen:

```
mvn clean compile exec:java
```

Python-Beispiel (HTML5 Client und Python-Server)

Diese Beispielanwendung enthält:

- Einen HTTP 1.1-Server, der HTTP Chunked Transfer Coding einsetzt (siehe Chunked Transfer Coding)
- Eine einfache HTML5 Benutzeroberfläche, die mit dem HTTP 1.1-Server interagiert (siehe unten):



Das Ziel dieses Beispiels besteht darin, zu zeigen, wie Amazon Polly verwendet wird, um Sprache aus einer browserbasierten HTML5 Anwendung zu streamen. Die Verwendung des von Amazon Polly produzierten Audiostreams bei der Synthese des Textes ist der empfohlene Ansatz für Anwendungsfälle, in denen Reaktionsfähigkeit ein wichtiger Faktor ist (z. B. Dialogsysteme, Screenreader usw.).

Zur Ausführung dieser Beispielanwendung benötigen Sie Folgendes:

- Webbrowser, der den Standards 1 HTML5 und EcmaScript 5 entspricht (z. B. Chrome 23.0 oder höher, Firefox 21.0 oder höher, Internet Explorer 9.0 oder höher)
- Python in einer höheren Version als 3.0

So testen Sie die Anwendung

1. Speichern Sie den Server-Code als Datei `server.py`. Sie finden den Code unter Python-Beispiel: Python-Servercode (server.py).
2. Speichern Sie den HTML5 Client-Code unter. `index.html` Sie finden den Code unter Python-Beispiel: HTML5 Benutzerschnittstelle (index.html).
3. Navigieren Sie zu dem Pfad, unter dem Sie die Datei "server.py" gespeichert haben, und führen Sie dort den nachfolgenden Befehl aus, um die Anwendung zu starten. (Auf einigen Systemen müssen Sie in diesem Befehl möglicherweise `python3` statt `python` verwenden.)

   ```
   $ python  server.py
   ```

   Sobald die Anwendung startet, wird eine URL im Terminal angezeigt.
4. Öffnen Sie die im Terminal angezeigte URL in einem Webbrowser.

   Sie können die Adresse und den Port, die der Anwendungsserver verwenden soll, als Parameter an `server.py` übergeben. Führen Sie `python server.py -h` aus, um weitere Informationen zu erhalten.
5. Um eine Sprachausgabe abzuspielen, wählen Sie eine Stimme aus der Liste aus, geben Text ein und klicken auf Read. Die Sprache wird abgespielt, sobald Amazon Polly den ersten verwendbaren Teil der Audiodaten überträgt.
6. Um den Python-Server nach Abschluss des Anwendungstests anzuhalten, drücken Sie Strg+C in dem Terminal, in dem der Server ausgeführt wird.

> **Note**
>
> Der Server erstellt mithilfe des AWS SDK für Python (Boto) einen Boto3-Client. Der Client verwendet die in der AWS Konfigurationsdatei auf Ihrem Computer gespeicherten Anmeldeinformationen, um die Anfragen an Amazon Polly zu signieren und zu authentifizieren. Weitere Informationen zum Erstellen der AWS Konfigurationsdatei und zum Speichern der Anmeldeinformationen finden Sie unter Konfiguration von AWS Command Line Interface im AWS Command Line Interface Benutzerhandbuch.

Python-Beispiel: HTML5 Benutzerschnittstelle (index.html)

Dieser Abschnitt enthält den unter beschriebenen Code für den HTML5 ClientPython-Beispiel (HTML5 Client und Python-Server).

```
<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',
            'mulaw': 'audio/mulaw',
            'alaw': 'audio/alaw'
        };

        /**
         * 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-Beispiel: Python-Servercode (server.py)

In diesem Abschnitt finden Sie den Code für den unter Python-Beispiel (HTML5 Client und Python-Server) beschriebenen Python-Server.

```
"""
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",
                 "mulaw": "audio/mulaw",
                 "alaw": "audio/alaw"}
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()
```

iOS-Beispiel

Im folgenden Beispiel wird das iOS-SDK für Amazon Polly verwendet, um den angegebenen Text mit einer Stimme zu lesen, die aus einer Liste von Stimmen ausgewählt wurde.

Der abgebildete Code deckt die wichtigsten Tasks ab, korrigiert jedoch keine Fehler. Den vollständigen Code finden Sie in der AWS Mobile SDK for iOS Amazon Polly Polly-Demo.

Initialisieren

```
// 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()
```

Abrufen einer Liste der verfügbaren Stimmen

```
// 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
})
```

Generieren der Sprachausgabe

```
// 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-Beispiel

Im folgenden Beispiel wird das Android SDK für Amazon Polly verwendet, um den angegebenen Text mit einer Stimme zu lesen, die aus einer Liste von Stimmen ausgewählt wurde.

Der abgebildete Code deckt die wichtigsten Tasks ab, korrigiert jedoch keine Fehler. Den vollständigen Code finden Sie in der AWS Mobile SDK für Android Amazon Polly Polly-Demo.

Initialisieren

```
// 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);
```

Abrufen einer Liste der verfügbaren Stimmen

```
// 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();
```

Abrufen der URL des Audiostreams

```
// 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);
```

Abspielen der Sprachausgabe

```
// 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();
    }
});
```

Codebeispiele für Amazon Polly mit AWS SDKs

Die folgenden Codebeispiele zeigen, wie Amazon Polly mit einem AWS Software Development Kit (SDK) verwendet wird.

Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Während Aktionen Ihnen zeigen, wie Sie einzelne Service-Funktionen aufrufen, können Sie Aktionen im Kontext der zugehörigen Szenarien anzeigen.

Szenarien sind Codebeispiele, die Ihnen zeigen, wie Sie bestimmte Aufgaben ausführen, indem Sie mehrere Funktionen innerhalb eines Service aufrufen oder mit anderen AWS-Services kombinieren.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterAmazon Polly mit einem AWS SDK verwenden. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Codebeispiele

- Grundlegende Beispiele für die Verwendung von Amazon Polly AWS SDKs
  - Aktionen für Amazon Polly mit AWS SDKs
    - Verwenden Sie DeleteLexiconmit einem AWS SDK oder CLI
    - Verwenden Sie DescribeVoicesmit einem AWS SDK
    - Verwenden Sie GetLexiconmit einem AWS SDK oder CLI
    - Verwenden Sie GetSpeechSynthesisTaskmit einem AWS SDK oder CLI
    - Verwenden Sie ListLexiconsmit einem AWS SDK oder CLI
    - Verwenden Sie ListSpeechSynthesisTasksmit einem AWS SDK oder CLI
    - Verwenden Sie PutLexiconmit einem AWS SDK oder CLI
    - Verwenden Sie StartSpeechSynthesisTaskmit einem AWS SDK oder CLI
    - Verwenden Sie SynthesizeSpeechmit einem AWS SDK
- Szenarien für die Verwendung von Amazon Polly AWS SDKs
  - Konvertieren Sie Text in Sprache und zurück in Text mit einem AWS SDK
  - Erstellen Sie eine Lippensynchronisationsanwendung mit Amazon Polly mithilfe eines AWS SDK
  - Erstellen einer Anwendung, die Kundenfeedback analysiert und Audio generiert
  - Erste Schritte mit der Text-zu-Sprache-Synthese

Grundlegende Beispiele für die Verwendung von Amazon Polly AWS SDKs

Die folgenden Codebeispiele zeigen, wie die Grundlagen von Amazon Polly mit AWS SDKs verwendet werden.

Beispiele

- Aktionen für Amazon Polly mit AWS SDKs
  - Verwenden Sie DeleteLexiconmit einem AWS SDK oder CLI
  - Verwenden Sie DescribeVoicesmit einem AWS SDK
  - Verwenden Sie GetLexiconmit einem AWS SDK oder CLI
  - Verwenden Sie GetSpeechSynthesisTaskmit einem AWS SDK oder CLI
  - Verwenden Sie ListLexiconsmit einem AWS SDK oder CLI
  - Verwenden Sie ListSpeechSynthesisTasksmit einem AWS SDK oder CLI
  - Verwenden Sie PutLexiconmit einem AWS SDK oder CLI
  - Verwenden Sie StartSpeechSynthesisTaskmit einem AWS SDK oder CLI
  - Verwenden Sie SynthesizeSpeechmit einem AWS SDK

Aktionen für Amazon Polly mit AWS SDKs

Die folgenden Codebeispiele zeigen, wie einzelne Amazon Polly Polly-Aktionen mit AWS SDKs ausgeführt werden. Jedes Beispiel enthält einen Link zu GitHub, wo Sie Anweisungen zum Einrichten und Ausführen des Codes finden.

Diese Auszüge rufen die Amazon Polly API auf und sind Codeauszüge aus größeren Programmen, die im Kontext ausgeführt werden müssen. Sie können Aktionen im Kontext unter Szenarien für die Verwendung von Amazon Polly AWS SDKs anzeigen.

Die folgenden Beispiele enthalten nur die am häufigsten verwendeten Aktionen. Eine vollständige Liste finden Sie in der API-Referenz für Amazon Polly.

Beispiele

- Verwenden Sie DeleteLexiconmit einem AWS SDK oder CLI
- Verwenden Sie DescribeVoicesmit einem AWS SDK

- Verwenden Sie GetLexiconmit einem AWS SDK oder CLI
- Verwenden Sie GetSpeechSynthesisTaskmit einem AWS SDK oder CLI
- Verwenden Sie ListLexiconsmit einem AWS SDK oder CLI
- Verwenden Sie ListSpeechSynthesisTasksmit einem AWS SDK oder CLI
- Verwenden Sie PutLexiconmit einem AWS SDK oder CLI
- Verwenden Sie StartSpeechSynthesisTaskmit einem AWS SDK oder CLI
- Verwenden Sie SynthesizeSpeechmit einem AWS SDK

Verwenden Sie **`DeleteLexicon`**mit einem AWS SDK oder CLI

Die folgenden Code-Beispiele zeigen, wie `DeleteLexicon` verwendet wird.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Erste Schritte mit der Text-zu-Sprache-Synthese

.NET

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
using System;
using System.Threading.Tasks;
using Amazon.Polly;
using Amazon.Polly.Model;

/// <summary>
/// Deletes an existing Amazon Polly lexicon using the AWS SDK for .NET.
/// </summary>
public class DeleteLexicon
{
    public static async Task Main()
```

```
        {
            string lexiconName = "SampleLexicon";

            var client = new AmazonPollyClient();

            var success = await DeletePollyLexiconAsync(client, lexiconName);

            if (success)
            {
                Console.WriteLine($"Successfully deleted {lexiconName}.");
            }
            else
            {
                Console.WriteLine($"Could not delete {lexiconName}.");
            }
        }

        /// <summary>
        /// Deletes the named Amazon Polly lexicon.
        /// </summary>
        /// <param name="client">The initialized Amazon Polly client object.</
param>
        /// <param name="lexiconName">The name of the Amazon Polly lexicon to
        /// delete.</param>
        /// <returns>A Boolean value indicating the success of the operation.</
returns>
        public static async Task<bool> DeletePollyLexiconAsync(
            AmazonPollyClient client,
            string lexiconName)
        {
            var deleteLexiconRequest = new DeleteLexiconRequest()
            {
                Name = lexiconName,
            };

            var response = await client.DeleteLexiconAsync(deleteLexiconRequest);

            return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
        }
    }
```

- Einzelheiten zur API finden Sie DeleteLexicon in der AWS SDK für .NET API-Referenz.

CLI

AWS CLI

So löschen Sie ein Lexikon

Im folgenden Beispiel für `delete-lexicon` wird das angegebene Lexikon gelöscht.

```
aws polly delete-lexicon \
    --name w3c
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Weitere Informationen finden Sie unter Verwenden des DeleteLexicon Vorgangs im Amazon Polly Developer Guide.

- Einzelheiten zur API finden Sie DeleteLexiconin der AWS CLI Befehlsreferenz.

SAP ABAP

SDK für SAP ABAP

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
TRY.
    lo_ply->deletelexicon( iv_name ).
    MESSAGE 'Lexicon deleted successfully.' TYPE 'I'.
  CATCH /aws1/cx_plylexiconnotfoundex.
    MESSAGE 'Lexicon not found.' TYPE 'E'.
  CATCH /aws1/cx_plyservicefailureex.
    MESSAGE 'Service failure occurred.' TYPE 'E'.
ENDTRY.
```

- Einzelheiten zur API finden Sie DeleteLexiconin der API-Referenz zum AWS SDK für SAP ABAP.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterAmazon Polly mit einem AWS SDK verwenden. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwenden Sie **`DescribeVoices`**mit einem AWS SDK

Die folgenden Code-Beispiele zeigen, wie `DescribeVoices` verwendet wird.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Erste Schritte mit der Text-zu-Sprache-Synthese

.NET

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
using System;
using System.Threading.Tasks;
using Amazon.Polly;
using Amazon.Polly.Model;

public class DescribeVoices
{
    public static async Task Main()
    {
        var client = new AmazonPollyClient();

        var allVoicesRequest = new DescribeVoicesRequest();
        var enUsVoicesRequest = new DescribeVoicesRequest()
        {
            LanguageCode = "en-US",
        };

        try
```

```
            {
                string nextToken;
                do
                {
                    var allVoicesResponse = await
client.DescribeVoicesAsync(allVoicesRequest);
                    nextToken = allVoicesResponse.NextToken;
                    allVoicesRequest.NextToken = nextToken;

                    Console.WriteLine("\nAll voices: ");
                    allVoicesResponse.Voices.ForEach(voice =>
                    {
                        DisplayVoiceInfo(voice);
                    });
                }
                while (nextToken is not null);

                do
                {
                    var enUsVoicesResponse = await
client.DescribeVoicesAsync(enUsVoicesRequest);
                    nextToken = enUsVoicesResponse.NextToken;
                    enUsVoicesRequest.NextToken = nextToken;

                    Console.WriteLine("\nen-US voices: ");
                    enUsVoicesResponse.Voices.ForEach(voice =>
                    {
                        DisplayVoiceInfo(voice);
                    });
                }
                while (nextToken is not null);
            }
            catch (Exception ex)
            {
                Console.WriteLine("Exception caught: " + ex.Message);
            }
        }

        public static void DisplayVoiceInfo(Voice voice)
        {
            Console.WriteLine($" Name: {voice.Name}\tGender:
{voice.Gender}\tLanguageName: {voice.LanguageName}");
        }
    }
```

- Einzelheiten zur API finden Sie DescribeVoicesin der AWS SDK für .NET API-Referenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.polly.PollyClient;
import software.amazon.awssdk.services.polly.model.DescribeVoicesRequest;
import software.amazon.awssdk.services.polly.model.DescribeVoicesResponse;
import software.amazon.awssdk.services.polly.model.PollyException;
import software.amazon.awssdk.services.polly.model.Voice;
import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class DescribeVoicesSample {
    public static void main(String args[]) {
        PollyClient polly = PollyClient.builder()
                .region(Region.US_WEST_2)
                .build();

        describeVoice(polly);
        polly.close();
    }
```

```
    public static void describeVoice(PollyClient polly) {
        try {
            DescribeVoicesRequest voicesRequest = DescribeVoicesRequest.builder()
                    .languageCode("en-US")
                    .build();

            DescribeVoicesResponse enUsVoicesResult =
 polly.describeVoices(voicesRequest);
            List<Voice> voices = enUsVoicesResult.voices();
            for (Voice myVoice : voices) {
                System.out.println("The ID of the voice is " + myVoice.id());
                System.out.println("The gender of the voice is " +
 myVoice.gender());
            }

        } catch (PollyException e) {
            System.err.println("Exception caught: " + e);
            System.exit(1);
        }
    }
}
```

- Einzelheiten zur API finden Sie DescribeVoicesin der AWS SDK for Java 2.x API-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
class PollyWrapper:
    """Encapsulates Amazon Polly functions."""

    def __init__(self, polly_client, s3_resource):
        """
        :param polly_client: A Boto3 Amazon Polly client.
```

```
        :param s3_resource: A Boto3 Amazon Simple Storage Service (Amazon S3)
resource.
        """
        self.polly_client = polly_client
        self.s3_resource = s3_resource
        self.voice_metadata = None


    def describe_voices(self):
        """
        Gets metadata about available voices.

        :return: The list of voice metadata.
        """
        try:
            response = self.polly_client.describe_voices()
            self.voice_metadata = response["Voices"]
            logger.info("Got metadata about %s voices.",
len(self.voice_metadata))
        except ClientError:
            logger.exception("Couldn't get voice metadata.")
            raise
        else:
            return self.voice_metadata
```

- Einzelheiten zur API finden Sie DescribeVoicesin AWS SDK for Python (Boto3) API Reference.

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
require 'aws-sdk-polly' # In v2: require 'aws-sdk'

begin
  # Create an Amazon Polly client using
  # credentials from the shared credentials file ~/.aws/credentials
  # and the configuration (region) from the shared configuration file ~/.aws/
config
  polly = Aws::Polly::Client.new

  # Get US English voices
  resp = polly.describe_voices(language_code: 'en-US')

  resp.voices.each do |v|
    puts v.name
    puts "  #{v.gender}"
    puts
  end
rescue StandardError => e
  puts 'Could not get voices'
  puts 'Error message:'
  puts e.message
end
```

- Einzelheiten zur API finden Sie DescribeVoicesin der AWS SDK für Ruby API-Referenz.

Rust

SDK für Rust

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
async fn list_voices(client: &Client) -> Result<(), Error> {
    let resp = client.describe_voices().send().await?;

    println!("Voices:");

    let voices = resp.voices();
```

```
    for voice in voices {
        println!("  Name:     {}", voice.name().unwrap_or("No name!"));
        println!(
            "  Language: {}",
            voice.language_name().unwrap_or("No language!")
        );

        println!();
    }

    println!("Found {} voices", voices.len());

    Ok(())
}
```

- Einzelheiten zur API finden Sie DescribeVoicesin der API-Referenz zum AWS SDK für Rust.

SAP ABAP

SDK für SAP ABAP

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
    TRY.
        " Only pass optional parameters if they have values
        IF iv_engine IS NOT INITIAL AND iv_language IS NOT INITIAL.
          oo_result = lo_ply->describevoices(
            iv_engine = iv_engine
            iv_languagecode = iv_language ).
        ELSEIF iv_engine IS NOT INITIAL.
          oo_result = lo_ply->describevoices(
            iv_engine = iv_engine ).
        ELSEIF iv_language IS NOT INITIAL.
          oo_result = lo_ply->describevoices(
            iv_languagecode = iv_language ).
        ELSE.
          oo_result = lo_ply->describevoices( ).
```

```
        ENDIF.
        MESSAGE 'Retrieved voice metadata.' TYPE 'I'.
      CATCH /aws1/cx_plyinvalidnexttokenex.
        MESSAGE 'The NextToken is invalid.' TYPE 'E'.
      CATCH /aws1/cx_plyservicefailureex.
        MESSAGE 'Service failure occurred.' TYPE 'E'.
    ENDTRY.
```

- Einzelheiten zur API finden Sie DescribeVoicesin der API-Referenz zum AWS SDK für SAP ABAP.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterAmazon Polly mit einem AWS SDK verwenden. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwenden Sie **`GetLexicon`**mit einem AWS SDK oder CLI

Die folgenden Code-Beispiele zeigen, wie `GetLexicon` verwendet wird.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Erste Schritte mit der Text-zu-Sprache-Synthese

.NET

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
using System;
using System.Threading.Tasks;
using Amazon.Polly;
using Amazon.Polly.Model;
```

```
    /// <summary>
    /// Retrieves information about a specific Amazon Polly lexicon.
    /// </summary>
    public class GetLexicon
    {
        public static async Task Main(string[] args)
        {
            string lexiconName = "SampleLexicon";

            var client = new AmazonPollyClient();

            await GetPollyLexiconAsync(client, lexiconName);
        }

        public static async Task GetPollyLexiconAsync(AmazonPollyClient client,
string lexiconName)
        {
            var getLexiconRequest = new GetLexiconRequest()
            {
                Name = lexiconName,
            };

            try
            {
                var response = await client.GetLexiconAsync(getLexiconRequest);
                Console.WriteLine($"Lexicon:\n Name: {response.Lexicon.Name}");
                Console.WriteLine($"Content: {response.Lexicon.Content}");
            }
            catch (Exception ex)
            {
                Console.WriteLine("Error: " + ex.Message);
            }
        }
    }
```

- Einzelheiten zur API finden Sie GetLexiconin der AWS SDK für .NET API-Referenz.

CLI

AWS CLI

So rufen Sie den Inhalt eines Lexikons ab

Im folgenden Beispiel für `get-lexicon` wird der Inhalt des angegebenen Aussprache-Lexikons abgerufen.

```
aws polly get-lexicon \
    --name w3c
```

Ausgabe:

```
{
    "Lexicon": {
        "Content": "<?xml version=\"1.0\" encoding=\"UTF-8\"?>\n<lexicon version=
\"1.0\" \n      xmlns=    \"http://www.w3.org/2005/01/pronunciation-lexicon
\"\n      xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\" \n
 xsi:schemaLocation=\"http://www.w3.org/2005/01/pronunciation-lexicon \n
 http://www.w3.org/TR/2007/CR-pronunciation-    lexicon-20071212/pls.xsd\"\n
   alphabet=\"ipa\" \n      xml:lang=\"en-US\">\n  <lexeme>\n    <grapheme>W3C</
grapheme>\n        <alias>World Wide Web Consortium</alias>\n  </lexeme>\n</
lexicon>\n",
        "Name": "w3c"
    },
    "LexiconAttributes": {
        "Alphabet": "ipa",
        "LanguageCode": "en-US",
        "LastModified": 1603908910.99,
        "LexiconArn": "arn:aws:polly:us-west-2:880185128111:lexicon/w3c",
        "LexemesCount": 1,
        "Size": 492
    }
}
```

Weitere Informationen finden Sie unter Verwenden des GetLexicon Vorgangs im Amazon Polly Developer Guide.

- Einzelheiten zur API finden Sie GetLexiconin der AWS CLI Befehlsreferenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
class PollyWrapper:
    """Encapsulates Amazon Polly functions."""

    def __init__(self, polly_client, s3_resource):
        """
        :param polly_client: A Boto3 Amazon Polly client.
        :param s3_resource: A Boto3 Amazon Simple Storage Service (Amazon S3)
 resource.
        """
        self.polly_client = polly_client
        self.s3_resource = s3_resource
        self.voice_metadata = None


    def get_lexicon(self, name):
        """
        Gets metadata and contents of an existing lexicon.

        :param name: The name of the lexicon to retrieve.
        :return: The retrieved lexicon.
        """
        try:
            response = self.polly_client.get_lexicon(Name=name)
            logger.info("Got lexicon %s.", name)
        except ClientError:
            logger.exception("Couldn't get lexicon %s.", name)
            raise
        else:
            return response
```

- Einzelheiten zur API finden Sie GetLexiconin AWS SDK for Python (Boto3) API Reference.

SAP ABAP

SDK für SAP ABAP

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
TRY.
    oo_result = lo_ply->getlexicon( iv_name ).
    DATA(lo_lexicon) = oo_result->get_lexicon( ).
    IF lo_lexicon IS BOUND.
      DATA(lv_lex_name) = lo_lexicon->get_name( ).
      MESSAGE |Retrieved lexicon: { lv_lex_name }| TYPE 'I'.
    ENDIF.
  CATCH /aws1/cx_plylexiconnotfoundex.
    MESSAGE 'Lexicon not found.' TYPE 'E'.
  CATCH /aws1/cx_plyservicefailureex.
    MESSAGE 'Service failure occurred.' TYPE 'E'.
ENDTRY.
```

- Einzelheiten zur API finden Sie GetLexiconin der API-Referenz zum AWS SDK für SAP ABAP.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterAmazon Polly mit einem AWS SDK verwenden. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwenden Sie **`GetSpeechSynthesisTask`**mit einem AWS SDK oder CLI

Die folgenden Code-Beispiele zeigen, wie `GetSpeechSynthesisTask` verwendet wird.

CLI

AWS CLI

So rufen Sie Informationen über eine Sprachsyntheseaufgabe ab

Im folgenden Beispiel für `get-speech-synthesis-task` werden Informationen über die angegebene Sprachsyntheseaufgabe abgerufen.

```
aws polly get-speech-synthesis-task \
    --task-id 70b61c0f-57ce-4715-a247-cae8729dcce9
```

Ausgabe:

```
{
    "SynthesisTask": {
        "TaskId": "70b61c0f-57ce-4715-a247-cae8729dcce9",
        "TaskStatus": "completed",
        "OutputUri": "https://s3.us-west-2.amazonaws.com/amzn-s3-demo-
bucket/70b61c0f-57ce-4715-a247-cae8729dcce9.mp3",
        "CreationTime": 1603911042.689,
        "RequestCharacters": 1311,
        "OutputFormat": "mp3",
        "TextType": "text",
        "VoiceId": "Joanna"
    }
}
```

Weitere Informationen finden Sie unter Erstellen von langen Audiodateien im Benutzerhandbuch für Amazon Polly.

- Einzelheiten zur API finden Sie GetSpeechSynthesisTaskin der AWS CLI Befehlsreferenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
class PollyWrapper:
    """Encapsulates Amazon Polly functions."""

    def __init__(self, polly_client, s3_resource):
        """
        :param polly_client: A Boto3 Amazon Polly client.
        :param s3_resource: A Boto3 Amazon Simple Storage Service (Amazon S3)
 resource.
        """
        self.polly_client = polly_client
        self.s3_resource = s3_resource
        self.voice_metadata = None


    def get_speech_synthesis_task(self, task_id):
        """
        Gets metadata about an asynchronous speech synthesis task, such as its
 status.

        :param task_id: The ID of the task to retrieve.
        :return: Metadata about the task.
        """
        try:
            response =
 self.polly_client.get_speech_synthesis_task(TaskId=task_id)
            task = response["SynthesisTask"]
            logger.info("Got synthesis task. Status is %s.", task["TaskStatus"])
        except ClientError:
            logger.exception("Couldn't get synthesis task %s.", task_id)
            raise
        else:
            return task
```

- Einzelheiten zur API finden Sie GetSpeechSynthesisTaskin AWS SDK for Python (Boto3) API Reference.

SAP ABAP

SDK für SAP ABAP

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
TRY.
    oo_result = lo_ply->getspeechsynthesistask( iv_task_id ).
    DATA(lo_task) = oo_result->get_synthesistask( ).
    IF lo_task IS BOUND.
      DATA(lv_status) = lo_task->get_taskstatus( ).
      MESSAGE |Task status: { lv_status }| TYPE 'I'.
    ENDIF.
  CATCH /aws1/cx_plyinvalidtaskidex.
    MESSAGE 'Invalid task ID.' TYPE 'E'.
  CATCH /aws1/cx_plyservicefailureex.
    MESSAGE 'Service failure occurred.' TYPE 'E'.
  CATCH /aws1/cx_plysynthesistsknotf00.
    MESSAGE 'Synthesis task not found.' TYPE 'E'.
ENDTRY.
```

- Einzelheiten zur API finden Sie GetSpeechSynthesisTaskin der API-Referenz zum AWS SDK für SAP ABAP.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterAmazon Polly mit einem AWS SDK verwenden. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwenden Sie **`ListLexicons`**mit einem AWS SDK oder CLI

Die folgenden Code-Beispiele zeigen, wie `ListLexicons` verwendet wird.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Erste Schritte mit der Text-zu-Sprache-Synthese

.NET

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
using System;
using System.Threading.Tasks;
using Amazon.Polly;
using Amazon.Polly.Model;

/// <summary>
/// Lists the Amazon Polly lexicons that have been defined. By default,
/// lists the lexicons that are defined in the same AWS Region as the default
/// user. To view Amazon Polly lexicons that are defined in a different AWS
/// Region, supply it as a parameter to the Amazon Polly constructor.
/// </summary>
public class ListLexicons
{
    public static async Task Main()
    {
        var client = new AmazonPollyClient();
        var request = new ListLexiconsRequest();

        try
        {
            Console.WriteLine("All voices: ");

            do
            {
                var response = await client.ListLexiconsAsync(request);
                request.NextToken = response.NextToken;

                response.Lexicons.ForEach(lexicon =>
                {
                    var attributes = lexicon.Attributes;
                    Console.WriteLine($"Name: {lexicon.Name}");
                    Console.WriteLine($"\tAlphabet: {attributes.Alphabet}");
```

```
                        Console.WriteLine($"\tLanguageCode:
{attributes.LanguageCode}");
                        Console.WriteLine($"\tLastModified:
{attributes.LastModified}");
                        Console.WriteLine($"\tLexemesCount:
{attributes.LexemesCount}");
                        Console.WriteLine($"\tLexiconArn:
{attributes.LexiconArn}");
                        Console.WriteLine($"\tSize: {attributes.Size}");
                    });
                }
                while (request.NextToken is not null);
            }
            catch (Exception ex)
            {
                Console.WriteLine($"Error: {ex.Message}");
            }
        }
    }
```

- Einzelheiten zur API finden Sie ListLexiconsin der AWS SDK für .NET API-Referenz.

CLI

AWS CLI

So listen Sie Ihre Lexika auf

Im folgenden Beispiel für `list-lexicons` werden Ihre Aussprachelexika aufgelistet.

```
aws polly list-lexicons
```

Ausgabe:

```
{
    "Lexicons": [
        {
            "Name": "w3c",
            "Attributes": {
                "Alphabet": "ipa",
                "LanguageCode": "en-US",
```

```
                "LastModified": 1603908910.99,
                "LexiconArn": "arn:aws:polly:us-east-2:123456789012:lexicon/w3c",
                "LexemesCount": 1,
                "Size": 492
            }
        }
    ]
}
```

Weitere Informationen finden Sie unter Verwenden des ListLexicons Vorgangs im Amazon Polly Developer Guide.

- Einzelheiten zur API finden Sie ListLexiconsin der AWS CLI Befehlsreferenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.polly.PollyClient;
import software.amazon.awssdk.services.polly.model.ListLexiconsResponse;
import software.amazon.awssdk.services.polly.model.ListLexiconsRequest;
import software.amazon.awssdk.services.polly.model.LexiconDescription;
import software.amazon.awssdk.services.polly.model.PollyException;
import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class ListLexicons {
```

```
    public static void main(String args[]) {
        PollyClient polly = PollyClient.builder()
                .region(Region.US_WEST_2)
                .build();

        listLexicons(polly);
        polly.close();
    }

    public static void listLexicons(PollyClient client) {
        try {
            ListLexiconsRequest listLexiconsRequest =
 ListLexiconsRequest.builder()
                    .build();

            ListLexiconsResponse listLexiconsResult =
 client.listLexicons(listLexiconsRequest);
            List<LexiconDescription> lexiconDescription =
 listLexiconsResult.lexicons();
            for (LexiconDescription lexDescription : lexiconDescription) {
                System.out.println("The name of the Lexicon is " +
 lexDescription.name());
            }

        } catch (PollyException e) {
            System.err.println("Exception caught: " + e);
            System.exit(1);
        }
    }
}
```

- Einzelheiten zur API finden Sie ListLexiconsin der AWS SDK for Java 2.x API-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
class PollyWrapper:
    """Encapsulates Amazon Polly functions."""

    def __init__(self, polly_client, s3_resource):
        """
        :param polly_client: A Boto3 Amazon Polly client.
        :param s3_resource: A Boto3 Amazon Simple Storage Service (Amazon S3)
 resource.
        """
        self.polly_client = polly_client
        self.s3_resource = s3_resource
        self.voice_metadata = None


    def list_lexicons(self):
        """
        Lists lexicons in the current account.

        :return: The list of lexicons.
        """
        try:
            response = self.polly_client.list_lexicons()
            lexicons = response["Lexicons"]
            logger.info("Got %s lexicons.", len(lexicons))
        except ClientError:
            logger.exception(
                "Couldn't get  %s.",
            )
            raise
        else:
            return lexicons
```

- Einzelheiten zur API finden Sie ListLexiconsin AWS SDK for Python (Boto3) API Reference.

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
require 'aws-sdk-polly' # In v2: require 'aws-sdk'

begin
  # Create an Amazon Polly client using
  # credentials from the shared credentials file ~/.aws/credentials
  # and the configuration (region) from the shared configuration file ~/.aws/
config
  polly = Aws::Polly::Client.new

  resp = polly.list_lexicons

  resp.lexicons.each do |l|
    puts l.name
    puts "  Alphabet:#{l.attributes.alphabet}"
    puts "  Language:#{l.attributes.language}"
    puts
  end
rescue StandardError => e
  puts 'Could not get lexicons'
  puts 'Error message:'
  puts e.message
end
```

- Einzelheiten zur API finden Sie ListLexiconsin der AWS SDK für Ruby API-Referenz.

Rust

SDK für Rust

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
async fn show_lexicons(client: &Client) -> Result<(), Error> {
    let resp = client.list_lexicons().send().await?;

    println!("Lexicons:");

    let lexicons = resp.lexicons();

    for lexicon in lexicons {
        println!("  Name:     {}", lexicon.name().unwrap_or_default());
        println!(
            "  Language: {:?}\n",
            lexicon
                .attributes()
                .as_ref()
                .map(|attrib| attrib
                    .language_code
                    .as_ref()
                    .expect("languages must have language codes"))
                .expect("languages must have attributes")
        );
    }

    println!();
    println!("Found {} lexicons.", lexicons.len());
    println!();

    Ok(())
}
```

- Einzelheiten zur API finden Sie ListLexiconsin der API-Referenz zum AWS SDK für Rust.

SAP ABAP

SDK für SAP ABAP

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
TRY.
    oo_result = lo_ply->listlexicons( ).
    DATA(lt_lexicons) = oo_result->get_lexicons( ).
    DATA(lv_count) = lines( lt_lexicons ).
    MESSAGE |Found { lv_count } lexicons| TYPE 'I'.
  CATCH /aws1/cx_plyinvalidnexttokenex.
    MESSAGE 'Invalid NextToken.' TYPE 'E'.
  CATCH /aws1/cx_plyservicefailureex.
    MESSAGE 'Service failure occurred.' TYPE 'E'.
ENDTRY.
```

- Einzelheiten zur API finden Sie ListLexiconsin der API-Referenz zum AWS SDK für SAP ABAP.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterAmazon Polly mit einem AWS SDK verwenden. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwenden Sie **`ListSpeechSynthesisTasks`**mit einem AWS SDK oder CLI

Die folgenden Code-Beispiele zeigen, wie `ListSpeechSynthesisTasks` verwendet wird.

CLI

AWS CLI

So listen Sie Ihre Sprachsyntheseaufgaben auf

Im folgenden Beispiel für `list-speech-synthesis-tasks` werden Ihre Sprachsyntheseaufgaben aufgelistet.

```
aws polly list-speech-synthesis-tasks
```

Ausgabe:

```
{
    "SynthesisTasks": [
        {
            "TaskId": "70b61c0f-57ce-4715-a247-cae8729dcce9",
            "TaskStatus": "completed",
            "OutputUri": "https://s3.us-west-2.amazonaws.com/amzn-s3-demo-
bucket/70b61c0f-57ce-4715-a247-cae8729dcce9.mp3",
            "CreationTime": 1603911042.689,
            "RequestCharacters": 1311,
            "OutputFormat": "mp3",
            "TextType": "text",
            "VoiceId": "Joanna"
        }
    ]
}
```

Weitere Informationen finden Sie unter Erstellen von langen Audiodateien im Benutzerhandbuch für Amazon Polly.

- Einzelheiten zur API finden Sie ListSpeechSynthesisTasksin der AWS CLI Befehlsreferenz.

SAP ABAP

SDK für SAP ABAP

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
    TRY.
        " Only pass optional parameters if they have values
        IF iv_max_results IS NOT INITIAL AND iv_status IS NOT INITIAL.
          oo_result = lo_ply->listspeechsynthesistasks(
            iv_maxresults = iv_max_results
```

```
          iv_status = iv_status ).
      ELSEIF iv_max_results IS NOT INITIAL.
        oo_result = lo_ply->listspeechsynthesistasks(
          iv_maxresults = iv_max_results ).
      ELSEIF iv_status IS NOT INITIAL.
        oo_result = lo_ply->listspeechsynthesistasks(
          iv_status = iv_status ).
      ELSE.
        oo_result = lo_ply->listspeechsynthesistasks( ).
      ENDIF.
      DATA(lt_tasks) = oo_result->get_synthesistasks( ).
      DATA(lv_count) = lines( lt_tasks ).
      MESSAGE |Found { lv_count } synthesis tasks| TYPE 'I'.
    CATCH /aws1/cx_plyinvalidnexttokenex.
      MESSAGE 'Invalid NextToken.' TYPE 'E'.
    CATCH /aws1/cx_plyservicefailureex.
      MESSAGE 'Service failure occurred.' TYPE 'E'.
  ENDTRY.
```

- Einzelheiten zur API finden Sie ListSpeechSynthesisTasksin der API-Referenz zum AWS SDK für SAP ABAP.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterAmazon Polly mit einem AWS SDK verwenden. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwenden Sie **`PutLexicon`**mit einem AWS SDK oder CLI

Die folgenden Code-Beispiele zeigen, wie `PutLexicon` verwendet wird.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Erste Schritte mit der Text-zu-Sprache-Synthese

.NET

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
    using System;
    using System.Threading.Tasks;
    using Amazon.Polly;
    using Amazon.Polly.Model;

    /// <summary>
    /// Creates a new Amazon Polly lexicon using the AWS SDK for .NET.
    /// </summary>
    public class PutLexicon
    {
        public static async Task Main()
        {
            string lexiconContent = "<?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>";
            string lexiconName = "SampleLexicon";

            var client = new AmazonPollyClient();
            var putLexiconRequest = new PutLexiconRequest()
            {
                Name = lexiconName,
                Content = lexiconContent,
            };
```

```
            try
            {
                var response = await client.PutLexiconAsync(putLexiconRequest);
                if (response.HttpStatusCode == System.Net.HttpStatusCode.OK)
                {
                    Console.WriteLine($"Successfully created Lexicon:
{lexiconName}.");
                }
                else
                {
                    Console.WriteLine($"Could not create Lexicon:
{lexiconName}.");
                }
            }
            catch (Exception ex)
            {
                Console.WriteLine("Exception caught: " + ex.Message);
            }
        }
    }
```

- Einzelheiten zur API finden Sie PutLexiconin der AWS SDK für .NET API-Referenz.

CLI

AWS CLI

So speichern Sie ein Lexikon

Im folgenden Beispiel für `put-lexicon` wird das angegebene Aussprachelexikon gespeichert. Die `example.pls` Datei spezifiziert ein PLS-compliant W3C-Lexikon.

```
aws polly put-lexicon \
    --name w3c \
    --content file://example.pls
```

Inhalt von `example.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>
}
```

Mit diesem Befehl wird keine Ausgabe zurückgegeben.

Weitere Informationen finden Sie unter Verwenden des PutLexicon Vorgangs im Amazon Polly Developer Guide.

- Einzelheiten zur API finden Sie PutLexiconin der AWS CLI Befehlsreferenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
class PollyWrapper:
    """Encapsulates Amazon Polly functions."""

    def __init__(self, polly_client, s3_resource):
        """
        :param polly_client: A Boto3 Amazon Polly client.
        :param s3_resource: A Boto3 Amazon Simple Storage Service (Amazon S3)
 resource.
        """
        self.polly_client = polly_client
        self.s3_resource = s3_resource
```

```
        self.voice_metadata = None


    def create_lexicon(self, name, content):
        """
        Creates a lexicon with the specified content. A lexicon contains custom
        pronunciations.

        :param name: The name of the lexicon.
        :param content: The content of the lexicon.
        """
        try:
            self.polly_client.put_lexicon(Name=name, Content=content)
            logger.info("Created lexicon %s.", name)
        except ClientError:
            logger.exception("Couldn't create lexicon %s.")
            raise
```

- Einzelheiten zur API finden Sie PutLexiconin AWS SDK for Python (Boto3) API Reference.

Rust

SDK für Rust

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
async fn make_lexicon(client: &Client, name: &str, from: &str, to: &str) ->
 Result<(), Error> {
    let content = format!("<?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>{}</grapheme><alias>{}</alias></lexeme>
    </lexicon>", from, to);
```

```
    client
        .put_lexicon()
        .name(name)
        .content(content)
        .send()
        .await?;

    println!("Added lexicon");

    Ok(())
}
```

- Einzelheiten zur API finden Sie PutLexiconin der API-Referenz zum AWS SDK für Rust.

SAP ABAP

SDK für SAP ABAP

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
    TRY.
        lo_ply->putlexicon(
          iv_name = iv_name
          iv_content = iv_content ).
        MESSAGE 'Lexicon created successfully.' TYPE 'I'.
      CATCH /aws1/cx_plyinvalidlexiconex.
        MESSAGE 'Invalid lexicon.' TYPE 'E'.
      CATCH /aws1/cx_plylexiconsizeexcdex.
        MESSAGE 'Lexicon size exceeded.' TYPE 'E'.
      CATCH /aws1/cx_plymaxlexemelengthe00.
        MESSAGE 'Maximum lexeme length exceeded.' TYPE 'E'.
      CATCH /aws1/cx_plymaxlexiconsnoexc00.
        MESSAGE 'Maximum number of lexicons exceeded.' TYPE 'E'.
      CATCH /aws1/cx_plyservicefailureex.
        MESSAGE 'Service failure occurred.' TYPE 'E'.
      CATCH /aws1/cx_plyunsuppedplsalpha00.
```

```
        MESSAGE 'Unsupported PLS alphabet.' TYPE 'E'.
      CATCH /aws1/cx_plyunsuppedplslangu00.
        MESSAGE 'Unsupported PLS language.' TYPE 'E'.
    ENDTRY.
```

- Einzelheiten zur API finden Sie PutLexiconin der API-Referenz zum AWS SDK für SAP ABAP.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterAmazon Polly mit einem AWS SDK verwenden. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwenden Sie **`StartSpeechSynthesisTask`**mit einem AWS SDK oder CLI

Die folgenden Code-Beispiele zeigen, wie `StartSpeechSynthesisTask` verwendet wird.

CLI

AWS CLI

So synthetisieren Sie Text

Im folgenden Beispiel für `start-speech-synthesis-task` wird der Text in `text_file.txt` synthetisiert und die resultierende MP3-Datei im angegebenen Bucket gespeichert .

```
aws polly start-speech-synthesis-task \
    --output-format mp3 \
    --output-s3-bucket-name amzn-s3-demo-bucket \
    --text  file://text_file.txt \
    --voice-id Joanna
```

Ausgabe:

```
{
    "SynthesisTask": {
        "TaskId": "70b61c0f-57ce-4715-a247-cae8729dcce9",
        "TaskStatus": "scheduled",
        "OutputUri": "https://s3.us-east-2.amazonaws.com/amzn-s3-demo-
bucket/70b61c0f-57ce-4715-a247-cae8729dcce9.mp3",
        "CreationTime": 1603911042.689,
```

```
        "RequestCharacters": 1311,
        "OutputFormat": "mp3",
        "TextType": "text",
        "VoiceId": "Joanna"
    }
}
```

Weitere Informationen finden Sie unter Erstellen von langen Audiodateien im Benutzerhandbuch für Amazon Polly.

- Einzelheiten zur API finden Sie StartSpeechSynthesisTaskin der AWS CLI Befehlsreferenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
class PollyWrapper:
    """Encapsulates Amazon Polly functions."""

    def __init__(self, polly_client, s3_resource):
        """
        :param polly_client: A Boto3 Amazon Polly client.
        :param s3_resource: A Boto3 Amazon Simple Storage Service (Amazon S3)
 resource.
        """
        self.polly_client = polly_client
        self.s3_resource = s3_resource
        self.voice_metadata = None


    def do_synthesis_task(
        self,
        text,
        engine,
        voice,
        audio_format,
```

```
        s3_bucket,
        lang_code=None,
        include_visemes=False,
        wait_callback=None,
    ):
        """
        Start an asynchronous task to synthesize speech or speech marks, wait for
        the task to complete, retrieve the output from Amazon S3, and return the
        data.

        An asynchronous task is required when the text is too long for near-real
time
        synthesis.

        :param text: The text to synthesize.
        :param engine: The kind of engine used. Can be standard or neural.
        :param voice: The ID of the voice to use.
        :param audio_format: The audio format to return for synthesized speech.
When
                             speech marks are synthesized, the output format is
JSON.
        :param s3_bucket: The name of an existing Amazon S3 bucket that you have
                          write access to. Synthesis output is written to this
bucket.
        :param lang_code: The language code of the voice to use. This has an
effect
                          only when a bilingual voice is selected.
        :param include_visemes: When True, a second request is made to Amazon
Polly
                                to synthesize a list of visemes, using the
specified
                                text and voice. A viseme represents the visual
position
                                of the face and mouth when saying part of a word.
        :param wait_callback: A callback function that is called periodically
during
                              task processing, to give the caller an opportunity
to
                              take action, such as to display status.
        :return: The audio stream that contains the synthesized speech and a list
                 of visemes that are associated with the speech audio.
        """
        try:
            kwargs = {
```

```
                "Engine": engine,
                "OutputFormat": audio_format,
                "OutputS3BucketName": s3_bucket,
                "Text": text,
                "VoiceId": voice,
            }
            if lang_code is not None:
                kwargs["LanguageCode"] = lang_code
            response = self.polly_client.start_speech_synthesis_task(**kwargs)
            speech_task = response["SynthesisTask"]
            logger.info("Started speech synthesis task %s.",
speech_task["TaskId"])

            viseme_task = None
            if include_visemes:
                kwargs["OutputFormat"] = "json"
                kwargs["SpeechMarkTypes"] = ["viseme"]
                response =
self.polly_client.start_speech_synthesis_task(**kwargs)
                viseme_task = response["SynthesisTask"]
                logger.info("Started viseme synthesis task %s.",
viseme_task["TaskId"])
        except ClientError:
            logger.exception("Couldn't start synthesis task.")
            raise
        else:
            bucket = self.s3_resource.Bucket(s3_bucket)
            audio_stream = self._wait_for_task(
                10, speech_task["TaskId"], "speech", wait_callback, bucket
            )

            visemes = None
            if include_visemes:
                viseme_data = self._wait_for_task(
                    10, viseme_task["TaskId"], "viseme", wait_callback, bucket
                )
                visemes = [
                    json.loads(v) for v in viseme_data.read().decode().split() if
v
                ]

            return audio_stream, visemes
```

- Einzelheiten zur API finden Sie StartSpeechSynthesisTaskin AWS SDK for Python (Boto3) API Reference.

SAP ABAP

SDK für SAP ABAP

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
TRY.
    " Only pass optional parameters if they have values
    IF iv_lang_code IS NOT INITIAL AND iv_s3_key_prefix IS NOT INITIAL.
      oo_result = lo_ply->startspeechsynthesistask(
        iv_engine = iv_engine
        iv_outputformat = iv_audio_format
        iv_outputs3bucketname = iv_s3_bucket
        iv_outputs3keyprefix = iv_s3_key_prefix
        iv_text = iv_text
        iv_voiceid = iv_voice_id
        iv_languagecode = iv_lang_code ).
    ELSEIF iv_lang_code IS NOT INITIAL.
      oo_result = lo_ply->startspeechsynthesistask(
        iv_engine = iv_engine
        iv_outputformat = iv_audio_format
        iv_outputs3bucketname = iv_s3_bucket
        iv_text = iv_text
        iv_voiceid = iv_voice_id
        iv_languagecode = iv_lang_code ).
    ELSEIF iv_s3_key_prefix IS NOT INITIAL.
      oo_result = lo_ply->startspeechsynthesistask(
        iv_engine = iv_engine
        iv_outputformat = iv_audio_format
        iv_outputs3bucketname = iv_s3_bucket
        iv_outputs3keyprefix = iv_s3_key_prefix
        iv_text = iv_text
```

```
            iv_voiceid = iv_voice_id ).
        ELSE.
          oo_result = lo_ply->startspeechsynthesistask(
            iv_engine = iv_engine
            iv_outputformat = iv_audio_format
            iv_outputs3bucketname = iv_s3_bucket
            iv_text = iv_text
            iv_voiceid = iv_voice_id ).
        ENDIF.
        MESSAGE 'Speech synthesis task started.' TYPE 'I'.
      CATCH /aws1/cx_plyinvalids3bucketex.
        MESSAGE 'Invalid S3 bucket.' TYPE 'E'.
      CATCH /aws1/cx_plyinvalidssmlex.
        MESSAGE 'Invalid SSML.' TYPE 'E'.
      CATCH /aws1/cx_plylexiconnotfoundex.
        MESSAGE 'Lexicon not found.' TYPE 'E'.
      CATCH /aws1/cx_plyservicefailureex.
        MESSAGE 'Service failure occurred.' TYPE 'E'.
      CATCH /aws1/cx_plytextlengthexcdex.
        MESSAGE 'Text length exceeded maximum.' TYPE 'E'.
    ENDTRY.
```

- Einzelheiten zur API finden Sie StartSpeechSynthesisTaskin der API-Referenz zum AWS SDK für SAP ABAP.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterAmazon Polly mit einem AWS SDK verwenden. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Verwenden Sie **`SynthesizeSpeech`**mit einem AWS SDK

Die folgenden Code-Beispiele zeigen, wie `SynthesizeSpeech` verwendet wird.

Beispiele für Aktionen sind Codeauszüge aus größeren Programmen und müssen im Kontext ausgeführt werden. Im folgenden Codebeispiel können Sie diese Aktion im Kontext sehen:

- Erste Schritte mit der Text-zu-Sprache-Synthese

.NET

SDK für .NET

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
    using System;
    using System.IO;
    using System.Threading.Tasks;
    using Amazon.Polly;
    using Amazon.Polly.Model;

    public class SynthesizeSpeech
    {
        public static async Task Main()
        {
            string outputFileName = "speech.mp3";
            string text = "Twas brillig, and the slithy toves did gyre and gimbol
in the wabe";

            var client = new AmazonPollyClient();
            var response = await PollySynthesizeSpeech(client, text);

            WriteSpeechToStream(response.AudioStream, outputFileName);
        }

        /// <summary>
        /// Calls the Amazon Polly SynthesizeSpeechAsync method to convert text
        /// to speech.
        /// </summary>
        /// <param name="client">The Amazon Polly client object used to connect
        /// to the Amazon Polly service.</param>
        /// <param name="text">The text to convert to speech.</param>
        /// <returns>A SynthesizeSpeechResponse object that includes an
AudioStream
        /// object with the converted text.</returns>
        private static async Task<SynthesizeSpeechResponse>
PollySynthesizeSpeech(IAmazonPolly client, string text)
```

```
        {
            var synthesizeSpeechRequest = new SynthesizeSpeechRequest()
            {
                OutputFormat = OutputFormat.Mp3,
                VoiceId = VoiceId.Joanna,
                Text = text,
            };

            var synthesizeSpeechResponse =
                await client.SynthesizeSpeechAsync(synthesizeSpeechRequest);

            return synthesizeSpeechResponse;
        }

        /// <summary>
        /// Writes the AudioStream returned from the call to
        /// SynthesizeSpeechAsync to a file in MP3 format.
        /// </summary>
        /// <param name="audioStream">The AudioStream returned from the
        /// call to the SynthesizeSpeechAsync method.</param>
        /// <param name="outputFileName">The full path to the file in which to
        /// save the audio stream.</param>
        private static void WriteSpeechToStream(Stream audioStream, string
outputFileName)
        {
            var outputStream = new FileStream(
                outputFileName,
                FileMode.Create,
                FileAccess.Write);
            byte[] buffer = new byte[2 * 1024];
            int readBytes;

            while ((readBytes = audioStream.Read(buffer, 0, 2 * 1024)) > 0)
            {
                outputStream.Write(buffer, 0, readBytes);
            }

            // Flushes the buffer to avoid losing the last second or so of
            // the synthesized text.
            outputStream.Flush();
            Console.WriteLine($"Saved {outputFileName} to disk.");
        }
    }
```

Synthetisieren Sie Sprache aus Text mithilfe von Sprachmarken mit Amazon Polly mithilfe eines AWS SDK.

```
    using System;
    using System.Collections.Generic;
    using System.IO;
    using System.Threading.Tasks;
    using Amazon.Polly;
    using Amazon.Polly.Model;

    public class SynthesizeSpeechMarks
    {
        public static async Task Main()
        {
            var client = new AmazonPollyClient();
            string outputFileName = "speechMarks.json";

            var synthesizeSpeechRequest = new SynthesizeSpeechRequest()
            {
                OutputFormat = OutputFormat.Json,
                SpeechMarkTypes = new List<string>
                {
                    SpeechMarkType.Viseme,
                    SpeechMarkType.Word,
                },
                VoiceId = VoiceId.Joanna,
                Text = "This is a sample text to be synthesized.",
            };

            try
            {
                using (var outputStream = new FileStream(outputFileName,
FileMode.Create, FileAccess.Write))
                {
                    var synthesizeSpeechResponse = await
client.SynthesizeSpeechAsync(synthesizeSpeechRequest);
                    var buffer = new byte[2 * 1024];
                    int readBytes;

                    var inputStream = synthesizeSpeechResponse.AudioStream;
```

```
                    while ((readBytes = inputStream.Read(buffer, 0, 2 * 1024)) >
0)
                    {
                        outputStream.Write(buffer, 0, readBytes);
                    }
                }
            }
            catch (Exception ex)
            {
                Console.WriteLine($"Error: {ex.Message}");
            }
        }
    }
```

- Einzelheiten zur API finden Sie unter SynthesizeSpeech AWS SDK für .NETAPI-Referenz.

Java

SDK für Java 2.x

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
import javazoom.jl.decoder.JavaLayerException;
import software.amazon.awssdk.core.ResponseInputStream;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.polly.PollyClient;
import software.amazon.awssdk.services.polly.model.DescribeVoicesRequest;
import software.amazon.awssdk.services.polly.model.Voice;
import software.amazon.awssdk.services.polly.model.DescribeVoicesResponse;
import software.amazon.awssdk.services.polly.model.OutputFormat;
import software.amazon.awssdk.services.polly.model.PollyException;
import software.amazon.awssdk.services.polly.model.SynthesizeSpeechRequest;
import software.amazon.awssdk.services.polly.model.SynthesizeSpeechResponse;
import java.io.IOException;
import java.io.InputStream;
import javazoom.jl.player.advanced.AdvancedPlayer;
```

```
import javazoom.jl.player.advanced.PlaybackEvent;
import javazoom.jl.player.advanced.PlaybackListener;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class PollyDemo {
    private static final String SAMPLE = "Congratulations. You have successfully
 built this working demo " +
            " of Amazon Polly in Java Version 2. 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 static void main(String args[]) {
        PollyClient polly = PollyClient.builder()
                .region(Region.US_WEST_2)
                .build();

        talkPolly(polly);
        polly.close();
    }

    public static void talkPolly(PollyClient polly) {
        try {
            DescribeVoicesRequest describeVoiceRequest =
 DescribeVoicesRequest.builder()
                    .engine("standard")
                    .build();

            DescribeVoicesResponse describeVoicesResult =
 polly.describeVoices(describeVoiceRequest);
            Voice voice = describeVoicesResult.voices().stream()
                    .filter(v -> v.name().equals("Joanna"))
                    .findFirst()
                    .orElseThrow(() -> new RuntimeException("Voice not found"));
```

```
            InputStream stream = synthesize(polly, SAMPLE, voice,
 OutputFormat.MP3);
            AdvancedPlayer player = new AdvancedPlayer(stream,

 javazoom.jl.player.FactoryRegistry.systemRegistry().createAudioDevice());
            player.setPlayBackListener(new PlaybackListener() {
                public void playbackStarted(PlaybackEvent evt) {
                    System.out.println("Playback started");
                    System.out.println(SAMPLE);
                }

                public void playbackFinished(PlaybackEvent evt) {
                    System.out.println("Playback finished");
                }
            });

            // play it!
            player.play();

        } catch (PollyException | JavaLayerException | IOException e) {
            System.err.println(e.getMessage());
            System.exit(1);
        }
    }

    public static InputStream synthesize(PollyClient polly, String text, Voice
 voice, OutputFormat format)
            throws IOException {
        SynthesizeSpeechRequest synthReq = SynthesizeSpeechRequest.builder()
                .text(text)
                .voiceId(voice.id())
                .outputFormat(format)
                .build();

        ResponseInputStream<SynthesizeSpeechResponse> synthRes =
 polly.synthesizeSpeech(synthReq);
        return synthRes;
    }
}
```

- Einzelheiten zur API finden Sie SynthesizeSpeechin der AWS SDK for Java 2.x API-Referenz.

Python

SDK für Python (Boto3)

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
class PollyWrapper:
    """Encapsulates Amazon Polly functions."""

    def __init__(self, polly_client, s3_resource):
        """
        :param polly_client: A Boto3 Amazon Polly client.
        :param s3_resource: A Boto3 Amazon Simple Storage Service (Amazon S3)
 resource.
        """
        self.polly_client = polly_client
        self.s3_resource = s3_resource
        self.voice_metadata = None


    def synthesize(
        self, text, engine, voice, audio_format, lang_code=None,
 include_visemes=False
    ):
        """
        Synthesizes speech or speech marks from text, using the specified voice.

        :param text: The text to synthesize.
        :param engine: The kind of engine used. Can be standard or neural.
        :param voice: The ID of the voice to use.
        :param audio_format: The audio format to return for synthesized speech.
 When
                             speech marks are synthesized, the output format is
 JSON.
        :param lang_code: The language code of the voice to use. This has an
 effect
                          only when a bilingual voice is selected.
```

```
        :param include_visemes: When True, a second request is made to Amazon
Polly
                                to synthesize a list of visemes, using the
specified
                                text and voice. A viseme represents the visual
position
                                of the face and mouth when saying part of a word.
        :return: The audio stream that contains the synthesized speech and a list
                 of visemes that are associated with the speech audio.
        """
        try:
            kwargs = {
                "Engine": engine,
                "OutputFormat": audio_format,
                "Text": text,
                "VoiceId": voice,
            }
            if lang_code is not None:
                kwargs["LanguageCode"] = lang_code
            response = self.polly_client.synthesize_speech(**kwargs)
            audio_stream = response["AudioStream"]
            logger.info("Got audio stream spoken by %s.", voice)
            visemes = None
            if include_visemes:
                kwargs["OutputFormat"] = "json"
                kwargs["SpeechMarkTypes"] = ["viseme"]
                response = self.polly_client.synthesize_speech(**kwargs)
                visemes = [
                    json.loads(v)
                    for v in response["AudioStream"].read().decode().split()
                    if v
                ]
                logger.info("Got %s visemes.", len(visemes))
        except ClientError:
            logger.exception("Couldn't get audio stream.")
            raise
        else:
            return audio_stream, visemes
```

- Einzelheiten zur API finden Sie SynthesizeSpeechin AWS SDK for Python (Boto3) API Reference.

Ruby

SDK für Ruby

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
require 'aws-sdk-polly' # In v2: require 'aws-sdk'

begin
  # Get the filename from the command line
  if ARGV.empty?
    puts 'You must supply a filename'
    exit 1
  end

  filename = ARGV[0]

  # Open file and get the contents as a string
  if File.exist?(filename)
    contents = IO.read(filename)
  else
    puts "No such file: #{filename}"
    exit 1
  end

  # Create an Amazon Polly client using
  # credentials from the shared credentials file ~/.aws/credentials
  # and the configuration (region) from the shared configuration file ~/.aws/
config
  polly = Aws::Polly::Client.new

  resp = polly.synthesize_speech({
                                   output_format: 'mp3',
                                   text: contents,
                                   voice_id: 'Joanna'
                                 })
```

```
  # Save output
  # Get just the file name
  #  abc/xyz.txt -> xyx.txt
  name = File.basename(filename)

  # Split up name so we get just the xyz part
  parts = name.split('.')
  first_part = parts[0]
  mp3_file = "#{first_part}.mp3"

  IO.copy_stream(resp.audio_stream, mp3_file)

  puts "Wrote MP3 content to: #{mp3_file}"
rescue StandardError => e
  puts 'Got error:'
  puts 'Error message:'
  puts e.message
end
```

- Einzelheiten zur API finden Sie SynthesizeSpeechin der AWS SDK für Ruby API-Referenz.

Rust

SDK für Rust

> **Note**
>
> Es gibt noch mehr dazu GitHub. Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
async fn synthesize(client: &Client, filename: &str) -> Result<(), Error> {
    let content = fs::read_to_string(filename);

    let resp = client
        .synthesize_speech()
        .output_format(OutputFormat::Mp3)
        .text(content.unwrap())
        .voice_id(VoiceId::Joanna)
        .send()
        .await?;
```

```
    // Get MP3 data from response and save it
    let mut blob = resp
        .audio_stream
        .collect()
        .await
        .expect("failed to read data");

    let parts: Vec<&str> = filename.split('.').collect();
    let out_file = format!("{}{}", String::from(parts[0]), ".mp3");

    let mut file = tokio::fs::File::create(out_file)
        .await
        .expect("failed to create file");

    file.write_all_buf(&mut blob)
        .await
        .expect("failed to write to file");

    Ok(())
}
```

- Einzelheiten zur API finden Sie SynthesizeSpeechin der API-Referenz zum AWS SDK für Rust.

SAP ABAP

SDK für SAP ABAP

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie das AWS -Code-Beispiel- einrichten und ausführen.

```
    TRY.
        " Only pass optional language code if it has a value
        IF iv_lang_code IS NOT INITIAL.
          oo_result = lo_ply->synthesizespeech(
            iv_engine = iv_engine
```

```
            iv_outputformat = iv_output_fmt
            iv_text = iv_text
            iv_voiceid = iv_voice_id
            iv_languagecode = iv_lang_code ).
        ELSE.
          oo_result = lo_ply->synthesizespeech(
            iv_engine = iv_engine
            iv_outputformat = iv_output_fmt
            iv_text = iv_text
            iv_voiceid = iv_voice_id ).
        ENDIF.
        MESSAGE 'Speech synthesized successfully.' TYPE 'I'.
      CATCH /aws1/cx_plyinvalidssmlex.
        MESSAGE 'Invalid SSML.' TYPE 'E'.
      CATCH /aws1/cx_plylexiconnotfoundex.
        MESSAGE 'Lexicon not found.' TYPE 'E'.
      CATCH /aws1/cx_plyservicefailureex.
        MESSAGE 'Service failure occurred.' TYPE 'E'.
      CATCH /aws1/cx_plytextlengthexcdex.
        MESSAGE 'Text length exceeded maximum.' TYPE 'E'.
    ENDTRY.
```

- Einzelheiten zur API finden Sie SynthesizeSpeechin der API-Referenz zum AWS SDK für SAP ABAP.

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterAmazon Polly mit einem AWS SDK verwenden. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Szenarien für die Verwendung von Amazon Polly AWS SDKs

Die folgenden Codebeispiele zeigen Ihnen, wie Sie allgemeine Szenarien in Amazon Polly mit AWS SDKs implementieren. Diese Szenarien zeigen Ihnen, wie Sie bestimmte Aufgaben durch den Aufruf mehrerer Funktionen innerhalb von Amazon Polly oder in Kombination mit anderen AWS-Services ausführen können. Jedes Szenario enthält einen Link zum vollständigen Quell-Code, wo Sie Anleitungen zum Einrichten und Ausführen des Codes finden.

Szenarien zielen auf eine mittlere Erfahrungsebene ab, um Ihnen zu helfen, Service-Aktionen im Kontext zu verstehen.

Beispiele

- Konvertieren Sie Text in Sprache und zurück in Text mit einem AWS SDK
- Erstellen Sie eine Lippensynchronisationsanwendung mit Amazon Polly mithilfe eines AWS SDK
- Erstellen einer Anwendung, die Kundenfeedback analysiert und Audio generiert
- Erste Schritte mit der Text-zu-Sprache-Synthese

Konvertieren Sie Text in Sprache und zurück in Text mit einem AWS SDK

Wie das aussehen kann, sehen Sie am nachfolgenden Beispielcode:

- Verwenden Sie Amazon Polly, um eine Klartext-Eingabedatei (UTF-8) zu einer Audiodatei zu synthetisieren.
- Laden Sie die Audiodatei in einen Amazon-S3-Bucket hoch.
- Konvertieren Sie die Audiodatei mit Amazon Transcribe in Text.
- Zeigen Sie den Text an.

Rust

SDK für Rust

Verwenden Sie Amazon Polly, um eine Nur-Text-Eingabedatei (UTF-8) in eine Audiodatei zu synthetisieren, die Audiodatei in einen Amazon S3 S3-Bucket hochzuladen, Amazon Transcribe zu verwenden, um diese Audiodatei in Text zu konvertieren und den Text anzuzeigen.

Den vollständigen Quellcode und Anweisungen zur Einrichtung und Ausführung finden Sie im vollständigen Beispiel unter. GitHub

In diesem Beispiel verwendete Dienste

- Amazon Polly
- Amazon S3
- Amazon Transcribe

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterAmazon Polly mit einem AWS SDK verwenden. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Erstellen Sie eine Lippensynchronisationsanwendung mit Amazon Polly mithilfe eines AWS SDK

Das folgende Codebeispiel zeigt, wie Sie eine Lippensynchronisationsanwendung mit Amazon Polly erstellen.

Python

SDK für Python (Boto3)

Zeigt, wie Amazon Polly und Tkinter verwendet werden, um eine Lippensynchronisierungsanwendung zu erstellen, die ein animiertes sprechendes Gesicht zusammen mit der von Amazon Polly synthetisierten Sprache anzeigt. Lip-sync wird erreicht, indem bei Amazon Polly eine Liste von Visemen angefordert wird, die mit der synthetisierten Sprache übereinstimmen.

- Erhalten Sie Sprachmetadaten von Amazon Polly und zeigen Sie diese in einer Tkinter-Anwendung an.
- Erhalten Sie synthetisierte Sprachaufnahmen und passende Viseme-Sprachmarkierungen von Amazon Polly.
- Spielen Sie die Audioaufnahme mit synchronisierten Mundbewegungen in einem animierten Gesicht ab.
- Asynchrone Syntheseaufgaben für lange Texte übermitteln und die Ausgabe aus einem Bucket von Amazon Simple Storage Service (Amazon S3) abrufen.

Den vollständigen Quellcode und Anweisungen zur Einrichtung und Ausführung finden Sie im vollständigen Beispiel unter. GitHub

In diesem Beispiel verwendete Dienste

- Amazon Polly

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterAmazon Polly mit einem AWS SDK verwenden. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Erstellen einer Anwendung, die Kundenfeedback analysiert und Audio generiert

Die folgenden Codebeispiele zeigen, wie Sie eine Anwendung erstellen, die Kundenkommentarkarten analysiert, sie aus ihrer Originalsprache übersetzt, ihre Stimmung ermittelt und aus dem übersetzten Text eine Audiodatei generiert.

.NET

SDK für .NET

Diese Beispielanwendung analysiert und speichert Kundenfeedback-Karten. Sie ist auf die Anforderungen eines fiktiven Hotels in New York City zugeschnitten. Das Hotel erhält Feedback von Gästen in Form von physischen Kommentarkarten in verschiedenen Sprachen. Dieses Feedback wird über einen Webclient in die App hochgeladen. Nachdem ein Bild einer Kommentarkarte hochgeladen wurde, werden folgende Schritte ausgeführt:

- Der Text wird mithilfe von Amazon Textract aus dem Bild extrahiert.
- Amazon Comprehend ermittelt die Stimmung und die Sprache des extrahierten Textes.
- Der extrahierte Text wird mithilfe von Amazon Translate ins Englische übersetzt.
- Amazon Polly generiert auf der Grundlage des extrahierten Texts eine Audiodatei.

Die vollständige App kann mithilfe des AWS CDK bereitgestellt werden. Den Quellcode und Anweisungen zur Bereitstellung finden Sie im Projekt unter GitHub.

In diesem Beispiel verwendete Dienste

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

Java

SDK für Java 2.x

Diese Beispielanwendung analysiert und speichert Kundenfeedback-Karten. Sie ist auf die Anforderungen eines fiktiven Hotels in New York City zugeschnitten. Das Hotel erhält

Feedback von Gästen in Form von physischen Kommentarkarten in verschiedenen Sprachen. Dieses Feedback wird über einen Webclient in die App hochgeladen. Nachdem ein Bild einer Kommentarkarte hochgeladen wurde, werden folgende Schritte ausgeführt:

- Der Text wird mithilfe von Amazon Textract aus dem Bild extrahiert.
- Amazon Comprehend ermittelt die Stimmung und die Sprache des extrahierten Textes.
- Der extrahierte Text wird mithilfe von Amazon Translate ins Englische übersetzt.
- Amazon Polly generiert auf der Grundlage des extrahierten Texts eine Audiodatei.

Die vollständige App kann mithilfe des AWS CDK bereitgestellt werden. Den Quellcode und Anweisungen zur Bereitstellung finden Sie im Projekt unter GitHub.

In diesem Beispiel verwendete Dienste

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

JavaScript

SDK für JavaScript (v3)

Diese Beispielanwendung analysiert und speichert Kundenfeedback-Karten. Sie ist auf die Anforderungen eines fiktiven Hotels in New York City zugeschnitten. Das Hotel erhält Feedback von Gästen in Form von physischen Kommentarkarten in verschiedenen Sprachen. Dieses Feedback wird über einen Webclient in die App hochgeladen. Nachdem ein Bild einer Kommentarkarte hochgeladen wurde, werden folgende Schritte ausgeführt:

- Der Text wird mithilfe von Amazon Textract aus dem Bild extrahiert.
- Amazon Comprehend ermittelt die Stimmung und die Sprache des extrahierten Textes.
- Der extrahierte Text wird mithilfe von Amazon Translate ins Englische übersetzt.
- Amazon Polly generiert auf der Grundlage des extrahierten Texts eine Audiodatei.

Die vollständige App kann mithilfe des AWS CDK bereitgestellt werden. Den Quellcode und Anweisungen zur Bereitstellung finden Sie im Projekt unter GitHub. Die folgenden Auszüge zeigen, wie der innerhalb von Lambda-Funktionen verwendet AWS SDK für JavaScript wird.

```
import {
  ComprehendClient,
  DetectDominantLanguageCommand,
  DetectSentimentCommand,
} from "@aws-sdk/client-comprehend";

/**
 * Determine the language and sentiment of the extracted text.
 *
 * @param {{ source_text: string}} extractTextOutput
 */
export const handler = async (extractTextOutput) => {
  const comprehendClient = new ComprehendClient({});

  const detectDominantLanguageCommand = new DetectDominantLanguageCommand({
    Text: extractTextOutput.source_text,
  });

  // The source language is required for sentiment analysis and
  // translation in the next step.
  const { Languages } = await comprehendClient.send(
    detectDominantLanguageCommand,
  );

  const languageCode = Languages[0].LanguageCode;

  const detectSentimentCommand = new DetectSentimentCommand({
    Text: extractTextOutput.source_text,
    LanguageCode: languageCode,
  });

  const { Sentiment } = await comprehendClient.send(detectSentimentCommand);

  return {
    sentiment: Sentiment,
    language_code: languageCode,
  };
};
```

```
import {
  DetectDocumentTextCommand,
  TextractClient,
} from "@aws-sdk/client-textract";
```

```
/**
 * Fetch the S3 object from the event and analyze it using Amazon Textract.
 *
 * @param {import("@types/aws-lambda").EventBridgeEvent<"Object Created">}
 eventBridgeS3Event
 */
export const handler = async (eventBridgeS3Event) => {
  const textractClient = new TextractClient();

  const detectDocumentTextCommand = new DetectDocumentTextCommand({
    Document: {
      S3Object: {
        Bucket: eventBridgeS3Event.bucket,
        Name: eventBridgeS3Event.object,
      },
    },
  });

  // Textract returns a list of blocks. A block can be a line, a page, word, etc.
  // Each block also contains geometry of the detected text.
  // For more information on the Block type, see https://docs.aws.amazon.com/
textract/latest/dg/API_Block.html.
  const { Blocks } = await textractClient.send(detectDocumentTextCommand);

  // For the purpose of this example, we are only interested in words.
  const extractedWords = Blocks.filter((b) => b.BlockType === "WORD").map(
    (b) => b.Text,
  );

  return extractedWords.join(" ");
};
```

```
import { PollyClient, SynthesizeSpeechCommand } from "@aws-sdk/client-polly";
import { S3Client } from "@aws-sdk/client-s3";
import { Upload } from "@aws-sdk/lib-storage";

/**
 * Synthesize an audio file from text.
 *
 * @param {{ bucket: string, translated_text: string, object: string}}
 sourceDestinationConfig
 */
```

```
export const handler = async (sourceDestinationConfig) => {
  const pollyClient = new PollyClient({});

  const synthesizeSpeechCommand = new SynthesizeSpeechCommand({
    Engine: "neural",
    Text: sourceDestinationConfig.translated_text,
    VoiceId: "Ruth",
    OutputFormat: "mp3",
  });

  const { AudioStream } = await pollyClient.send(synthesizeSpeechCommand);

  const audioKey = `${sourceDestinationConfig.object}.mp3`;

  // Store the audio file in S3.
  const s3Client = new S3Client();
  const upload = new Upload({
    client: s3Client,
    params: {
      Bucket: sourceDestinationConfig.bucket,
      Key: audioKey,
      Body: AudioStream,
      ContentType: "audio/mp3",
    },
  });

  await upload.done();
  return audioKey;
};
```

```
import {
  TranslateClient,
  TranslateTextCommand,
} from "@aws-sdk/client-translate";

/**
 * Translate the extracted text to English.
 *
 * @param {{ extracted_text: string, source_language_code: string}}
 textAndSourceLanguage
 */
export const handler = async (textAndSourceLanguage) => {
  const translateClient = new TranslateClient({});
```

```
  const translateCommand = new TranslateTextCommand({
    SourceLanguageCode: textAndSourceLanguage.source_language_code,
    TargetLanguageCode: "en",
    Text: textAndSourceLanguage.extracted_text,
  });

  const { TranslatedText } = await translateClient.send(translateCommand);

  return { translated_text: TranslatedText };
};
```

In diesem Beispiel verwendete Dienste

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

Ruby

SDK für Ruby

Diese Beispielanwendung analysiert und speichert Kundenfeedback-Karten. Sie ist auf die Anforderungen eines fiktiven Hotels in New York City zugeschnitten. Das Hotel erhält Feedback von Gästen in Form von physischen Kommentarkarten in verschiedenen Sprachen. Dieses Feedback wird über einen Webclient in die App hochgeladen. Nachdem ein Bild einer Kommentarkarte hochgeladen wurde, werden folgende Schritte ausgeführt:

- Der Text wird mithilfe von Amazon Textract aus dem Bild extrahiert.
- Amazon Comprehend ermittelt die Stimmung und die Sprache des extrahierten Textes.
- Der extrahierte Text wird mithilfe von Amazon Translate ins Englische übersetzt.
- Amazon Polly generiert auf der Grundlage des extrahierten Texts eine Audiodatei.

Die vollständige App kann mithilfe des AWS CDK bereitgestellt werden. Quellcode und Anweisungen zur Bereitstellung finden Sie im Projekt unter. GitHub

In diesem Beispiel verwendete Dienste

- Amazon Comprehend

- Lambda
- Amazon Polly
- Amazon Textract
- Amazon Translate

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterAmazon Polly mit einem AWS SDK verwenden. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Erste Schritte mit der Text-zu-Sprache-Synthese

Wie das aussehen kann, sehen Sie am nachfolgenden Beispielcode:

- Bereinigen von Ressourcen

Bash

AWS CLI mit Bash-Skript

> **Note**
>
> Es gibt noch mehr dazu. GitHub Hier finden Sie das vollständige Beispiel und erfahren, wie Sie es im Tutorials und Beispiele-Repository für Entwickler einrichten und ausführen.

```
#!/bin/bash

# Amazon Polly Getting Started Script
# This script demonstrates how to use Amazon Polly with the AWS CLI

set -euo pipefail

# Set up logging
LOG_FILE="polly-tutorial.log"
SCRIPT_DIR="$(cd "$(dirname "${BASH_SOURCE[0]}")" && pwd)"
WORK_DIR=$(mktemp -d)
trap 'cleanup_temp' EXIT
```

```
cleanup_temp() {
    rm -rf "$WORK_DIR"
}

echo "Starting Amazon Polly tutorial at $(date)" > "$LOG_FILE"

# Function to log commands and their output
log_cmd() {
    echo "Running: $1" | tee -a "$LOG_FILE"
    # Use bash array to safely handle arguments
    bash -c "$1" 2>&1 | tee -a "$LOG_FILE" || return $?
}

# Function to check for errors
check_error() {
    if echo "$1" | grep -iq "error"; then
        echo "ERROR detected in output. Exiting script." | tee -a "$LOG_FILE"
        echo "$1" | tee -a "$LOG_FILE"
        return 1
    fi
    return 0
}

# Function to handle errors and cleanup
handle_error() {
    local line_number=$1
    echo "Error occurred at line $line_number. Attempting cleanup..." | tee -a
 "$LOG_FILE"
    cleanup
    exit 1
}

# Function to clean up resources
cleanup() {
    echo "" | tee -a "$LOG_FILE"
    echo "============================================================" | tee -a
 "$LOG_FILE"
    echo "CLEANUP PROCESS" | tee -a "$LOG_FILE"
    echo "============================================================" | tee -a
 "$LOG_FILE"

    # Delete lexicon if it exists
    if [[ -n "${LEXICON_NAME:-}" ]]; then
        echo "Deleting lexicon: $LEXICON_NAME" | tee -a "$LOG_FILE"
```

```
        if aws polly delete-lexicon --name "$LEXICON_NAME" 2>&1 | tee -a
 "$LOG_FILE"; then
            echo "Lexicon deleted successfully." | tee -a "$LOG_FILE"
        else
            echo "Warning: Failed to delete lexicon." | tee -a "$LOG_FILE"
        fi
    fi

    # Remove audio files
    for file in output.mp3 ssml-output.mp3 lexicon-output.mp3 example.pls; do
        if [[ -f "$file" ]]; then
            rm -f "$file"
            echo "Removed $file" | tee -a "$LOG_FILE"
        fi
    done

    echo "Cleanup complete." | tee -a "$LOG_FILE"
}

# Trap errors with line number
trap 'handle_error ${LINENO}' ERR

# Verify AWS CLI is available
if ! command -v aws &> /dev/null; then
    echo "AWS CLI is not installed. Please install it first." | tee -a
 "$LOG_FILE"
    exit 1
fi

# Verify AWS credentials are configured
if ! aws sts get-caller-identity &> /dev/null; then
    echo "AWS credentials are not configured. Please configure them first." | tee
 -a "$LOG_FILE"
    exit 1
fi

# Step 1: Verify Amazon Polly is available
echo "Step 1: Verifying Amazon Polly availability" | tee -a "$LOG_FILE"
if aws polly describe-voices --query 'Voices[0].Name' --output text &> /dev/null;
 then
    echo "Amazon Polly is available. Proceeding with tutorial." | tee -a
 "$LOG_FILE"
else
```

```
    echo "Amazon Polly is not available in your AWS CLI installation or region."
 | tee -a "$LOG_FILE"
    echo "Please update your AWS CLI to the latest version or check your region."
 | tee -a "$LOG_FILE"
    exit 1
fi

# Step 2: List available voices
echo "" | tee -a "$LOG_FILE"
echo "Step 2: Listing available voices" | tee -a "$LOG_FILE"
log_cmd "aws polly describe-voices --language-code en-US --output text --query
 'Voices[0:3].[Id, LanguageCode, Gender]'" || true

# Step 3: Basic text-to-speech conversion
echo "" | tee -a "$LOG_FILE"
echo "Step 3: Converting text to speech" | tee -a "$LOG_FILE"
OUTPUT_FILE="${WORK_DIR}/output.mp3"
POLLY_TEXT="Hello, welcome to Amazon Polly. This is a sample text to speech
 conversion."
log_cmd "aws polly synthesize-speech --output-format mp3 --voice-id Joanna --text
 '$POLLY_TEXT' '$OUTPUT_FILE'" || true

if [[ -f "$OUTPUT_FILE" ]]; then
    echo "Successfully created output.mp3 file." | tee -a "$LOG_FILE"
    echo "You can play this file with your preferred audio player." | tee -a
 "$LOG_FILE"
    cp "$OUTPUT_FILE" output.mp3
else
    echo "Failed to create output.mp3 file." | tee -a "$LOG_FILE"
    exit 1
fi

# Step 4: Using SSML for enhanced speech
echo "" | tee -a "$LOG_FILE"
echo "Step 4: Using SSML for enhanced speech" | tee -a "$LOG_FILE"
SSML_OUTPUT="${WORK_DIR}/ssml-output.mp3"
SSML_TEXT='<speak>Hello! <break time="1s"/> This is a sample of <emphasis>SSML
 enhanced speech</emphasis>.</speak>'
log_cmd "aws polly synthesize-speech --output-format mp3 --voice-id Matthew --
text-type ssml --text '$SSML_TEXT' '$SSML_OUTPUT'" || true

if [[ -f "$SSML_OUTPUT" ]]; then
    echo "Successfully created ssml-output.mp3 file." | tee -a "$LOG_FILE"
```

```
    echo "You can play this file with your preferred audio player." | tee -a
 "$LOG_FILE"
    cp "$SSML_OUTPUT" ssml-output.mp3
else
    echo "Failed to create ssml-output.mp3 file." | tee -a "$LOG_FILE"
    exit 1
fi

# Step 5: Working with lexicons
echo "" | tee -a "$LOG_FILE"
echo "Step 5: Working with lexicons" | tee -a "$LOG_FILE"

# Generate a random identifier for the lexicon (max 20 chars, alphanumeric only)
LEXICON_NAME="example$(openssl rand -hex 6 | cut -c 1-10)"
echo "Using lexicon name: $LEXICON_NAME" | tee -a "$LOG_FILE"

# Create a lexicon file
echo "Creating lexicon file..." | tee -a "$LOG_FILE"
LEXICON_FILE="${WORK_DIR}/example.pls"
cat > "$LEXICON_FILE" << 'EOF'
<?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>AWS</grapheme>
    <alias>Amazon Web Services</alias>
  </lexeme>
</lexicon>
EOF

# Upload the lexicon
echo "Uploading lexicon..." | tee -a "$LOG_FILE"
LEXICON_ARN=$(aws polly put-lexicon --name "$LEXICON_NAME" --content
 file://"$LEXICON_FILE" --query 'LexiconArn' --output text 2>&1) || true
if [[ -n "$LEXICON_ARN" && "$LEXICON_ARN" != "" && ! "$LEXICON_ARN" =~ error ]];
 then
    echo "Lexicon uploaded with ARN: $LEXICON_ARN" | tee -a "$LOG_FILE"
```

```
    aws polly tag-resource --resource-arn "$LEXICON_ARN" --tags
 Key=project,Value=doc-smith Key=tutorial,Value=amazon-polly-gs 2>&1 | tee -a
 "$LOG_FILE" || true
else
    echo "Lexicon uploaded." | tee -a "$LOG_FILE"
fi

# List available lexicons
echo "Listing available lexicons..." | tee -a "$LOG_FILE"
log_cmd "aws polly list-lexicons --output text --query 'Lexicons[*].[Name]'" ||
 true

# Get details about the lexicon
echo "Getting details about the lexicon..." | tee -a "$LOG_FILE"
log_cmd "aws polly get-lexicon --name '$LEXICON_NAME' --output text --query
 'Lexicon.Name'" || true

# Use the lexicon when synthesizing speech
echo "Using the lexicon for speech synthesis..." | tee -a "$LOG_FILE"
LEXICON_OUTPUT="${WORK_DIR}/lexicon-output.mp3"
LEXICON_TEXT="I work with AWS every day."
log_cmd "aws polly synthesize-speech --output-format mp3 --voice-id Joanna --
lexicon-names '$LEXICON_NAME' --text '$LEXICON_TEXT' '$LEXICON_OUTPUT'" || true

if [[ -f "$LEXICON_OUTPUT" ]]; then
    echo "Successfully created lexicon-output.mp3 file." | tee -a "$LOG_FILE"
    echo "You can play this file with your preferred audio player." | tee -a
 "$LOG_FILE"
    cp "$LEXICON_OUTPUT" lexicon-output.mp3
else
    echo "Failed to create lexicon-output.mp3 file." | tee -a "$LOG_FILE"
    exit 1
fi

# Summary of created resources
echo "" | tee -a "$LOG_FILE"
echo "============================================================" | tee -a
 "$LOG_FILE"
echo "TUTORIAL SUMMARY" | tee -a "$LOG_FILE"
echo "============================================================" | tee -a
 "$LOG_FILE"
echo "Created resources:" | tee -a "$LOG_FILE"
echo "1. Lexicon: $LEXICON_NAME" | tee -a "$LOG_FILE"
echo "2. Audio files:" | tee -a "$LOG_FILE"
```

```
echo "   - output.mp3" | tee -a "$LOG_FILE"
echo "   - ssml-output.mp3" | tee -a "$LOG_FILE"
echo "   - lexicon-output.mp3" | tee -a "$LOG_FILE"
echo "" | tee -a "$LOG_FILE"

# Cleanup with auto-confirmation
echo "" | tee -a "$LOG_FILE"
echo "============================================================" | tee -a
 "$LOG_FILE"
echo "CLEANUP CONFIRMATION" | tee -a "$LOG_FILE"
echo "============================================================" | tee -a
 "$LOG_FILE"
echo "Cleaning up all created resources..." | tee -a "$LOG_FILE"
cleanup

echo "" | tee -a "$LOG_FILE"
echo "Tutorial completed successfully!" | tee -a "$LOG_FILE"
echo "Log file: $LOG_FILE" | tee -a "$LOG_FILE"
```

- Weitere API-Informationen finden Sie in den folgenden Themen der AWS CLI - Befehlsreferenz.
  - DeleteLexicon
  - DescribeVoices
  - GetLexicon
  - Hilfe
  - ListLexicons
  - PutLexicon
  - SynthesizeSpeech

Eine vollständige Liste der AWS SDK-Entwicklerhandbücher und Codebeispiele finden Sie unterAmazon Polly mit einem AWS SDK verwenden. Dieses Thema enthält auch Informationen zu den ersten Schritten und Details zu früheren SDK-Versionen.

Sicherheit bei Amazon Polly

Cloud-Sicherheit hat AWS höchste Priorität. Als AWS Kunde profitieren Sie von einer Rechenzentrums- und Netzwerkarchitektur, die darauf ausgelegt sind, die Anforderungen der sicherheitssensibelsten Unternehmen zu erfüllen.

Sicherheit ist eine gemeinsame Verantwortung von Ihnen AWS und Ihnen. Das Modell der geteilten Verantwortung beschreibt dies als Sicherheit der Cloud und Sicherheit in der Cloud:

- Sicherheit der Cloud — AWS ist verantwortlich für den Schutz der Infrastruktur, die AWS Dienste in der AWS Cloud ausführt. AWS bietet Ihnen auch Dienste, die Sie sicher nutzen können. Third-party Prüfer testen und verifizieren regelmäßig die Wirksamkeit unserer Sicherheitsmaßnahmen im Rahmen der AWS . Weitere Informationen zu den Compliance-Programmen, die für Amazon Polly gelten, finden Sie unter AWS Services im Umfang nach Compliance-Programm AWS .
- Sicherheit in der Cloud — Ihre Verantwortung richtet sich nach dem AWS Service, den Sie nutzen. Sie sind auch für andere Faktoren verantwortlich, etwa für die Vertraulichkeit Ihrer Daten, die Anforderungen Ihres Unternehmens und die geltenden Gesetze und Vorschriften.

Diese Dokumentation hilft Ihnen zu verstehen, wie Sie das Modell der gemeinsamen Verantwortung bei der Verwendung von Amazon Polly anwenden können. In den folgenden Themen erfahren Sie, wie Sie Amazon Polly konfigurieren, um Ihre Sicherheits- und Compliance-Ziele zu erreichen. Sie erfahren auch, wie Sie andere AWS Dienste nutzen können, mit denen Sie Ihre Amazon Polly Polly-Ressourcen überwachen und sichern können.

Topics

Datenschutz bei Amazon Polly

Amazon Polly entspricht dem Modell der AWS gemeinsamen Verantwortung, das Vorschriften und Richtlinien zum Datenschutz beinhaltet. AWS ist für den Schutz der globalen Infrastruktur verantwortlich, auf der AWS alle Dienste ausgeführt werden. AWS behält die Kontrolle über die auf dieser Infrastruktur gehosteten Daten, einschließlich der Sicherheitskonfigurationskontrollen für den Umgang mit Kundeninhalten und personenbezogenen Daten. AWS Kunden und APN-Partner, die entweder als Datenverantwortliche oder als Datenverarbeiter agieren, sind für alle personenbezogenen Daten verantwortlich, die sie in die AWS Cloud stellen.

Aus Datenschutzgründen empfehlen wir Ihnen, die AWS Kontoanmeldeinformationen zu schützen und einzelne Benutzer mit AWS Identity and Access Management (IAM) einzurichten, sodass jeder Benutzer nur die Berechtigungen erhält, die für die Erfüllung seiner Aufgaben erforderlich sind. Außerdem empfehlen wir, die Daten mit folgenden Methoden schützen:

- Verwenden Sie für jedes Konto die Multi-Faktor-Authentifizierung (MFA).
- Wird verwendet SSL/TLS , um mit AWS Ressourcen zu kommunizieren.
- Richten Sie die API und die Protokollierung von Benutzeraktivitäten mit ein AWS CloudTrail.
- Verwenden Sie AWS Verschlüsselungslösungen zusammen mit allen Standardsicherheitskontrollen innerhalb der AWS Dienste.

Wir empfehlen dringend, in Freitextfeldern wie z. B. im Feld Name keine sensiblen, identifizierenden Informationen wie Kontonummern von Kunden einzugeben. Dies gilt auch, wenn Sie mit Amazon Polly oder anderen AWS Diensten über die Konsole AWS CLI, API oder AWS SDKs arbeiten. Alle Daten, die Sie in Amazon Polly oder andere Dienste eingeben, werden möglicherweise zur Aufnahme in Diagnoseprotokolle aufgenommen. Wenn Sie eine URL für einen externen Server bereitstellen, schließen Sie keine Anmeldeinformationen zur Validierung Ihrer Anforderung an den betreffenden Server in die URL ein.

Weitere Informationen zum Datenschutz enthält der Blog-Beitrag AWS Shared Responsibility Model and GDPR im AWS -Sicherheitsblog.

Verschlüsselung im Ruhezustand

Die Ausgabe Ihrer Amazon Polly Polly-Sprachsynthese kann auf Ihrem eigenen System gespeichert werden. Sie können auch Amazon Polly anrufen und die Datei dann mit einem beliebigen Verschlüsselungsschlüssel Ihrer Wahl verschlüsseln und sie in Amazon Simple Storage Service

(Amazon S3) oder einem anderen sicheren Speicher speichern. Der Amazon Polly the section called “SynthesizeSpeech” Polly-Betrieb ist staatenlos und nicht mit einer Kundenidentität verknüpft. Sie können ihn nicht später von Amazon Polly abrufen.

Verschlüsselung während der Übertragung

Alle Texteinsendungen sind während der Übertragung durch TLS geschützt. Amazon Polly speichert den Inhalt von Texteinsendungen nicht.

Richtlinie für den Datenverkehr zwischen Netzwerken

Der Zugriff auf Amazon Polly erfolgt über die AWS Konsole, CLI oder SDKs. Die Kommunikation nutzt die Transport Layer Security (TLS)-Sitzungsverschlüsselung für Vertraulichkeit und digitale Signaturen, um Authentifizierung und Integrität zu unterstützen.

Identity and Access Management in Amazon Polly

AWS Identity and Access Management (IAM) hilft einem Administrator AWS-Service , den Zugriff auf Ressourcen sicher zu kontrollieren. AWS IAM-Administratoren kontrollieren, wer authentifiziert (angemeldet) und autorisiert werden kann (über Berechtigungen verfügt), um Amazon Polly Polly-Ressourcen zu verwenden. IAM ist ein Programm AWS-Service , das Sie ohne zusätzliche Kosten nutzen können.

Themen

- Zielgruppe
- Authentifizierung mit Identitäten
- Verwalten des Zugriffs mit Richtlinien
- So arbeitet Amazon Polly mit IAM
- Identity-based Richtlinienbeispiele für Amazon Polly
- Amazon Polly API-Berechtigungen: Referenz zu Aktionen, Berechtigungen und Ressourcen
- Fehlerbehebung bei Identität und Zugriff auf Amazon Polly

Zielgruppe

Wie Sie AWS Identity and Access Management (IAM) verwenden, hängt von Ihrer Rolle ab:

- Servicebenutzer – Fordern Sie von Ihrem Administrator Berechtigungen an, wenn Sie nicht auf Features zugreifen können (siehe Fehlerbehebung bei Identität und Zugriff auf Amazon Polly).
- Serviceadministrator – Bestimmen Sie den Benutzerzugriff und stellen Sie Berechtigungsanfragen (siehe So arbeitet Amazon Polly mit IAM).
- IAM-Administrator – Schreiben Sie Richtlinien zur Zugriffsverwaltung (siehe Identity-based Richtlinienbeispiele für Amazon Polly).

Authentifizierung mit Identitäten

Authentifizierung ist die Art und Weise, wie Sie sich AWS mit Ihren Identitätsdaten anmelden. Sie müssen sich als IAM-Benutzer authentifizieren oder eine IAM-Rolle annehmen. Root-Benutzer des AWS-Kontos

Sie können sich als föderierte Identität anmelden, indem Sie Anmeldeinformationen aus einer Identitätsquelle wie AWS IAM Identity Center (IAM Identity Center), Single Sign-On-Authentifizierung oder Anmeldeinformationen verwenden. Google/Facebook Weitere Informationen zum Anmelden finden Sie unter So melden Sie sich bei Ihrem AWS-Konto an im Benutzerhandbuch für AWS-Anmeldung .

AWS Bietet für den programmatischen Zugriff ein SDK und eine CLI zum kryptografischen Signieren von Anfragen. Weitere Informationen finden Sie unter AWS Signature Version 4 for API requests im IAM-Benutzerhandbuch.

AWS-Konto Root-Benutzer

Wenn Sie einen erstellen AWS-Konto, beginnen Sie mit einer Anmeldeidentität, dem sogenannten AWS-Konto Root-Benutzer, der vollständigen Zugriff auf alle AWS-Services Ressourcen hat. Wir raten ausdrücklich davon ab, den Root-Benutzer für Alltagsaufgaben zu verwenden. Eine Liste der Aufgaben, für die Sie sich als Root-Benutzer anmelden müssen, finden Sie unter Tasks that require root user credentials im IAM-Benutzerhandbuch.

Verbundidentität

Es hat sich bewährt, dass menschliche Benutzer für den Zugriff AWS-Services mithilfe temporärer Anmeldeinformationen einen Verbund mit einem Identitätsanbieter verwenden müssen.

Eine föderierte Identität ist ein Benutzer aus Ihrem Unternehmensverzeichnis, Ihrem Directory Service Web-Identitätsanbieter oder der AWS-Services mithilfe von Anmeldeinformationen

aus einer Identitätsquelle zugreift. Verbundene Identitäten übernehmen Rollen, die temporäre Anmeldeinformationen bereitstellen.

Für die zentrale Zugriffsverwaltung empfehlen wir AWS IAM Identity Center. Weitere Informationen finden Sie unter Was ist IAM Identity Center? im AWS IAM Identity Center -Benutzerhandbuch.

IAM-Benutzer und -Gruppen

Ein IAM-Benutzer ist eine Identität mit bestimmten Berechtigungen für eine einzelne Person oder Anwendung. Wir empfehlen die Verwendung temporärer Anmeldeinformationen anstelle von IAM-Benutzern mit langfristigen Anmeldeinformationen. Weitere Informationen finden Sie im IAM-Benutzerhandbuch unter Erfordern, dass menschliche Benutzer den Verbund mit einem Identitätsanbieter verwenden müssen, um AWS mithilfe temporärer Anmeldeinformationen darauf zugreifen zu können.

Eine IAM-Gruppe spezifiziert eine Sammlung von IAM-Benutzern und erleichtert die Verwaltung von Berechtigungen für große Gruppen von Benutzern. Weitere Informationen finden Sie unter Anwendungsfälle für IAM-Benutzer im IAM-Benutzerhandbuch.

IAM-Rollen

Eine IAM-Rolle ist eine Identität mit spezifischen Berechtigungen, die temporäre Anmeldeinformationen bereitstellt. Sie können eine Rolle übernehmen, indem Sie von einer Benutzer- zu einer IAM-Rolle (Konsole) wechseln AWS CLI oder einen AWS API-Vorgang aufrufen. Weitere Informationen finden Sie unter Methoden, um eine Rolle zu übernehmen im IAM-Benutzerhandbuch.

IAM-Rollen sind nützlich für den Verbundbenutzer-Zugriff, temporäre IAM-Benutzerberechtigungen, kontoübergreifenden Zugriff, serviceübergreifenden Zugriff und Anwendungen, die auf Amazon EC2 laufen. Weitere Informationen finden Sie unter Kontoübergreifender Ressourcenzugriff in IAM im IAM-Benutzerhandbuch.

Verwalten des Zugriffs mit Richtlinien

Sie kontrollieren den Zugriff, AWS indem Sie Richtlinien erstellen und diese an AWS Identitäten oder Ressourcen anhängen. Eine Richtlinie definiert Berechtigungen, wenn sie mit einer Identität oder Ressource verknüpft sind. AWS bewertet diese Richtlinien, wenn ein Principal eine Anfrage stellt. Die meisten Richtlinien werden AWS als JSON-Dokumente gespeichert. Weitere Informationen zu JSON-Richtliniendokumenten finden Sie unter Übersicht über JSON-Richtlinien im IAM-Benutzerhandbuch.

Mit Hilfe von Richtlinien legen Administratoren fest, wer Zugriff auf was hat, indem sie definieren, welches Prinzipal welche Aktionen auf welchen Ressourcenund unter welchen Bedingungendurchführen darf.

Standardmäßig haben Benutzer, Gruppen und Rollen keine Berechtigungen. Ein IAM-Administrator erstellt IAM-Richtlinien und fügt sie zu Rollen hinzu, die die Benutzer dann übernehmen können. IAM-Richtlinien definieren Berechtigungen unabhängig von der Methode, die zur Ausführung der Operation verwendet wird.

Identity-based Richtlinien

Identity-based Richtlinien sind Richtliniendokumente für JSON-Berechtigungen, die Sie an eine Identität (Benutzer, Gruppe oder Rolle) anhängen. Diese Richtlinien steuern, welche Aktionen Identitäten für welche Ressourcen und unter welchen Bedingungen ausführen können. Informationen zum Erstellen identitätsbasierter Richtlinien finden Sie unter Definieren benutzerdefinierter IAM-Berechtigungen mit vom Kunden verwalteten Richtlinien im IAM-Benutzerhandbuch.

Identity-based Richtlinien können Inline-Richtlinien (direkt in eine einzelne Identität eingebettet) oder verwaltete Richtlinien (eigenständige Richtlinien, die mehreren Identitäten zugeordnet sind) sein. Informationen dazu, wie Sie zwischen verwalteten und Inline-Richtlinien wählen, finden Sie unter Choose between managed policies and inline policies im IAM-Benutzerhandbuch.

Resource-based Richtlinien

Resource-based Richtlinien sind JSON-Richtliniendokumente, die Sie an eine Ressource anhängen. Beispiele hierfür sind Vertrauensrichtlinien für IAM-Rollen und Amazon S3-Bucket-Richtlinien. In Services, die ressourcenbasierte Richtlinien unterstützen, können Service-Administratoren sie verwenden, um den Zugriff auf eine bestimmte Ressource zu steuern. Sie müssen in einer ressourcenbasierten Richtlinie einen Prinzipal angeben.

Resource-based Richtlinien sind Inline-Richtlinien, die sich in diesem Dienst befinden. Sie können AWS verwaltete Richtlinien von IAM nicht in einer ressourcenbasierten Richtlinie verwenden.

Weitere Richtlinientypen

AWS unterstützt zusätzliche Richtlinientypen, mit denen die maximalen Berechtigungen festgelegt werden können, die durch gängigere Richtlinientypen gewährt werden:

- Berechtigungsgrenzen – Eine Berechtigungsgrenze legt die maximalen Berechtigungen fest, die eine identitätsbasierte Richtlinie einer IAM-Entität erteilen kann. Weitere Informationen finden Sie unter Berechtigungsgrenzen für IAM-Entitäten im -IAM-Benutzerhandbuch.

- Service-Kontrollrichtlinien (SCPs) – SCPs legen die maximalen Berechtigungen für eine Organisation oder Organisationseinheit in AWS Organizations fest. Weitere Informationen finden Sie unter Service-Kontrollrichtlinien im AWS Organizations -Benutzerhandbuch.
- Ressourcen-Kontrollrichtlinien (RCPs) – RCPs definieren die maximale Anzahl an Berechtigungen, die Ressourcen in Ihren Konten zur Verfügung stehen. Weitere Informationen finden Sie unter Ressourcen-Kontrollrichtlinien im AWS Organizations -Benutzerhandbuch.
- Sitzungsrichtlinien – Sitzungsrichtlinien sind erweiterte Richtlinien, die als Parameter übergeben werden, wenn Sie eine temporäre Sitzung für eine Rolle oder einen Verbundbenutzer erstellen. Weitere Informationen finden Sie unter Sitzungsrichtlinien im IAM-Benutzerhandbuch.

Mehrere Richtlinientypen

Wenn mehrere Arten von Richtlinien für eine Anfrage gelten, sind die sich daraus ergebenden Berechtigungen schwieriger zu verstehen. Informationen darüber, wie AWS bestimmt wird, ob eine Anfrage zulässig ist, wenn mehrere Richtlinientypen betroffen sind, finden Sie unter Bewertungslogik für Richtlinien im IAM-Benutzerhandbuch.

So arbeitet Amazon Polly mit IAM

Bevor Sie IAM verwenden, um den Zugriff auf Amazon Polly zu verwalten, sollten Sie sich darüber informieren, welche IAM-Funktionen für Amazon Polly verfügbar sind.

IAM-Funktionen, die Sie mit Amazon Polly verwenden können

IAM-Feature	Amazon Polly Polly-Unterstützung
Identity-based Richtlinien	Ja
Resource-based Richtlinien	Nein
Richtlinienaktionen	Ja
Richtlinienressourcen	Ja
Richtlinienbedingungsschlüssel (services pezifisch)	Nein
ACLs	Nein

IAM-Feature	Amazon Polly Polly-Unterstützung
ABAC (Tags in Richtlinien)	Nein
Temporäre Anmeldeinformationen	Ja
Forward Access Sessions (FAS) für Amazon Polly	Ja
Servicerollen	Nein
Service-linked Rollen	Nein

Einen allgemeinen Überblick darüber, wie Amazon Polly und andere AWS Dienste mit den meisten IAM-Funktionen funktionieren, finden Sie im AWS IAM-Benutzerhandbuch unter Dienste, die mit IAM funktionieren.

Identity-based Richtlinien für Amazon Polly

Unterstützt Richtlinien auf Identitätsbasis: Ja

Identity-based Richtlinien sind Richtliniendokumente für JSON-Berechtigungen, die Sie an eine Identität anhängen können, z. B. an einen IAM-Benutzer, eine Benutzergruppe oder eine Rolle. Diese Richtlinien steuern, welche Aktionen die Benutzer und Rollen für welche Ressourcen und unter welchen Bedingungen ausführen können. Informationen zum Erstellen identitätsbasierter Richtlinien finden Sie unter Definieren benutzerdefinierter IAM-Berechtigungen mit vom Kunden verwalteten Richtlinien im IAM-Benutzerhandbuch.

Mit identitätsbasierten IAM-Richtlinien können Sie angeben, welche Aktionen und Ressourcen zugelassen oder abgelehnt werden. Darüber hinaus können Sie die Bedingungen festlegen, unter denen Aktionen zugelassen oder abgelehnt werden. Informationen zu sämtlichen Elementen, die Sie in einer JSON-Richtlinie verwenden, finden Sie in der IAM-Referenz für JSON-Richtlinienelemente im IAM-Benutzerhandbuch.

Identity-based Richtlinienbeispiele für Amazon Polly

Beispiele für identitätsbasierte Richtlinien von Amazon Polly finden Sie unter. Identity-based Richtlinienbeispiele für Amazon Polly

Resource-based Richtlinien innerhalb von Amazon Polly

Unterstützt ressourcenbasierte Richtlinien: Nein

Resource-based Richtlinien sind JSON-Richtliniendokumente, die Sie an eine Ressource anhängen. Beispiele für ressourcenbasierte Richtlinien sind IAM-Rollen-Vertrauensrichtlinien und Amazon-S3-Bucket-Richtlinien. In Services, die ressourcenbasierte Richtlinien unterstützen, können Service-Administratoren sie verwenden, um den Zugriff auf eine bestimmte Ressource zu steuern. Für die Ressource, an welche die Richtlinie angehängt ist, legt die Richtlinie fest, welche Aktionen ein bestimmter Prinzipal unter welchen Bedingungen für diese Ressource ausführen kann. Sie müssen in einer ressourcenbasierten Richtlinie einen Prinzipal angeben. Zu den Prinzipalen können Konten, Benutzer, Rollen, Verbundbenutzer oder gehören. AWS-Services

Um kontoübergreifenden Zugriff zu ermöglichen, können Sie ein gesamtes Konto oder IAM-Entitäten in einem anderen Konto als Prinzipal in einer ressourcenbasierten Richtlinie angeben. Weitere Informationen finden Sie unter Kontoübergreifender Ressourcenzugriff in IAM im IAM-Benutzerhandbuch.

Politische Maßnahmen für Amazon Polly

Unterstützt Richtlinienaktionen: Ja

Administratoren können mithilfe von AWS JSON-Richtlinien angeben, wer auf was Zugriff hat. Das heißt, welcher Prinzipal Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen kann.

Das Element `Action` einer JSON-Richtlinie beschreibt die Aktionen, mit denen Sie den Zugriff in einer Richtlinie zulassen oder verweigern können. Nehmen Sie Aktionen in eine Richtlinie auf, um Berechtigungen zur Ausführung des zugehörigen Vorgangs zu erteilen.

Eine Liste der Amazon Polly-Aktionen finden Sie unter Von Amazon Polly definierte Aktionen in der Service Authorization Reference.

Richtlinienaktionen in Amazon Polly verwenden das folgende Präfix vor der Aktion:

```
polly
```

Um mehrere Aktionen in einer einzigen Anweisung anzugeben, trennen Sie sie mit Kommata:

```
"Action": [
```

```
      "polly:action1",
      "polly:action2"
         ]
```

Beispiele für identitätsbasierte Richtlinien von Amazon Polly finden Sie unter. Identity-based Richtlinienbeispiele für Amazon Polly

Richtlinienressourcen für Amazon Polly

Unterstützt Richtlinienressourcen: Ja

Administratoren können mithilfe von AWS JSON-Richtlinien angeben, wer auf was Zugriff hat. Das heißt, welcher Prinzipal Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen kann.

Das JSON-Richtlinienelement `Resource` gibt die Objekte an, auf welche die Aktion angewendet wird. Als Best Practice geben Sie eine Ressource mit dem zugehörigen Amazon-Ressourcennamen (ARN) an. Verwenden Sie für Aktionen, die keine Berechtigungen auf Ressourcenebene unterstützen, einen Platzhalter (*), um anzugeben, dass die Anweisung für alle Ressourcen gilt.

```
"Resource": "*"
```

Eine Liste der Amazon Polly-Ressourcentypen und ihrer ARNs finden Sie unter Von Amazon Polly definierte Ressourcen in der Service Authorization Reference. Informationen darüber, mit welchen Aktionen Sie den ARN jeder Ressource angeben können, finden Sie unter Von Amazon Polly definierte Aktionen.

Beispiele für identitätsbasierte Richtlinien von Amazon Polly finden Sie unter. Identity-based Richtlinienbeispiele für Amazon Polly

Schlüssel zu den Versicherungsbedingungen für Amazon Polly

Unterstützt servicespezifische Richtlinienbedingungsschlüssel: Nein

Administratoren können mithilfe von AWS JSON-Richtlinien angeben, wer auf was Zugriff hat. Das heißt, welcher Prinzipal Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen kann.

Das Element `Condition` gibt an, wann Anweisungen auf der Grundlage definierter Kriterien ausgeführt werden. Sie können bedingte Ausdrücke erstellen, die Bedingungsoperatoren verwenden, z. B. ist gleich oder kleiner als, damit die Bedingung in der Richtlinie mit Werten in der Anforderung übereinstimmt. Eine Übersicht aller AWS globalen Bedingungsschlüssel finden Sie unter Kontextschlüssel für AWS globale Bedingungen im IAM-Benutzerhandbuch.

Eine Liste der Amazon Polly-Bedingungsschlüssel finden Sie unter Bedingungsschlüssel für Amazon Polly in der Service Authorization Reference. Informationen zu den Aktionen und Ressourcen, mit denen Sie einen Bedingungsschlüssel verwenden können, finden Sie unter Von Amazon Polly definierte Aktionen.

Beispiele für identitätsbasierte Richtlinien von Amazon Polly finden Sie unter. Identity-based Richtlinienbeispiele für Amazon Polly

ACLs bei Amazon Polly

Unterstützt ACLs: Nein

Zugriffssteuerungslisten (ACLs) steuern, welche Prinzipale (Kontomitglieder, Benutzer oder Rollen) auf eine Ressource zugreifen können. ACLs sind ähnlich wie ressourcenbasierte Richtlinien, verwenden jedoch nicht das JSON-Richtliniendokumentformat.

ABAC mit Amazon Polly

Unterstützt ABAC (Tags in Richtlinien): Nein

Attribute-based Access Control (ABAC) ist eine Autorisierungsstrategie, die Berechtigungen auf der Grundlage von Attributen definiert, die als Tags bezeichnet werden. Sie können Tags an IAM-Entitäten und AWS -Ressourcen anhängen und dann ABAC-Richtlinien entwerfen, die Operationen zulassen, wenn das Tag des Prinzipals mit dem Tag auf der Ressource übereinstimmt.

Um den Zugriff auf der Grundlage von Tags zu steuern, geben Sie im Bedingungselement einer Richtlinie Tag-Informationen an, indem Sie die Schlüssel `aws:ResourceTag/`*`key-name`*, `aws:RequestTag/`*`key-name`*, oder Bedingung `aws:TagKeys` verwenden.

Wenn ein Service alle drei Bedingungsschlüssel für jeden Ressourcentyp unterstützt, lautet der Wert für den Service Ja. Wenn ein Service alle drei Bedingungsschlüssel für nur einige Ressourcentypen unterstützt, lautet der Wert Teilweise.

Weitere Informationen zu ABAC finden Sie unter Definieren von Berechtigungen mit ABAC-Autorisierung im IAM-Benutzerhandbuch. Um ein Tutorial mit Schritten zur Einstellung von ABAC anzuzeigen, siehe Attributbasierte Zugriffskontrolle (ABAC) verwenden im IAM-Benutzerhandbuch.

Temporäre Anmeldeinformationen mit Amazon Polly verwenden

Unterstützt temporäre Anmeldeinformationen: Ja

Temporäre Anmeldeinformationen ermöglichen kurzfristigen Zugriff auf AWS Ressourcen und werden automatisch erstellt, wenn Sie den Verbund verwenden oder die Rollen wechseln. AWS empfiehlt, temporäre Anmeldeinformationen dynamisch zu generieren, anstatt langfristige Zugriffsschlüssel zu verwenden. Weitere Informationen finden Sie unter Temporäre Anmeldeinformationen in IAM und AWS-Services , die mit IAM funktionieren im IAM-Benutzerhandbuch.

Cross-service Forward Access Sessions (FAS) für Amazon Polly

Unterstützt Forward Access Sessions (FAS): Ja

Forward Access Sessions (FAS) verwenden die Berechtigungen des Principals, der einen aufruft AWS-Service, kombiniert mit der Anfrage, Anfragen an AWS-Service nachgelagerte Dienste zu stellen. Einzelheiten zu den Richtlinien für FAS-Anforderungen finden Sie unter Zugriffssitzungen weiterleiten.

Servicerollen für Amazon Polly

Unterstützt Servicerollen: Nein

Eine Servicerolle ist eine IAM-Rolle, die ein Service annimmt, um Aktionen in Ihrem Namen auszuführen. Ein IAM-Administrator kann eine Servicerolle innerhalb von IAM erstellen, ändern und löschen. Weitere Informationen finden Sie unter Erstellen einer Rolle zum Delegieren von Berechtigungen an einen AWS-Service im IAM-Benutzerhandbuch.

> **Warning**
>
> Das Ändern der Berechtigungen für eine Servicerolle kann die Funktionalität von Amazon Polly beeinträchtigen. Bearbeiten Sie Servicerollen nur, wenn Amazon Polly Sie dazu anleitet.

Service-linked Rollen für Amazon Polly

Unterstützt serviceverknüpfte Rollen: Ja

Eine dienstbezogene Rolle ist eine Art von Servicerolle, die mit einer AWS-Service verknüpft ist. Der Dienst kann die Rolle übernehmen, eine Aktion in Ihrem Namen auszuführen. Service-linked Rollen erscheinen in Ihrem Dienst AWS-Konto und gehören dem Dienst. Ein IAM-Administrator kann die Berechtigungen für Service-verknüpfte Rollen anzeigen, aber nicht bearbeiten.

Details zum Erstellen oder Verwalten von serviceverknüpften Rollen finden Sie unter AWS -Services, die mit IAM funktionieren. Suchen Sie in der Tabelle nach einem Dienst, der einen `Yes` in der Service-linked Rollenspalte enthält. Wählen Sie den Link Yes (Ja) aus, um die Dokumentation für die serviceverknüpfte Rolle für diesen Service anzuzeigen.

IAM-Rollen in Amazon Polly

Sie können einer IAM-Rolle eine identitätsbasierte Berechtigungsrichtlinie zuordnen, um kontoübergreifende Berechtigungen zu gewähren. Der Administrator in Konto A kann beispielsweise wie folgt eine Rolle erstellen, um einem anderen AWS Konto (z. B. Konto B) oder einem Dienst kontoübergreifende Berechtigungen zu gewähren: AWS

1. Der Administrator von Konto A erstellt eine IAM-Rolle und fügt ihr eine Berechtigungsrichtlinie an, die Berechtigungen für Ressourcen in Konto A erteilt.
2. Der Administrator von Konto A weist der Rolle eine Vertrauensrichtlinie zu, die Konto B als den Prinzipal identifiziert, der die Rolle übernehmen kann.
3. Der Administrator von Konto B kann dann die Berechtigungen zur Übernahme der Rolle an alle Benutzer in Konto B delegieren. Auf diese Weise können Benutzer in Konto B Ressourcen in Konto A erstellen oder darauf zugreifen. Der Principal in der Vertrauensrichtlinie kann auch ein AWS Dienstprinzipal sein, wenn Sie einem AWS Dienst die Berechtigungen zur Übernahme der Rolle erteilen möchten.

Weitere Informationen zum Delegieren von Berechtigungen mithilfe von IAM finden Sie unter Zugriffsverwaltung im IAM-Benutzerhandbuch.

Nachfolgend sehen Sie eine Beispielrichtlinie, die den Benutzer dazu berechtigt, Lexika in einer Region zu speichern, Lexika abzurufen sowie alle aktuell verfügbaren Lexika aufzulisten.

Amazon Polly unterstützt Identity-based Richtlinien für Aktionen auf Ressourcenebene. In einigen Fällen kann die Ressource durch einen ARN begrenzt sein. Dies gilt für die Operationen `SynthesizeSpeech`, `StartSpeechSynthesisTask`, `PutLexicon`, `GetLexicon` und `DeleteLexicon`. In diesen Fällen wird der `Resource`-Wert durch den ARN angezeigt. Beispiel:

`arn:aws:polly:us-east-2:account-id:lexicon/*` als `Resource`-Wert definiert Berechtigungen für alle Lexika im Besitz des angegebenen Benutzers in der Region `us-east-2`.

Allerdings verwenden nicht alle Operationen ARNs. Dies ist bei den `ListSpeechSynthesisTasks` Operationen`DescribeVoices`, `ListLexiconsGetSpeechSynthesisTasks`, und der Fall.

Weitere Informationen zu Benutzern, Gruppen, Rollen und Berechtigungen finden Sie im Thema Identitäten (Benutzer, Gruppen und Rollen) im IAM-Benutzerhandbuch.

Identity-based Richtlinienbeispiele für Amazon Polly

Standardmäßig sind Benutzer und Rollen nicht berechtigt, Amazon Polly Polly-Ressourcen zu erstellen oder zu ändern. Ein IAM-Administrator muss IAM-Richtlinien erstellen, die Benutzern die Berechtigung erteilen, Aktionen für die Ressourcen auszuführen, die sie benötigen.

Informationen dazu, wie Sie unter Verwendung dieser beispielhaften JSON-Richtliniendokumente eine identitätsbasierte IAM-Richtlinie erstellen, finden Sie unter Erstellen von IAM-Richtlinien (Konsole) im IAM-Benutzerhandbuch.

Einzelheiten zu den von Amazon Polly definierten Aktionen und Ressourcentypen, einschließlich des Formats der ARNs für jeden Ressourcentyp, finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für Amazon Polly in der Service Authorization Reference.

Themen

- Best Practices für Richtlinien
- Verwenden der Amazon Polly Polly-Konsole
- Gewähren der Berechtigung zur Anzeige der eigenen Berechtigungen für Benutzer
- AWS verwaltete (vordefinierte) Richtlinien für Amazon Polly
- Amazon Polly aktualisiert auf AWS Verwaltete Richtlinien
- Customer-managed politische Beispiele

Best Practices für Richtlinien

Identity-based Richtlinien legen fest, ob jemand Amazon Polly Polly-Ressourcen in Ihrem Konto erstellen, darauf zugreifen oder diese löschen kann. Dies kann zusätzliche Kosten für Ihr verursachen AWS-Konto. Wenn Sie identitätsbasierte Richtlinien erstellen oder bearbeiten, befolgen Sie diese Richtlinien und Empfehlungen:

- Erste Schritte mit AWS verwalteten Richtlinien und Umstellung auf Berechtigungen mit den geringsten Rechten — Verwenden Sie die AWS verwalteten Richtlinien, die Berechtigungen für viele gängige Anwendungsfälle gewähren, um damit zu beginnen, Ihren Benutzern und Workloads Berechtigungen zu gewähren. Sie sind in Ihrem verfügbar. AWS-Konto Wir empfehlen Ihnen, die Berechtigungen weiter zu reduzieren, indem Sie vom AWS Kunden verwaltete Richtlinien definieren, die speziell auf Ihre Anwendungsfälle zugeschnitten sind. Weitere Informationen finden Sie unter Von AWS verwaltete Richtlinien oder Von AWS verwaltete Richtlinien für Auftragsfunktionen im IAM-Benutzerhandbuch.
- Anwendung von Berechtigungen mit den geringsten Rechten – Wenn Sie mit IAM-Richtlinien Berechtigungen festlegen, gewähren Sie nur die Berechtigungen, die für die Durchführung einer Aufgabe erforderlich sind. Sie tun dies, indem Sie die Aktionen definieren, die für bestimmte Ressourcen unter bestimmten Bedingungen durchgeführt werden können, auch bekannt als die geringsten Berechtigungen. Weitere Informationen zur Verwendung von IAM zum Anwenden von Berechtigungen finden Sie unter Richtlinien und Berechtigungen in IAM im IAM-Benutzerhandbuch.
- Verwenden von Bedingungen in IAM-Richtlinien zur weiteren Einschränkung des Zugriffs – Sie können Ihren Richtlinien eine Bedingung hinzufügen, um den Zugriff auf Aktionen und Ressourcen zu beschränken. Sie können beispielsweise eine Richtlinienbedingung schreiben, um festzulegen, dass alle Anforderungen mithilfe von SSL gesendet werden müssen. Sie können auch Bedingungen verwenden, um Zugriff auf Serviceaktionen zu gewähren, wenn diese für einen bestimmten Zweck verwendet werden AWS-Service, z. CloudFormation B. Weitere Informationen finden Sie unter IAM-JSON-Richtlinienelemente: Bedingung im IAM-Benutzerhandbuch.
- Verwenden von IAM Access Analyzer zur Validierung Ihrer IAM-Richtlinien, um sichere und funktionale Berechtigungen zu gewährleisten – IAM Access Analyzer validiert neue und vorhandene Richtlinien, damit die Richtlinien der IAM-Richtliniensprache (JSON) und den bewährten IAM-Methoden entsprechen. IAM Access Analyzer stellt mehr als 100 Richtlinienprüfungen und umsetzbare Empfehlungen zur Verfügung, damit Sie sichere und funktionale Richtlinien erstellen können. Weitere Informationen finden Sie unter Richtlinienvalidierung mit IAM Access Analyzer im IAM-Benutzerhandbuch.
- Multi-Faktor-Authentifizierung (MFA) erforderlich — Wenn Sie ein Szenario haben, das IAM-Benutzer oder einen Root-Benutzer in Ihrem System erfordert AWS-Konto, aktivieren Sie MFA für zusätzliche Sicherheit. Um MFA beim Aufrufen von API-Vorgängen anzufordern, fügen Sie Ihren Richtlinien MFA-Bedingungen hinzu. Weitere Informationen finden Sie unter Sicherer API-Zugriff mit MFA im IAM-Benutzerhandbuch.

Weitere Informationen zu bewährten Methoden in IAM finden Sie unter Best Practices für die Sicherheit in IAM im IAM-Benutzerhandbuch.

Verwenden der Amazon Polly Polly-Konsole

Um auf die Amazon Polly Polly-Konsole zugreifen zu können, benötigen Sie ein Mindestmaß an Berechtigungen. Diese Berechtigungen müssen es Ihnen ermöglichen, Details zu den Amazon Polly Polly-Ressourcen in Ihrem AWS-Konto aufzulisten und anzuzeigen. Wenn Sie eine identitätsbasierte Richtlinie erstellen, die strenger ist als die mindestens erforderlichen Berechtigungen, funktioniert die Konsole nicht wie vorgesehen für Entitäten (Benutzer oder Rollen) mit dieser Richtlinie.

Sie müssen Benutzern, die nur die API AWS CLI oder die AWS API aufrufen, keine Mindestberechtigungen für die Konsole gewähren. Stattdessen sollten Sie nur Zugriff auf die Aktionen zulassen, die der API-Operation entsprechen, die die Benutzer ausführen möchten.

Um sicherzustellen, dass Benutzer und Rollen weiterhin die Amazon Polly-Konsole verwenden können, fügen Sie den Entitäten auch die Amazon Polly *ConsoleAccess* - oder *ReadOnly* AWS verwaltete Richtlinie hinzu. Weitere Informationen finden Sie unter Hinzufügen von Berechtigungen zu einem Benutzer im IAM-Benutzerhandbuch.

Um die Amazon Polly Polly-Konsole zu verwenden, gewähren Sie allen Amazon Polly Polly-APIs Berechtigungen. Weitere Berechtigungen sind nicht erforderlich. Um die vollständige Konsolenfunktionalität zu erhalten, können Sie die folgende Richtlinie verwenden:

Gewähren der Berechtigung zur Anzeige der eigenen Berechtigungen für Benutzer

In diesem Beispiel wird gezeigt, wie Sie eine Richtlinie erstellen, die IAM-Benutzern die Berechtigung zum Anzeigen der eingebundenen Richtlinien und verwalteten Richtlinien gewährt, die ihrer Benutzeridentität angefügt sind. Diese Richtlinie beinhaltet Berechtigungen zum Ausführen dieser Aktion auf der Konsole oder programmgesteuert mithilfe der API oder. AWS CLI AWS

```
{
    "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 verwaltete (vordefinierte) Richtlinien für Amazon Polly

AWS adressiert viele gängige Anwendungsfälle durch die Bereitstellung eigenständiger IAM-Richtlinien, die von erstellt und verwaltet werden. AWS Diese AWS verwalteten Richtlinien gewähren die erforderlichen Berechtigungen für allgemeine Anwendungsfälle, sodass Sie nicht erst untersuchen müssen, welche Berechtigungen benötigt werden. Weitere Informationen finden Sie unter AWS - verwaltete Richtlinien im IAM-Benutzerhandbuch.

Die folgenden AWS verwalteten Richtlinien, die Sie Benutzern in Ihrem Konto zuordnen können, sind spezifisch für Amazon Polly:

- AmazonPollyReadOnlyAccess— Gewährt schreibgeschützten Zugriff auf Ressourcen, ermöglicht das Auflisten von Lexika, das Abrufen von Lexika, das Auflisten verfügbarer Stimmen und das Synthetisieren von Sprache (einschließlich der Anwendung von Lexika auf die synthetisierte Sprache).

  Weitere Einzelheiten zu dieser Richtlinie, einschließlich der neuesten Version des JSON-Richtliniendokuments, finden Sie unter AmazonPollyReadOnlyAccess im AWS Referenzhandbuch für verwaltete Richtlinien.

- AmazonPollyFullAccess— Gewährt vollen Zugriff auf Ressourcen und alle unterstützten Operationen.

  Weitere Einzelheiten zu dieser Richtlinie, einschließlich der neuesten Version des JSON-Richtliniendokuments, finden Sie unter AmazonPollyFullAccess im AWS Referenzhandbuch für verwaltete Richtlinien.

> **Note**
>
> Sie können diese Berechtigungsrichtlinien prüfen, indem Sie sich bei der IAM-Konsole anmelden und dort nach bestimmten Richtlinien suchen.

Sie können auch Ihre eigenen benutzerdefinierten IAM-Richtlinien erstellen, um Berechtigungen für Amazon Polly Polly-Aktionen und -Ressourcen zu gewähren. Die benutzerdefinierten Richtlinien können Sie dann den IAM-Benutzern oder -Gruppen zuweisen, die diese Berechtigungen benötigen.

Amazon Polly aktualisiert auf AWS Verwaltete Richtlinien

Sehen Sie sich Details zu Aktualisierungen der AWS verwalteten Richtlinien für Amazon Polly an, seit dieser Service begonnen hat, diese Änderungen zu verfolgen. Abonnieren Sie den RSS-Feed auf der Amazon Polly Document History-Seite, um automatische Benachrichtigungen über Änderungen an dieser Seite zu erhalten.

Amazon Polly aktualisiert auf AWS Verwaltete Richtlinien

Richtlinie	Änderungen	Date
AmazonPollyReadOnlyAccess	Aktualisierte verwaltete Richtlinie — Die Genehmigung `polly:StartSpeechSynthesisStream` zur Unterstützung bidirektionaler Streaming-Sprachsynthesefunktionen wurde hinzugefügt.	19. März 2026

Customer-managed politische Beispiele

In diesem Abschnitt finden Sie Beispielbenutzerrichtlinien, die Berechtigungen für verschiedene Amazon Polly Polly-Aktionen gewähren. Diese Richtlinien funktionieren, wenn Sie AWS SDKs oder

die verwenden. AWS CLI Wenn Sie die Konsole verwenden, gewähren Sie allen Amazon Polly Polly-APIs Berechtigungen.

> **Note**
>
> In allen Beispielen werden die Region „us-east-2“ und fiktive Konto-IDs verwendet.

Beispiele

- Beispiel 1: Alle Amazon Polly Polly-Aktionen zulassen
- Beispiel 2: Alle Amazon Polly Polly-Aktionen zulassen außer DeleteLexicon
- Beispiel 3: Zulassen DeleteLexicon
- Beispiel 4: Erlaube das Löschen eines Lexikons in einer bestimmten Region
- Beispiel 5: Erlaube das angegebene DeleteLexicon Lexikon

Beispiel 1: Alle Amazon Polly Polly-Aktionen zulassen

Nach der Registrierung (siehe Erste Schritte mit Amazon Polly) erstellen Sie einen Administratorbenutzer, der Ihr Konto verwaltet. Dieser Administrator kann unter anderem Benutzer erstellen und Benutzerberechtigungen verwalten.

Sie können einen Benutzer erstellen, der über Berechtigungen für alle Amazon Polly Polly-Aktionen verfügt. Stellen Sie sich diesen Benutzer als dienstspezifischen Administrator für die Arbeit mit Amazon Polly vor. Diesem Benutzer können Sie die folgende Berechtigungsrichtlinie zuweisen:

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [{
      "Sid": "AllowAllPollyActions",
      "Effect": "Allow",
      "Action": [
         "polly:*"],
      "Resource": "*"
      }
   ]
```

```
}
```

Beispiel 2: Alle Amazon Polly Polly-Aktionen zulassen außer DeleteLexicon

Die folgende Berechtigungsrichtlinie erteilt dem Benutzer Berechtigungen zur Ausführung sämtlicher Aktionen außer der Aktion `DeleteLexicon`. Die Berechtigung zum Löschen wird in allen Regionen explizit abgelehnt.

Beispiel 3: Zulassen DeleteLexicon

Die folgende Berechtigungsrichtlinie gewährt dem Benutzer die Berechtigung zum Löschen aller Lexika in seinem Besitz. Es spielt keine Rolle, zu welchem Projekt das Lexikon gehört oder in welcher Region es sich befindet.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [{
      "Sid": "AllowDeleteLexicon",
      "Effect": "Allow",
      "Action": [
         "polly:DeleteLexicon"],
      "Resource": "*"
      }
   ]
}
```

Beispiel 4: Erlaube das Löschen eines Lexikons in einer bestimmten Region

Die folgende Berechtigungsrichtlinie erteilt dem Benutzer Berechtigungen zur Löschung jedes beliebigen Lexikons, das sich in einer bestimmten Region befindet (hier „us-east-2“). Es spielt keine Rolle, zu welchem Projekt das Lexikon gehört.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [{
```

```
      "Sid": "AllowDeleteSpecifiedRegion",
      "Effect": "Allow",
      "Action": [
         "polly:DeleteLexicon"],
      "Resource": "arn:aws:polly:us-east-2:111122223333:lexicon/*"
      }
   ]
}
```

Beispiel 5: Erlaube das angegebene DeleteLexicon Lexikon

Die folgende Berechtigungsrichtlinie erteilt dem Benutzer Berechtigungen zur Löschung eines bestimmten in Ihrem Besitz befindlichen Lexikons (hier „myLexicon“) in einer bestimmten Region (hier „us-east-2“).

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [{
      "Sid": "AllowDeleteForSpecifiedLexicon",
      "Effect": "Allow",
      "Action": [
         "polly:DeleteLexicon"],
      "Resource": "arn:aws:polly:us-east-2:111122223333:lexicon/myLexicon"
      }
   ]
}
```

Amazon Polly API-Berechtigungen: Referenz zu Aktionen, Berechtigungen und Ressourcen

Wenn Sie eine Berechtigungsrichtlinie einrichten, die Sie an eine IAM-Identität anhängen können (identitätsbasierte Richtlinien), können Sie die folgende als Referenz verwenden. Die enthält jeden Amazon Polly Polly-API-Vorgang, die entsprechenden Aktionen, für die Sie Berechtigungen zur Ausführung der Aktion erteilen können, und die AWS Ressource, für die Sie die Berechtigungen erteilen können. Die Aktionen geben Sie im Feld `Action` und den Wert für die Ressource im Feld `Resource` der Richtlinie an.

Sie können in Ihren Amazon Polly Polly-Richtlinien allgemeine Bedingungsschlüssel verwenden AWS, um Bedingungen auszudrücken. Eine vollständige Liste der Schlüssel für die gesamte AWS Breite finden Sie im IAM-Benutzerhandbuch unter Verfügbare Schlüssel.

> **Note**
>
> Um eine Aktion anzugeben, verwenden Sie das Präfix `polly` gefolgt vom Namen der API-Operation (z. B. `polly:GetLexicon`).

Amazon Polly unterstützt Identity-based Richtlinien für Aktionen auf Ressourcenebene. Daher wird für den Wert `Resource` der ARN angegeben. Beispiel: `arn:aws:polly:`*`us-east-2`*`:`*`account-id`*`:lexicon/*` als `Resource`-Wert definiert Berechtigungen für alle Lexika im Besitz des angegebenen Benutzers in der Region `us-east-2`.

Da Amazon Polly keine Berechtigungen für Aktionen auf Ressourcenebene unterstützt, geben die meisten Richtlinien ein Platzhalterzeichen (*) als Wert an. `Resource` Wenn Berechtigungen auf eine bestimmte Region eingeschränkt werden müssen, wird das Platzhalterzeichen durch den entsprechenden ARN ersetzt: `arn:aws:polly:`*`region`*`:`*`account-id`*`:lexicon/*`.

Amazon Polly Polly-API und erforderliche Berechtigungen für Aktionen

API-Operation: DeleteLexicon

Erforderliche Berechtigungen (API-Aktion): `polly:DeleteLexicon`

Ressourcen: `arn:aws:polly:`*`region`*`:`*`account-id`*`:lexicon/`*`LexiconName`*

API-Operation: DescribeVoices

Erforderliche Berechtigungen (API-Aktion): `polly:DescribeVoices`

Ressourcen: `arn:aws:polly:`*`region`*`:`*`account-id`*`:lexicon/`*`voice-name`*

API-Operation: GetLexicon

Erforderliche Berechtigungen (API-Aktion): `polly:GetLexicon`

Ressourcen: `arn:aws:polly:`*`region`*`:`*`account-id`*`:lexicon/`*`voice-name`*

API-Operation: ListLexicons

Erforderliche Berechtigungen (API-Aktion): `polly:ListLexicons`

Ressourcen: `arn:aws:polly:region:account-id:lexicon/*`

API-Operation: PutLexicon

Erforderliche Berechtigungen (API-Aktion): `polly:ListLexicons`

Ressourcen: *

API-Operation: SynthesizeSpeech

Erforderliche Berechtigungen (API-Aktion): `polly:SynthesizeSpeech`

Ressourcen: *

Fehlerbehebung bei Identität und Zugriff auf Amazon Polly

Verwenden Sie die folgenden Informationen, um häufig auftretende Probleme zu diagnostizieren und zu beheben, die bei der Arbeit mit Amazon Polly und IAM auftreten können.

Themen

- Ich bin nicht berechtigt, eine Aktion in Amazon Polly durchzuführen
- Ich bin nicht berechtigt, iam durchzuführen: PassRole
- Ich möchte Personen außerhalb meiner Umgebung zulassen AWS-Konto um auf meine Amazon Polly Polly-Ressourcen zuzugreifen

Ich bin nicht berechtigt, eine Aktion in Amazon Polly durchzuführen

Wenn Sie eine Fehlermeldung erhalten, dass Sie nicht zur Durchführung einer Aktion berechtigt sind, müssen Ihre Richtlinien aktualisiert werden, damit Sie die Aktion durchführen können.

Der folgende Beispielfehler tritt auf, wenn der IAM-Benutzer `mateojackson` versucht, über die Konsole Details zu einer fiktiven *`my-example-widget`*-Ressource anzuzeigen, jedoch nicht über `polly:`*`GetWidget`*-Berechtigungen verfügt.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 polly:GetWidget on resource: my-example-widget
```

In diesem Fall muss die Richtlinie für den Benutzer `mateojackson` aktualisiert werden, damit er mit der `polly:GetWidget`-Aktion auf die *`my-example-widget`*-Ressource zugreifen kann.

Wenn Sie Hilfe benötigen, wenden Sie sich an Ihren AWS Administrator. Ihr Administrator hat Ihnen Ihre Anmeldeinformationen zur Verfügung gestellt.

Ich bin nicht berechtigt, iam durchzuführen: PassRole

Wenn Sie eine Fehlermeldung erhalten, dass Sie nicht berechtigt sind, die `iam:PassRole` Aktion durchzuführen, müssen Ihre Richtlinien aktualisiert werden, damit Sie eine Rolle an Amazon Polly übergeben können.

Einige AWS-Services ermöglichen es Ihnen, eine bestehende Rolle an diesen Service zu übergeben, anstatt eine neue Servicerolle oder eine dienstbezogene Rolle zu erstellen. Hierzu benötigen Sie Berechtigungen für die Übergabe der Rolle an den Dienst.

Der folgende Beispielfehler tritt auf, wenn ein IAM-Benutzer mit dem Namen `marymajor` versucht, die Konsole zu verwenden, um eine Aktion in Amazon Polly auszuführen. Die Aktion erfordert jedoch, dass der Service über Berechtigungen verfügt, die durch eine Servicerolle gewährt werden. Mary besitzt keine Berechtigungen für die Übergabe der Rolle an den Dienst.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

In diesem Fall müssen die Richtlinien von Mary aktualisiert werden, um die Aktion `iam:PassRole` ausführen zu können.

Wenn Sie Hilfe benötigen, wenden Sie sich an Ihren AWS Administrator. Ihr Administrator hat Ihnen Ihre Anmeldeinformationen zur Verfügung gestellt.

Ich möchte Personen außerhalb meiner Umgebung zulassen AWS-Konto um auf meine Amazon Polly Polly-Ressourcen zuzugreifen

Sie können eine Rolle erstellen, mit der Benutzer anderer Konten oder Personen außerhalb Ihrer Organisation auf Ihre Ressourcen zugreifen können. Sie können festlegen, wem die Übernahme der Rolle anvertraut wird. Im Fall von Diensten, die ressourcenbasierte Richtlinien oder Zugriffskontrolllisten (Access Control Lists, ACLs) verwenden, können Sie diese Richtlinien verwenden, um Personen Zugriff auf Ihre Ressourcen zu gewähren.

Weitere Informationen dazu finden Sie hier:

- Informationen darüber, ob Amazon Polly diese Funktionen unterstützt, finden Sie unterSo arbeitet Amazon Polly mit IAM.
- Informationen dazu, wie Sie Zugriff auf Ihre Ressourcen gewähren können, AWS-Konten die Ihnen gehören, finden Sie im IAM-Benutzerhandbuch unter Gewähren des Zugriffs auf einen IAM-Benutzer in einem anderen AWS-Konto , den Sie besitzen.
- Informationen dazu, wie Sie Dritten Zugriff auf Ihre Ressourcen gewähren können AWS-Konten, finden Sie AWS-Konten im IAM-Benutzerhandbuch unter Gewähren des Zugriffs für Dritte.
- Informationen dazu, wie Sie über einen Identitätsverbund Zugriff gewähren, finden Sie unter Gewähren von Zugriff für extern authentifizierte Benutzer (Identitätsverbund) im IAM-Benutzerhandbuch.
- Informationen zum Unterschied zwischen der Verwendung von Rollen und ressourcenbasierten Richtlinien für den kontoübergreifenden Zugriff finden Sie unter Kontoübergreifender Ressourcenzugriff in IAM im IAM-Benutzerhandbuch.

Protokollierung und Überwachung in Amazon Polly

Die Überwachung ist ein wichtiger Bestandteil der Aufrechterhaltung der Zuverlässigkeit, Verfügbarkeit und Leistung Ihrer Amazon Polly Polly-Anwendungen. Um Amazon Polly API-Aufrufe zu überwachen, können Sie verwenden AWS CloudTrail. Verwenden Sie Amazon CloudWatch Logs, um den Status Ihrer Jobs zu überwachen.

- Amazon CloudWatch Alarms — Mithilfe von CloudWatch Alarmen beobachten Sie eine einzelne Metrik über einen von Ihnen festgelegten Zeitraum. Wenn die Metrik einen bestimmten Schwellenwert überschreitet, wird eine Benachrichtigung an ein Thema oder eine AWS Auto Scaling Richtlinie von Amazon Simple Notification Service gesendet. CloudWatchAlarme lösen keine Aktionen aus, wenn sich eine Metrik in einem bestimmten Status befindet. Der Status muss sich stattdessen geändert haben und für eine festgelegte Anzahl an Zeiträumen aufrechterhalten worden sein. Weitere Informationen finden Sie unter Integration CloudWatch mit Amazon Polly.
- CloudTrail Protokolle — CloudTrail enthält eine Aufzeichnung der Aktionen, die von einem Benutzer, einer Rolle oder einem AWS Service in Amazon Polly ausgeführt wurden. Anhand der von gesammelten Informationen können Sie feststellen CloudTrail, welche Anfrage an Amazon Polly gestellt wurde. Sie können auch die IP-Adresse, von der die Anforderung ausging, den Ersteller und den Erstellungszeitpunkt sowie weitere Details bestimmen. Weitere Informationen finden Sie unter Protokollieren von Amazon Polly API-Aufrufen mit AWS CloudTrail.

Konformitätsprüfung für Amazon Polly

Third-party Prüfer bewerten die Sicherheit und Konformität von Amazon Polly im Rahmen mehrerer AWS Compliance-Programme. Hierzu zählen unter anderem SOC, PCI, FedRAMP und HIPAA.

Eine Liste der AWS Services im Rahmen bestimmter Compliance-Programme finden Sie unter AWS Services im Umfang nach Compliance-Programmen AWS . Allgemeine Informationen finden Sie unter AWS Compliance-Programme AWS .

Sie können Prüfberichte von Drittanbietern unter herunterladen AWS Artifact. Weitere Informationen finden Sie unter Berichte herunterladen in AWS Artifact.

Ihre Compliance-Verantwortung bei der Nutzung von Amazon Polly hängt von der Sensibilität Ihrer Daten, den Compliance-Zielen Ihres Unternehmens und den geltenden Gesetzen und Vorschriften ab. AWS bietet die folgenden Ressourcen zur Unterstützung bei der Einhaltung von Vorschriften:

- Schnellstartanleitungen für Sicherheit und Compliance – In diesen Bereitstellungsleitfäden werden architektonische Überlegungen erörtert und Schritte für die Bereitstellung von sicherheits- und konformitätsorientierten Basisumgebungen auf AWS angegeben.
- Whitepaper „Architecting for HIPAA Security and Compliance“ — In diesem Whitepaper wird beschrieben, wie Unternehmen Anwendungen entwickeln können. AWS HIPAA-compliant
- AWS Compliance-Ressourcen — Diese Sammlung von Arbeitsmappen und Leitfäden kann auf Ihre Branche und Ihren Standort zutreffen.
- Bewertung von Ressourcen anhand von Regeln im AWS Config Entwicklerhandbuch — Der AWS Config Service bewertet, wie gut Ihre Ressourcenkonfigurationen den internen Praktiken, Branchenrichtlinien und Vorschriften entsprechen.
- AWS Security Hub CSPM— Dieser AWS Service bietet einen umfassenden Überblick über Ihren Sicherheitsstatus, sodass Sie überprüfen können AWS , ob Sie die Sicherheitsstandards und Best Practices der Branche einhalten.

Resilienz bei Amazon Polly

Die AWS globale Infrastruktur basiert auf AWS Regionen und Availability Zones. AWS Regionen bieten mehrere physisch getrennte und isolierte Availability Zones, die über Netzwerke mit niedriger Latenz, hohem Durchsatz und hoher Redundanz miteinander verbunden sind. Mithilfe von Availability Zones können Sie Anwendungen und Datenbanken erstellen und ausführen, die automatisch

Failover zwischen Zonen ausführen, ohne dass es zu Unterbrechungen kommt. Availability Zones sind besser verfügbar, fehlertoleranter und skalierbarer als herkömmliche Infrastrukturen mit einem oder mehreren Rechenzentren.

Weitere Informationen zu AWS Regionen und Availability Zones finden Sie unter AWS Globale Infrastruktur.

Infrastruktursicherheit in Amazon Polly

Als verwalteter Service ist Amazon Polly durch die AWS globalen Netzwerksicherheitsverfahren geschützt, die im Whitepaper Amazon Web Services: Sicherheitsprozesse im Überblick beschrieben sind.

Sie verwenden AWS veröffentlichte API-Aufrufe, um über das Netzwerk auf Amazon Polly zuzugreifen. Kunden müssen Transport Layer Security (TLS) 1.0 oder neuer unterstützen. Wir empfehlen TLS 1.2 oder neuer. Kunden müssen außerdem Cipher Suites mit Perfect Forward Secrecy (PFS) wie Ephemeral (DHE) oder Elliptic Curve Ephemeral Diffie-Hellman (ECDHE) unterstützen. Diffie-Hellman Die meisten modernen Systemen wie Java 7 und höher unterstützen diese Modi.

Außerdem müssen Anforderungen mit einer Zugriffsschlüssel-ID und einem geheimen Zugriffsschlüssel signiert sein, der einem IAM-Prinzipal zugeordnet ist. Alternativ können Sie mit AWS -Security-Token-Service (AWS STS) temporäre Sicherheitsanmeldeinformationen erstellen, um die Anforderungen zu signieren.

Bewährte Sicherheitsmethoden für Amazon Polly

Ihr Vertrauen, Ihre Privatsphäre und die Sicherheit Ihrer Inhalte sind unsere obersten Prioritäten. Wir implementieren verantwortungsvolle und ausgeklügelte technische und physische Kontrollen, die den unbefugten Zugriff auf Ihre Inhalte oder deren Offenlegung verhindern und sicherstellen, dass unsere Nutzung unseren Verpflichtungen gegenüber Ihnen entspricht. Weitere Informationen finden Sie in den AWS Häufig gestellten Fragen zum Datenschutz.

Amazon Polly speichert den Inhalt von Texteinsendungen nicht.

Einen umfassenden Überblick über das Thema AWS Sicherheit, einschließlich Compliance, Penetrationstests, Bulletins und Ressourcen, finden Sie auf der AWS Cloud Security-Website.

Verwenden von Amazon Polly mit VPC-Endpunkten mit Schnittstellen

Wenn Sie Amazon Virtual Private Cloud (Amazon VPC) zum Hosten Ihrer AWS Ressourcen verwenden, können Sie eine private Verbindung zwischen Ihrer VPC und Amazon Polly herstellen. Sie können diese Verbindung verwenden, um Sprache mit Amazon Polly zu synthetisieren, ohne das öffentliche Internet zu nutzen.

Amazon VPC ist ein AWS Service, mit dem Sie AWS Ressourcen in einem von Ihnen definierten virtuellen Netzwerk starten können. Mit einer VPC haben Sie die Kontrolle über Ihre Netzwerkeinstellungen, wie IP-Adressbereich, Subnetze, Routing-Tabellen und Netzwerk-Gateways. Um Ihre VPC mit Amazon Polly zu verbinden, definieren Sie einen VPC-Schnittstellen-Endpunkt für Amazon Polly. Mit diesem Endpunkttyp können Sie eine Verbindung zu Ihrer VPC herstellen. AWS-Services Der Endpunkt bietet zuverlässige, skalierbare Konnektivität zu Amazon Polly, ohne dass ein Internet-Gateway, eine NAT-Instance (Network Address Translation) oder eine VPN-Verbindung erforderlich ist. Weitere Informationen finden Sie unter Was ist Amazon VPC im Amazon VPC-Benutzerhandbuch.

Schnittstelle, auf der VPC-Endpunkte basieren AWS PrivateLink, eine AWS Technologie, die private Kommunikation zwischen Benutzern AWS-Services über eine elastic network interface mit privaten IP-Adressen ermöglicht. Weitere Informationen finden Sie unter Neu — AWS PrivateLink für. AWS-Services

Die folgenden Schritte sind für Benutzer von Amazon VPC vorgesehen. Weitere Informationen finden Sie unter Erste Schritte im Amazon VPC-Benutzerhandbuch.

Verfügbarkeit

VPC-Endpunkte werden in allen Regionen unterstützt, in denen Amazon Polly unterstützt wird. Weitere Informationen zu AWS Regionen und Availability Zones finden Sie unter AWS Globale Infrastruktur.

FIPS-Endpunkte sind in den folgenden Regionen verfügbar:

- USA Ost (Nord-Virginia): (us-east-1)
- USA Ost (Ohio): (us-east-2)
- USA West (Nordkalifornien) (us-west-1)
- USA West (Oregon): (us-west-2)

- AWS GovCloud (US-West) (us-gov-west-1)
- Kanada (Zentral): (ca-central-1)

Die FIPS-Endpunkte haben die Form. `com.amazonaws.REGION.polly-fips`

Einen VPC-Endpunkt für Amazon Polly erstellen

Um Amazon Polly mit Ihrer VPC zu verwenden, erstellen Sie einen VPC-Schnittstellen-Endpunkt für Amazon Polly. Der Service, den Sie wählen können, ist com.amazonaws. ***Region***.polly. Sie müssen keine Einstellungen für Amazon Polly ändern. Weitere Informationen finden Sie unter Creating an Interface Endpoint im Amazon VPC-Benutzerhandbuch.

Testen der Verbindung zwischen Ihrer VPC und Amazon Polly

Nachdem Sie den Endpunkt erstellt haben, können Sie die Verbindung testen.

Um die Verbindung zwischen Ihrer VPC und Ihrem Amazon Polly-Endpunkt zu testen

1. Stellen Sie eine Verbindung mit einer Amazon EC2-Instance in Ihrer VPC her. Informationen zum Herstellen einer Verbindung finden Sie unter Connect to your Linux Instance oder Connecting to your Windows Instance in der Amazon EC2 EC2-Dokumentation.
2. Verwenden Sie in der Instance die verfügbaren Amazon AWS CLI Polly-Stimmen `aws polly describe-voices` von der To-Liste.

Wenn die Antwort auf den Befehl die Liste der verfügbaren Amazon Polly Polly-Stimmen enthält, war der Befehl erfolgreich und Ihr VPC-Endpunkt funktioniert.

Steuern des Zugriffs auf Ihren Amazon Polly Polly-Endpunkt

Eine VPC-Endpunktrichtlinie ist eine IAM-Ressourcenrichtlinie, die Sie einem Endpunkt beim Erstellen oder Ändern des Endpunkts zuordnen. Wenn Sie einem Endpunkt beim Erstellen keine Richtlinie zuordnen, wird ihm eine Standardrichtlinie mit Vollzugriff auf den Service zugeordnet. IAM-Benutzerrichtlinien oder servicespezifische Richtlinien werden von einer Endpunktrichtlinie nicht überschrieben oder ersetzt. Endpunktrichtlinien steuern unabhängig vom Endpunkt den Zugriff auf den angegebenen Service.

Endpunktrichtlinien müssen im JSON-Format erstellt werden.

Weitere Informationen finden Sie unter Steuerung des Zugriffs auf Services mit VPC-Endpunkten im Amazon VPC Benutzerhandbuch.

Im Folgenden finden Sie ein Beispiel für eine Endpunktrichtlinie für Amazon Polly. Diese Richtlinie ermöglicht es Benutzern, die sich über die VPC mit Amazon Polly verbinden, Stimmen zu beschreiben und Sprache mit Amazon Polly zu synthetisieren, und verhindert, dass sie andere Amazon Polly Polly-Aktionen ausführen.

```
{
  "Statement": [
    {
      "Sid": "SynthesisAndDescribeVoicesOnly",
      "Principal": "*",
      "Action": [
        "polly:DescribeVoices",
        "polly:SynthesizeSpeech"
      ],
      "Effect": "Allow",
      "Resource": "*"
    }
  ]
}
```

So ändern Sie die VPC-Endpunktrichtlinie für Amazon Polly

1. Öffnen Sie die Amazon-VPC-Konsole unter https://console.aws.amazon.com/vpc.
2. Wählen Sie im Navigationsbereich Endpunkte aus.
3. Wenn Sie den Endpunkt für Amazon Polly noch nicht erstellt haben, wählen Sie Endpunkt erstellen. Wählen Sie dann com.amazonaws aus. *Region*.polly und wählen Sie Create Endpoint.
4. Wählen Sie com.amazonaws aus. ***Region***.polly-Endpunkt und wählen Sie in der unteren Hälfte des Bildschirms die Registerkarte Richtlinie aus.
5. Wählen Sie Richtlinie bearbeiten und nehmen Sie die Änderungen an der Richtlinie vor.

Support für VPC-Kontextschlüssel

Amazon Polly unterstützt die Kontextschlüssel `aws:SourceVpc` und die `aws:SourceVpce` Kontextschlüssel, die den Zugriff auf bestimmte VPCs oder bestimmte VPC-Endpunkte einschränken können. Diese Schlüssel funktionieren nur, wenn der Benutzer VPC-Endpunkte verwendet. Weitere Informationen finden Sie im IAM-Benutzerhandbuch unter Für einige Dienste verfügbare Schlüssel.

Protokollieren von Amazon Polly API-Aufrufen mit AWS CloudTrail

Amazon Polly ist in einen Service integriert AWS CloudTrail, der eine Aufzeichnung der Aktionen eines Benutzers, einer Rolle oder eines AWS Dienstes in Amazon Polly bereitstellt. CloudTrail erfasst alle API-Aufrufe für Amazon Polly als Ereignisse. Zu den erfassten Aufrufen gehören Aufrufe von der Amazon Polly Polly-Konsole und Code-Aufrufe der Amazon Polly Polly-API-Operationen. Wenn Sie einen Trail erstellen, können Sie die kontinuierliche Übermittlung von CloudTrail Ereignissen an einen Amazon S3 S3-Bucket aktivieren, einschließlich Ereignissen für Amazon Polly. Wenn Sie keinen Trail konfigurieren, können Sie die neuesten Ereignisse trotzdem in der CloudTrail Konsole im Ereignisverlauf anzeigen. Anhand der von gesammelten Informationen können Sie die Anfrage CloudTrail, die an Amazon Polly gestellt wurde, die IP-Adresse, von der aus die Anfrage gestellt wurde, wer die Anfrage gestellt hat, wann sie gestellt wurde, und weitere Details ermitteln.

Weitere Informationen darüber CloudTrail, einschließlich der Konfiguration und Aktivierung, finden Sie im AWS CloudTrail Benutzerhandbuch.

Informationen zu Amazon Polly in CloudTrail

CloudTrail ist für Ihr AWS Konto aktiviert, wenn Sie das Konto erstellen. Wenn unterstützte Ereignisaktivitäten in Amazon Polly auftreten, wird diese Aktivität zusammen mit anderen AWS Serviceereignissen in der CloudTrail Ereignishistorie in einem Ereignis aufgezeichnet. Sie können aktuelle Ereignisse in Ihrem AWS Konto ansehen, suchen und herunterladen. Weitere Informationen finden Sie unter Ereignisse mit CloudTrail Ereignisverlauf anzeigen.

Für eine fortlaufende Aufzeichnung der Ereignisse in Ihrem AWS Konto, einschließlich der Ereignisse für Amazon Polly, erstellen Sie einen Trail. Ein Trail ermöglicht CloudTrail die Übermittlung von Protokolldateien an einen Amazon S3 S3-Bucket. Wenn Sie einen Trail in der Konsole erstellen, gilt der Trail standardmäßig für alle AWS Regionen. Der Trail protokolliert Ereignisse aus allen Regionen der AWS Partition und übermittelt die Protokolldateien an den von Ihnen angegebenen Amazon S3 S3-Bucket. Darüber hinaus können Sie andere AWS Dienste konfigurieren, um die in den CloudTrail Protokollen gesammelten Ereignisdaten weiter zu analysieren und darauf zu reagieren. Weitere Informationen finden Sie hier:

- Übersicht zum Erstellen eines Trails
- CloudTrail Unterstützte Dienste und Integrationen

- Konfiguration von Amazon SNS SNS-Benachrichtigungen für CloudTrail
- Empfangen von CloudTrail Protokolldateien aus mehreren Regionen und Empfangen von CloudTrail Protokolldateien von mehreren Konten

Amazon Polly unterstützt die Protokollierung der folgenden Aktionen als Ereignisse in CloudTrail Protokolldateien:

- DeleteLexicon
- DescribeVoices
- GetLexicon
- GetSpeechSynthesisTask
- ListLexicons
- ListSpeechSynthesisTasks
- PutLexicon
- StartSpeechSynthesisTask
- SynthesizeSpeech

Jeder Ereignis- oder Protokolleintrag enthält Informationen zu dem Benutzer, der die Anforderung generiert hat. Die Identitätsinformationen unterstützen Sie bei der Ermittlung der folgenden Punkte:

- Ob die Anfrage mit Root-Benutzer- oder AWS Identity and Access Management (IAM-) Benutzeranmeldedaten gestellt wurde.
- Gibt an, ob die Anforderung mit temporären Sicherheitsanmeldeinformationen für eine Rolle oder einen Verbundbenutzer gesendet wurde.
- Ob die Anfrage von einem anderen AWS Dienst gestellt wurde.

Weitere Informationen finden Sie unter CloudTrail userIdentity-Element.

Beispiel: Amazon Polly Polly-Protokolldateieinträge

Ein Trail ist eine Konfiguration, die die Übertragung von Ereignissen als Protokolldateien an einen von Ihnen angegebenen Amazon S3 S3-Bucket ermöglicht. CloudTrail Protokolldateien enthalten einen oder mehrere Protokolleinträge. Ein Ereignis stellt eine einzelne Anforderung aus einer beliebigen Quelle dar und enthält Informationen über die angeforderte Aktion, Datum und Uhrzeit der Aktion,

Anforderungsparameter usw. CloudTrail Protokolldateien sind kein geordneter Stack-Trace der öffentlichen API-Aufrufe, sodass sie nicht in einer bestimmten Reihenfolge angezeigt werden.

Das folgende Beispiel zeigt einen CloudTrail Protokolleintrag, der das demonstriert`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"
            }
        }

    ]
```

```
}
```

Integration CloudWatch mit Amazon Polly

Wenn Sie mit Amazon Polly interagieren, sendet Amazon Polly die folgenden Metriken und Dimensionen an CloudWatch jede Minute. Sie können die folgenden Verfahren verwenden, um die Metriken für Amazon Polly anzuzeigen.

Sie können Amazon Polly mithilfe von Amazon Polly überwachen CloudWatch, das Rohdaten von Amazon Polly sammelt und zu lesbaren Kennzahlen nahezu in Echtzeit verarbeitet. Diese Statistiken werden für einen Zeitraum von zwei Wochen aufgezeichnet, damit Sie auf `historical information` zugreifen können und einen besseren Überblick darüber erhalten, wie Ihre Webanwendung oder der Service ausgeführt werden. Standardmäßig werden Amazon Polly-Metrikdaten CloudWatch in Intervallen von 1 Minute gesendet. Weitere Informationen finden Sie unter Was ist Amazon CloudWatch im CloudWatch Amazon-Benutzerhandbuch.

CloudWatch Metriken abrufen (Konsole)

1. Öffnen Sie die CloudWatch Konsole unter https://console.aws.amazon.com/cloudwatch/.
2. Wählen Sie im Navigationsbereich Metriken aus.
3. Wählen Sie im Bereich CloudWatch Metriken nach Kategorie unter der Metrikkategorie für Amazon Polly eine Metrikkategorie aus, und scrollen Sie dann im oberen Bereich nach unten, um die vollständige Liste der Metriken anzuzeigen.

CloudWatch Metriken abrufen auf der AWS CLI

Der folgende Code zeigt die verfügbaren Messwerte für Amazon Polly an.

```
aws cloudwatch list-metrics --namespace "AWS/Polly"
```

Der vorherige Befehl gibt eine Liste von Amazon Polly-Metriken zurück, die der folgenden ähnelt. Das Element `MetricName` gibt an, worum es sich bei der Metrik handelt.

```
{
    "Metrics": [
        {
            "Namespace": "AWS/Polly",
            "Dimensions": [
```

```
                {
                    "Name": "Operation",
                    "Value": "SynthesizeSpeech"
                }
            ],
            "MetricName": "ResponseLatency"
        },
        {
            "Namespace": "AWS/Polly",
            "Dimensions": [
                {
                    "Name": "Operation",
                    "Value": "SynthesizeSpeech"
                }
            ],
            "MetricName": "RequestCharacters"
        }
```

Weitere Informationen finden Sie GetMetricStatisticsin der Amazon CloudWatch API-Referenz.

Amazon Polly Polly-Metriken

Amazon Polly erstellt für jede Anfrage die folgenden Metriken. Diese Metriken werden aggregiert und in Intervallen von einer Minute CloudWatch dorthin gesendet, wo sie verfügbar sind.

Metrik	Beschreibung
`RequestCharacters`	Die Anzahl der Zeichen in der Anfrage. Dies gilt nur für gebührenpflichtige Zeichen und schließt keine SSML-Tags ein. Gültige Dimension: Vorgang Gültige Statistiken: Minimum, Maximum, Durchschnitt SampleCount, Summe Einheit: Anzahl
`ResponseLatency`	Die Latenz zwischen dem Zeitpunkt, an dem die Anfrage gestellt wurde, und dem Start der Streaming -Antwort.

Metrik	Beschreibung
	Gültige Abmessungen: Betrieb Gültige Statistiken: Minimum, Maximum, Durchschnitt, SampleCount Einheit: Mikrosekunden
2XXCount	Bei einer erfolgreichen Antwort wurde ein HTTP 200-Level-Code zurückgegeben. Gültige Abmessungen: Betrieb Gültige Statistiken: Durchschnitt SampleCount, Summe Einheit: Anzahl
4XXCount	Bei einem Fehler wurde ein Fehlercode auf HTTP-Ebene 400 zurückgegeben. Für jede erfolgreiche Antwort wird eine Null (0) ausgegeben. Gültige Abmessungen: Betrieb Gültige Statistiken: Durchschnitt SampleCount, Summe Einheit: Anzahl
5XXCount	Bei einem Fehler wurde ein Fehlercode auf HTTP-Ebene 500 zurückgegeben. Für jede erfolgreiche Antwort wird eine Null (0) ausgegeben. Gültige Abmessungen: Betrieb Gültige Statistiken: Durchschnitt SampleCount, Summe Einheit: Anzahl

Dimensionen für Amazon Polly Metrics

Amazon Polly Polly-Metriken verwenden den AWS/Polly-Namespace und stellen Metriken für die folgende Dimension bereit:

Dimension	Beschreibung
`Operation`	Metriken werden nach der API-Methode gruppiert , auf die sie sich beziehen. Mögliche Werte sind `SynthesizeSpeech` `PutLexicon` `DescribeVoices` ,, usw.

Amazon Polly API-Referenz

Amazon Polly ist ein Webservice, der es einfach macht, Sprache aus Text zu synthetisieren.

Der Amazon Polly Polly-Service bietet API-Operationen für die Synthese hochwertiger Sprache aus Klartext und Speech Synthesis Markup Language (SSML) sowie die Verwaltung von Aussprachelexika, mit denen Sie die besten Ergebnisse für Ihre Anwendungsdomäne erzielen können.

Authentifizierte API-Aufrufe müssen mithilfe dem Signature Version 4-Signaturprozess signiert werden. Weitere Informationen finden Sie unter Signieren von API-Anfragen in der. AWSAllgemeine Amazon Web Services-Referenz

Themen

Aktionen

Folgende Aktionen werden unterstützt:

DeleteLexicon

Löscht das angegebene Aussprachelexikon, das in einem gespeichert ist. AWS-Region Ein gelöschtes Lexikon ist nicht für die Sprachsynthese verfügbar, und es ist auch nicht möglich, es mit dem Oder abzurufen. `GetLexicon` `ListLexicon` APIs

Weitere Informationen finden Sie unter Lexika verwalten.

Anforderungssyntax

```
DELETE /v1/lexicons/LexiconName HTTP/1.1
```

URI-Anfrageparameter

Die Anforderung verwendet die folgenden URI-Parameter.

LexiconName

Der Name des zu löschenden Lexikons. Es muss sich um ein vorhandenes Lexikon in der Region handeln.

Pattern: `[0-9A-Za-z]{1,20}`

Erforderlich: Ja

Anforderungstext

Der Anforderung besitzt keinen Anforderungstext.

Antwortsyntax

```
HTTP/1.1 200
```

Antwortelemente

Wenn die Aktion erfolgreich ist, gibt der Dienst eine HTTP 200-Antwort mit leerem HTTP-Textinhalt zurück.

Fehler

LexiconNotFoundException

Amazon Polly kann das angegebene Lexikon nicht finden. Dies kann durch ein fehlendes Lexikon, durch einen falsch geschriebenen Namen oder durch die Angabe eines Lexikons in einer anderen Region verursacht werden.

Vergewissern Sie sich, dass das Lexikon existiert, sich in der Region befindet (siehe ListLexicons) und ob Sie den Namen richtig geschrieben haben. Versuchen Sie es dann erneut.

HTTP-Statuscode: 404

ServiceFailureException

Ein unbekannter Zustand hat zu einem Dienstausfall geführt.

HTTP Status Code: 500

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einer der sprachspezifischen Sprachen AWS SDKs finden Sie im Folgenden:

- AWS Befehlszeilenschnittstelle V2
- AWS SDK for .NET V4
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK für JavaScript V3
- AWS SDK für Kotlin
- AWS SDK for PHP V3
- AWS SDK für Python
- AWS SDK for Ruby V3

DescribeVoices

Gibt die Liste der Stimmen zurück, die beim Anfordern von Sprachausgaben verwendet werden können Jede Stimme spricht eine bestimmte Sprache, ist entweder männlich oder weiblich und wird durch eine ID identifiziert, bei der es sich um die ASCII-Version des Sprachnamens handelt.

Bei der Sprachsynthese (`SynthesizeSpeech`) geben Sie die Stimmen-ID für die gewünschte Stimme aus der Liste der zurückgegebenen Stimmen an. `DescribeVoices`

Sie möchten beispielsweise, dass Ihre News-Reader-Anwendung Nachrichten in einer bestimmten Sprache liest, einem Benutzer aber die Möglichkeit gibt, die Stimme auszuwählen. Mithilfe dieser `DescribeVoices` Funktion können Sie dem Benutzer eine Liste verfügbarer Stimmen zur Auswahl zur Verfügung stellen.

Sie können optional einen Sprachcode angeben, um die verfügbaren Stimmen zu filtern. Wenn Sie beispielsweise angeben`en-US`, gibt der Vorgang eine Liste aller verfügbaren Stimmen in US-Englisch zurück.

Diese Operation erfordert zur Ausführung der `polly:DescribeVoices`-Aktion Berechtigungen.

Anforderungssyntax

```
GET /v1/voices?
Engine=Engine&IncludeAdditionalLanguageCodes=IncludeAdditionalLanguageCodes&LanguageCode=Langua
 HTTP/1.1
```

URI-Anfrageparameter

Die Anforderung verwendet die folgenden URI-Parameter.

Engine

Gibt die Engine (`standard`, `long-form` oder`generative`) an`neural`, die von Amazon Polly bei der Verarbeitung von Eingabetext für die Sprachsynthese verwendet wird.

Zulässige Werte: `standard | neural | long-form | generative`

IncludeAdditionalLanguageCodes

Boolescher Wert, der angibt, ob zweisprachige Stimmen zurückgegeben werden sollen, die die angegebene Sprache als zusätzliche Sprache verwenden. Wenn Sie beispielsweise alle

Sprachen anfordern, in denen US-Englisch (es-US) verwendet wird, und es eine italienische Stimme gibt, die sowohl Italienisch (it-IT) als auch US-Englisch spricht, wird diese Stimme aufgenommen, wenn Sie sie angeben, aber nicht, wenn Sie sie angebenyes. no

LanguageCode

Das Sprachkennzeichen (ISO-639-Code für den Sprachnamen — ISO-3166-Ländercode) zum Filtern der Liste der zurückgegebenen Stimmen. Wenn Sie diesen optionalen Parameter nicht angeben, werden alle verfügbaren Stimmen zurückgegeben.

Zulässige Werte: `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 | cs-CZ | de-CH | en-SG`

NextToken

Ein undurchsichtiges Paginierungstoken, das von der vorherigen `DescribeVoices` Operation zurückgegeben wurde. Falls vorhanden, gibt dies an, wo mit der Auflistung fortgefahren werden soll.

Längenbeschränkungen: Minimale Länge von 0. Maximale Länge von 4096.

Anforderungstext

Der Anforderung besitzt keinen Anforderungstext.

Antwortsyntax

```
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" ]
        }
    ]
}
```

Antwortelemente

Wenn die Aktion erfolgreich ist, sendet der Service eine HTTP 200-Antwort zurück.

Die folgenden Daten werden vom Service im JSON-Format zurückgegeben.

NextToken

Das Paginierungstoken, das in der nächsten Anfrage verwendet werden soll, um die Liste der Stimmen fortzusetzen. `NextToken`wird nur zurückgegeben, wenn die Antwort gekürzt ist.

Typ: Zeichenfolge

Längenbeschränkungen: Minimale Länge von 0. Maximale Länge von 4096.

Voices

Eine Liste von Stimmen mit ihren Eigenschaften.

Typ: Array von Voice-Objekten

Fehler

InvalidNextTokenException

Das NextToken ist ungültig. Stellen Sie sicher, dass es richtig geschrieben ist, und versuchen Sie es erneut.

HTTP-Statuscode: 400

ServiceFailureException

Ein unbekannter Zustand hat zu einem Dienstausfall geführt.

HTTP Status Code: 500

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einer der sprachspezifischen Sprachen AWS SDKs finden Sie im Folgenden:

- AWS Befehlszeilenschnittstelle V2
- AWS SDK for .NET V4
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK für JavaScript V3
- AWS SDK für Kotlin
- AWS SDK for PHP V3
- AWS SDK für Python
- AWS SDK for Ruby V3

GetLexicon

Gibt den Inhalt des angegebenen Aussprachelexikons zurück, das in einem AWS-Region gespeichert ist. Weitere Informationen finden Sie unter Lexika verwalten.

Anforderungssyntax

```
GET /v1/lexicons/LexiconName HTTP/1.1
```

URI-Anfrageparameter

Die Anforderung verwendet die folgenden URI-Parameter.

LexiconName

Name des Lexikons.

Pattern: `[0-9A-Za-z]{1,20}`

Erforderlich: Ja

Anforderungstext

Der Anforderung besitzt keinen Anforderungstext.

Antwortsyntax

```
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
    }
}
```

Antwortelemente

Wenn die Aktion erfolgreich ist, sendet der Service eine HTTP 200-Antwort zurück.

Die folgenden Daten werden vom Service im JSON-Format zurückgegeben.

Lexicon

Lexikon-Objekt, das den Namen und den Zeichenketteninhalt des Lexikons bereitstellt.

Typ: Lexicon Objekt

LexiconAttributes

Metadaten des Lexikons, einschließlich verwendeter phonetischer Alphabetik, Sprachcode, Lexikon-ARN, Anzahl der im Lexikon definierten Lexeme und Größe des Lexikons in Byte.

Typ: LexiconAttributes Objekt

Fehler

LexiconNotFoundException

Amazon Polly kann das angegebene Lexikon nicht finden. Dies kann durch ein fehlendes Lexikon, durch einen falsch geschriebenen Namen oder durch die Angabe eines Lexikons in einer anderen Region verursacht werden.

Vergewissern Sie sich, dass das Lexikon existiert, sich in der Region befindet (siehe ListLexicons) und ob Sie den Namen richtig geschrieben haben. Versuchen Sie es dann erneut.

HTTP-Statuscode: 404

ServiceFailureException

Ein unbekannter Zustand hat zu einem Dienstausfall geführt.

HTTP Status Code: 500

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einer der sprachspezifischen Sprachen AWS SDKs finden Sie im Folgenden:

- AWS Befehlszeilenschnittstelle V2
- AWS SDK for .NET V4
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK für JavaScript V3
- AWS SDK für Kotlin
- AWS SDK for PHP V3
- AWS SDK für Python
- AWS SDK for Ruby V3

GetSpeechSynthesisTask

Ruft ein bestimmtes SpeechSynthesisTask Objekt basierend auf seiner TaskID ab. Dieses Objekt enthält Informationen über die angegebene Sprachsynthese-Aufgabe, einschließlich des Status der Aufgabe, und einen Link zum S3-Bucket, der die Ausgabe der Aufgabe enthält.

Anforderungssyntax

```
GET /v1/synthesisTasks/TaskId HTTP/1.1
```

URI-Anfrageparameter

Die Anforderung verwendet die folgenden URI-Parameter.

TaskId

Die von Amazon Polly generierte Kennung für eine Sprachsyntheseaufgabe.

Pattern: `^[a-zA-Z0-9_-]{1,100}$`

Erforderlich: Ja

Anforderungstext

Der Anforderung besitzt keinen Anforderungstext.

Antwortsyntax

```
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"
   }
}
```

Antwortelemente

Wenn die Aktion erfolgreich ist, sendet der Service eine HTTP 200-Antwort zurück.

Die folgenden Daten werden vom Service im JSON-Format zurückgegeben.

SynthesisTask

SynthesisTask Objekt, das Informationen zur angeforderten Aufgabe bereitstellt, einschließlich Ausgabeformat, Erstellungszeit, Aufgabenstatus usw.

Typ: SynthesisTask Objekt

Fehler

InvalidTaskIdException

Die angegebene Aufgaben-ID ist nicht gültig. Bitte geben Sie eine gültige Task-ID ein und versuchen Sie es erneut.

HTTP-Statuscode: 400

ServiceFailureException

Ein unbekannter Zustand hat zu einem Dienstausfall geführt.

HTTP Status Code: 500

SynthesisTaskNotFoundException

Die Sprachsynthese-Aufgabe mit der angeforderten Task-ID wurde nicht gefunden.

HTTP-Statuscode: 400

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einer der sprachspezifischen Sprachen AWS SDKs finden Sie im Folgenden:

- AWS Befehlszeilenschnittstelle V2
- AWS SDK for .NET V4
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK für JavaScript V3
- AWS SDK für Kotlin
- AWS SDK for PHP V3
- AWS SDK für Python
- AWS SDK for Ruby V3

ListLexicons

Gibt eine Liste von Aussprachelexika zurück, die in einem AWS-Region gespeichert sind. Weitere Informationen finden Sie unter Lexika verwalten.

Anforderungssyntax

```
GET /v1/lexicons?NextToken=NextToken HTTP/1.1
```

URI-Anfrageparameter

Die Anforderung verwendet die folgenden URI-Parameter.

NextToken

Ein undurchsichtiges Paginierungstoken, das von einem vorherigen Vorgang zurückgegeben wurde. `ListLexicons` Falls vorhanden, gibt dies an, wo die Liste der Lexika fortgesetzt werden soll.

Längenbeschränkungen: Minimale Länge von 0. Maximale Länge von 4096.

Anforderungstext

Der Anforderung besitzt keinen Anforderungstext.

Antwortsyntax

```
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"
}
```

Antwortelemente

Wenn die Aktion erfolgreich ist, sendet der Service eine HTTP 200-Antwort zurück.

Die folgenden Daten werden vom Service im JSON-Format zurückgegeben.

Lexicons

Eine Liste von Lexikonnamen und Attributen.

Typ: Array von LexiconDescription-Objekten

NextToken

Das Paginierungstoken, das in der nächsten Anfrage verwendet werden soll, um die Auflistung der Lexika fortzusetzen. `NextToken`wird nur zurückgegeben, wenn die Antwort gekürzt ist.

Typ: Zeichenfolge

Längenbeschränkungen: Minimale Länge von 0. Maximale Länge von 4096.

Fehler

InvalidNextTokenException

Das NextToken ist ungültig. Stellen Sie sicher, dass es richtig geschrieben ist, und versuchen Sie es erneut.

HTTP-Statuscode: 400

ServiceFailureException

Ein unbekannter Zustand hat einen Dienstausfall verursacht.

HTTP Status Code: 500

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einer der sprachspezifischen Sprachen AWS SDKs finden Sie im Folgenden:

- AWS Befehlszeilenschnittstelle V2
- AWS SDK for .NET V4
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK für JavaScript V3
- AWS SDK für Kotlin
- AWS SDK for PHP V3
- AWS SDK für Python
- AWS SDK for Ruby V3

ListSpeechSynthesisTasks

Gibt eine Liste von SpeechSynthesisTask Objekten zurück, die nach ihrem Erstellungsdatum sortiert sind. Dieser Vorgang kann die Aufgaben nach ihrem Status filtern, sodass Benutzer beispielsweise nur Aufgaben auflisten können, die abgeschlossen sind.

Anforderungssyntax

```
GET /v1/synthesisTasks?MaxResults=MaxResults&NextToken=NextToken&Status=Status HTTP/1.1
```

URI-Anfrageparameter

Die Anforderung verwendet die folgenden URI-Parameter.

MaxResults

Maximale Anzahl von Sprachsyntheseaufgaben, die bei einem Listenvorgang zurückgegeben wurden.

Gültiger Bereich: Mindestwert 1. Maximalwert 100.

NextToken

Das Paginierungstoken, das in der nächsten Anforderung verwendet werden soll, um die Auflistung der Sprachsyntheseaufgaben fortzusetzen.

Längenbeschränkungen: Minimale Länge von 0. Maximale Länge von 4096.

Status

Status der Sprachsyntheseaufgaben, die in einem Listenvorgang zurückgegeben wurden

Zulässige Werte: `scheduled | inProgress | completed | failed`

Anforderungstext

Der Anforderung besitzt keinen Anforderungstext.

Antwortsyntax

```
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"
      }
   ]
}
```

Antwortelemente

Wenn die Aktion erfolgreich ist, sendet der Service eine HTTP 200-Antwort zurück.

Die folgenden Daten werden vom Service im JSON-Format zurückgegeben.

NextToken

Ein undurchsichtiges Paginierungstoken, das von der vorherigen List-Operation in dieser Anforderung zurückgegeben wurde. Falls vorhanden, gibt es an, wo die Auflistung fortgesetzt werden soll.

Typ: Zeichenfolge

Längenbeschränkungen: Minimale Länge von 0. Maximale Länge von 4096.

SynthesisTasks

Liste von SynthesisTask Objekten, die Informationen zur angegebenen Aufgabe in der Listenanforderung bereitstellt, einschließlich Ausgabeformat, Erstellungszeit, Aufgabenstatus usw.

Typ: Array von SynthesisTask-Objekten

Fehler

InvalidNextTokenException

Das NextToken ist ungültig. Stellen Sie sicher, dass es richtig geschrieben ist, und versuchen Sie es erneut.

HTTP-Statuscode: 400

ServiceFailureException

Ein unbekannter Zustand hat einen Dienstausfall verursacht.

HTTP Status Code: 500

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einer der sprachspezifischen Sprachen AWS SDKs finden Sie im Folgenden:

- AWS Befehlszeilenschnittstelle V2
- AWS SDK for .NET V4
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK für JavaScript V3
- AWS SDK für Kotlin
- AWS SDK for PHP V3
- AWS SDK für Python
- AWS SDK for Ruby V3

PutLexicon

Speichert ein Aussprachelexikon in einem AWS-Region. Wenn in der Region bereits ein Lexikon mit demselben Namen existiert, wird es durch das neue Lexikon überschrieben. Lexikonoperationen sind letztlich konsistent, daher kann es einige Zeit dauern, bis das Lexikon für die Operation verfügbar ist. SynthesizeSpeech

Weitere Informationen finden Sie unter Lexika verwalten.

Anforderungssyntax

```
PUT /v1/lexicons/LexiconName HTTP/1.1
Content-type: application/json

{
   "Content": "string"
}
```

URI-Anfrageparameter

Die Anforderung verwendet die folgenden URI-Parameter.

LexiconName

Name des Lexikons. Der Name muss dem regulären Expressformat [0-9a-Za-Z] {1,20} entsprechen. Das heißt, der Name ist eine alphanumerische Zeichenfolge mit bis zu 20 Zeichen, bei der Groß- und Kleinschreibung beachtet wird.

Pattern: `[0-9A-Za-z]{1,20}`

Erforderlich: Ja

Anforderungstext

Die Anforderung akzeptiert die folgenden Daten im JSON-Format.

Content

Inhalt des PLS-Lexikons als Zeichenkettendaten.

Typ: Zeichenfolge

Erforderlich: Ja

Antwortsyntax

```
HTTP/1.1 200
```

Antwortelemente

Wenn die Aktion erfolgreich ist, gibt der Dienst eine HTTP 200-Antwort mit leerem HTTP-Textinhalt zurück.

Fehler

InvalidLexiconException

Amazon Polly kann das angegebene Lexikon nicht finden. Stellen Sie sicher, dass der Name des Lexikons richtig geschrieben ist, und versuchen Sie es erneut.

HTTP-Statuscode: 400

LexiconSizeExceededException

Die maximale Größe des angegebenen Lexikons würde durch diesen Vorgang überschritten werden.

HTTP-Statuscode: 400

MaxLexemeLengthExceededException

Die maximale Größe des Lexems würde durch diesen Vorgang überschritten werden.

HTTP-Statuscode: 400

MaxLexiconsNumberExceededException

Die maximale Anzahl von Lexika würde durch diesen Vorgang überschritten werden.

HTTP-Statuscode: 400

ServiceFailureException

Ein unbekannter Zustand hat einen Dienstausfall verursacht.

HTTP Status Code: 500

UnsupportedPlsAlphabetException

Das im Lexikon angegebene Alphabet wird nicht unterstützt. Gültige Werte sind `x-sampa` und `ipa`.

HTTP-Statuscode: 400

UnsupportedPlsLanguageException

Die im Lexikon angegebene Sprache wird nicht unterstützt. Eine Liste der unterstützten Sprachen finden Sie unter Lexikonattribute.

HTTP-Statuscode: 400

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einer der sprachspezifischen Sprachen finden Sie im AWS SDKs Folgenden:

- AWS Befehlszeilenschnittstelle V2
- AWS SDK for .NET V4
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK für JavaScript V3
- AWS SDK für Kotlin
- AWS SDK for PHP V3
- AWS SDK für Python
- AWS SDK for Ruby V3

StartSpeechSynthesisStream

Synthetisiert UTF-8-Eingabe, Klartext oder SSML über eine bidirektionale Streaming-Verbindung. Geben Sie Syntheseparameter in HTTP/2-Headern an, senden Sie Text inkrementell als Ereignisse im Eingabestream und empfangen Sie synthetisiertes Audio, sobald es verfügbar ist.

Diese Operation dient als bidirektionales Gegenstück zu: `SynthesizeSpeech`

- SynthesizeSpeech

Anforderungssyntax

```
POST /v1/synthesisStream HTTP/1.1
x-amzn-Engine: Engine
x-amzn-LanguageCode: LanguageCode
x-amzn-LexiconNames: LexiconNames
x-amzn-OutputFormat: OutputFormat
x-amzn-SampleRate: SampleRate
x-amzn-VoiceId: VoiceId
Content-type: application/json

{
   "CloseStreamEvent": {
   },
   "TextEvent": {
      "FlushStreamConfiguration": {
         "Force": boolean
      },
      "Text": "string",
      "TextType": "string"
   }
}
```

URI-Anfrageparameter

Die Anforderung verwendet die folgenden URI-Parameter.

Engine

Gibt die Engine an, die Amazon Polly bei der Verarbeitung von Eingabetext für die Sprachsynthese verwenden soll. Derzeit wird nur die `generative` Engine unterstützt. Wenn Sie

eine Stimme angeben, die von der ausgewählten Engine nicht unterstützt wird, gibt Amazon Polly einen Fehler zurück.

Zulässige Werte: `standard | neural | long-form | generative`

Erforderlich: Ja

LanguageCode

Ein optionaler Parameter, der den Sprachcode für die Sprachsyntheseanforderung festlegt. Geben Sie diesen Parameter nur an, wenn Sie eine zweisprachige Stimme verwenden. Wenn eine zweisprachige Stimme verwendet wird und kein Sprachcode angegeben ist, verwendet Amazon Polly die Standardsprache der zweisprachigen Stimme.

Zulässige Werte: `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 | cs-CZ | de-CH | en-SG`

LexiconNames

Die Namen eines oder mehrerer Aussprachelexika, die der Service bei der Synthese anwenden soll. Amazon Polly wendet Lexika nur an, wenn die Lexikonsprache mit der Sprachsprache übereinstimmt.

Array-Mitglieder: Maximale Anzahl von 5 Elementen.

Pattern: `[0-9A-Za-z]{1,20}`

OutputFormat

Das Audioformat für die synthetisierte Sprache. Derzeit unterstützt Amazon Polly keine JSON-Sprachmarken.

Zulässige Werte: `json | mp3 | ogg_opus | ogg_vorbis | pcm | mulaw | alaw`

Erforderlich: Ja

SampleRate

Die Audiofrequenz, angegeben in Hz.

VoiceId

Die Stimme, die für die Synthese verwendet werden soll. Verwenden Sie den DescribeVoicesVorgang IDs, um eine Liste der verfügbaren Sprachbefehle abzurufen.

Zulässige Werte: `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 | Jitka | Sabrina | Jasmine | Jihye | Ambre | Beatrice | Florian | Lennart | Lorenzo | Tiffany`

Erforderlich: Ja

Anforderungstext

Die Anforderung akzeptiert die folgenden Daten im JSON-Format.

CloseStreamEvent

Ein Ereignis, das das Ende des Eingabestreams anzeigt.

Typ: CloseStreamEvent Objekt

Erforderlich: Nein

TextEvent

Ein Textereignis, das Inhalte enthält, die synthetisiert werden sollen.

Typ: TextEvent Objekt

Erforderlich: Nein

Antwortsyntax

```
HTTP/1.1 200
Content-type: application/json

{
   "AudioEvent": {
      "AudioChunk": blob
   },
   "ServiceFailureException": {
   },
   "ServiceQuotaExceededException": {
   },
   "StreamClosedEvent": {
      "RequestCharacters": number
   },
   "ThrottlingException": {
   },
   "ValidationException": {
   }
}
```

Antwortelemente

Wenn die Aktion erfolgreich ist, sendet der Service eine HTTP 200-Antwort zurück.

Die folgenden Daten werden vom Service im JSON-Format zurückgegeben.

AudioEvent

Ein Audioereignis, das synthetisierte Sprache enthält.

Typ: AudioEvent Objekt

ServiceFailureException

Ein unbekannter Zustand hat einen Dienstausfall verursacht.

Typ: Ausnahme

HTTP Status Code: 500

ServiceQuotaExceededException

Eine Ausnahme, die darauf hinweist, dass ein Dienstkontingent überschritten würde.

Typ: Ausnahme

HTTP-Statuscode: 402

StreamClosedEvent

Ein Ereignis mit zusammenfassenden Informationen, das darauf hinweist, dass der Stream geschlossen wurde.

Typ: StreamClosedEvent Objekt

ThrottlingException

Eine Ausnahme, die darauf hinweist, dass die Anfrage gedrosselt wurde.

Typ: Ausnahme

HTTP-Statuscode: 400

ValidationException

Eine Ausnahme, die darauf hinweist, dass die Eingabe nicht validiert wurde.

Typ: Ausnahme

HTTP-Statuscode: 400

Fehler

ServiceFailureException

Ein unbekannter Zustand hat zu einem Dienstausfall geführt.

HTTP Status Code: 500

ServiceQuotaExceededException

Die Anfrage würde dazu führen, dass ein Dienstkontingent überschritten wird.

quotaCode

Der Kontingentcode, der das spezifische Kontingent identifiziert.

serviceCode

Der Dienstcode, der den ursprünglichen Dienst identifiziert.

HTTP-Statuscode: 402

ThrottlingException

Die Anfrage wurde aufgrund der Anforderungsdrosselung abgelehnt.

throttlingReasons

Eine Liste von Gründen, die erklären, warum die Anfrage gedrosselt wurde.

HTTP-Statuscode: 400

ValidationException

Die Eingabe erfüllt die vom Dienst angegebenen Einschränkungen nicht.

fields

Die Felder, die den Validierungsfehler verursacht haben.

reason

Der Grund, warum die Anfrage nicht validiert werden konnte.

HTTP-Statuscode: 400

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einer der sprachspezifischen Sprachen AWS SDKs finden Sie im Folgenden:

- AWS Befehlszeilenschnittstelle V2
- AWS SDK for .NET V4
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK für JavaScript V3
- AWS SDK für Kotlin
- AWS SDK for PHP V3
- AWS SDK für Python
- AWS SDK for Ruby V3

StartSpeechSynthesisTask

Ermöglicht die Erstellung einer asynchronen Synthesetask, indem eine neue `SpeechSynthesisTask` gestartet wird. Dieser Vorgang erfordert alle Standardinformationen, die für die Sprachsynthese benötigt werden, sowie den Namen eines Amazon S3 S3-Buckets, in dem der Service die Ausgabe der Synthese-Aufgabe speichert, und zwei optionale Parameter (`OutputS3KeyPrefix`und`SnsTopicArn`). Sobald die Syntheseaufgabe erstellt wurde, gibt dieser Vorgang ein `SpeechSynthesisTask` Objekt zurück, das eine Kennung dieser Aufgabe sowie den aktuellen Status enthält. Das `SpeechSynthesisTask` Objekt ist nach dem Start der asynchronen Syntheseaufgabe 72 Stunden lang verfügbar.

Anforderungssyntax

```
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-Anfrageparameter

Die Anforderung verwendet keine URI-Parameter.

Anforderungstext

Die Anforderung akzeptiert die folgenden Daten im JSON-Format.

Engine

Gibt die Engine (`standard`, `long-form` oder`generative`) an`neural`, die Amazon Polly bei der Verarbeitung von Eingabetext für die Sprachsynthese verwenden soll. Die Verwendung einer Stimme, die für die gewählte Engine nicht unterstützt wird, führt zu einem Fehler.

Typ: Zeichenfolge

Zulässige Werte: `standard | neural | long-form | generative`

Erforderlich: Nein

LanguageCode

Optionaler Sprachcode für die Sprachsynthese-Anfrage. Dies ist nur erforderlich, wenn Sie eine zweisprachige Stimme wie Aditi verwenden, die entweder für indisches Englisch (en-IN) oder Hindi (Hi-in) verwendet werden kann.

Wenn eine zweisprachige Stimme verwendet wird und kein Sprachcode angegeben ist, verwendet Amazon Polly die Standardsprache der zweisprachigen Stimme. Die Standardsprache für jede Stimme ist die Sprache, die bei der Operation für den DescribeVoicesParameter zurückgegeben wurde. `LanguageCode` Wenn beispielsweise kein Sprachcode angegeben ist, verwendet Aditi indisches Englisch statt Hindi.

Typ: Zeichenfolge

Zulässige Werte: `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 | cs-CZ | de-CH | en-SG`

Erforderlich: Nein

LexiconNames

Liste mit einem oder mehreren Aussprache-Lexikonnamen, die der Dienst bei der Synthese anwenden soll. Lexika werden nur angewendet, wenn die Sprache des Lexikons mit der Sprache der Stimme übereinstimmt.

Typ: Zeichenfolgen-Array

Array-Mitglieder: Maximale Anzahl von 5 Elementen.

Pattern: `[0-9A-Za-z]{1,20}`

Erforderlich: Nein

OutputFormat

Das Format, in dem die zurückgegebene Ausgabe codiert wird. Für Audiostreams ist dies mp3, ogg_vorbis, ogg_opus, mu-law, a-law oder pcm. Bei Sprachzeichen ist dies json.

Typ: Zeichenfolge

Zulässige Werte: `json | mp3 | ogg_opus | ogg_vorbis | pcm | mulaw | alaw`

Erforderlich: Ja

OutputS3BucketName

Name des Amazon S3 S3-Buckets, in dem die Ausgabedatei gespeichert wird.

Typ: Zeichenfolge

Pattern: `^[a-z0-9][\.\-a-z0-9]{1,61}[a-z0-9]$`

Erforderlich: Ja

OutputS3KeyPrefix

Das Amazon S3 S3-Schlüsselpräfix für die Sprachausgabedatei.

Typ: Zeichenfolge

Pattern: `^[0-9a-zA-Z\/\!\-_\.\*\'\(\):;\$@=+\,\?&]{0,800}$`

Erforderlich: Nein

SampleRate

Die in Hz angegebene Audiofrequenz.

Die gültigen Werte für mp3 und ogg_vorbis sind „8000", „16000", „22050" und „24000". Der Standardwert für Standardstimmen ist „22050". Der Standardwert für neuronale Stimmen ist „24000". Der Standardwert für Stimmen in Langform ist „24000". Der Standardwert für generative Stimmen ist „24000".

Gültige Werte für pcm sind „8000" und „16000". Der Standardwert ist „16000".

Der gültige Wert für ogg_opus ist „48000".

Der gültige Wert für mu-law und a-law ist „8000".

Typ: Zeichenfolge

Erforderlich: Nein

SnsTopicArn

ARN für das SNS-Thema, das optional für die Bereitstellung von Statusbenachrichtigungen für eine Sprachsyntheseaufgabe verwendet wird.

Typ: Zeichenfolge

Pattern: `^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)$`

Erforderlich: Nein

SpeechMarkTypes

Der Typ der Sprachzeichen, die für den Eingabetext zurückgegeben wurden.

Typ: Zeichenfolgen-Array

Array-Mitglieder: Maximale Anzahl von 4 Elementen.

Zulässige Werte: `sentence | ssml | viseme | word`

Erforderlich: Nein

Text

Der zu synthetisierende Eingabetext. Wenn Sie ssml als angeben TextType, folgen Sie dem SSML-Format für den Eingabetext.

Typ: Zeichenfolge

Erforderlich: Ja

TextType

Gibt an, ob der Eingabetext Klartext oder SSML ist. Der Standardwert ist Klartext.

Typ: Zeichenfolge

Zulässige Werte: `ssml | text`

Erforderlich: Nein

VoiceId

Sprach-ID, die für die Synthese verwendet werden soll.

Typ: Zeichenfolge

Zulässige Werte: `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 | Jitka | Sabrina | Jasmine | Jihye | Ambre | Beatrice | Florian | Lennart | Lorenzo | Tiffany`

Erforderlich: Ja

Antwortsyntax

```
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"
   }
}
```

Antwortelemente

Wenn die Aktion erfolgreich ist, sendet der Service eine HTTP 200-Antwort zurück.

Die folgenden Daten werden vom Service im JSON-Format zurückgegeben.

SynthesisTask

SynthesisTask Objekt, das Informationen und Attribute zu einer neu eingereichten Sprachsyntheseaufgabe bereitstellt.

Typ: SynthesisTask Objekt

Fehler

EngineNotSupportedException

Diese Engine ist mit der von Ihnen angegebenen Stimme nicht kompatibel. Wählen Sie eine neue Stimme, die mit der Engine kompatibel ist, oder wechseln Sie die Engine und starten Sie den Vorgang erneut.

HTTP-Statuscode: 400

InvalidS3BucketException

Der angegebene Amazon S3 S3-Bucket-Name ist ungültig. Bitte überprüfen Sie Ihre Eingabe mit den Anforderungen für die Benennung von S3-Buckets und versuchen Sie es erneut.

HTTP-Statuscode: 400

InvalidS3KeyException

Das angegebene Amazon S3 S3-Schlüsselpräfix ist ungültig. Bitte geben Sie einen gültigen S3-Objektschlüsselnamen an.

HTTP-Statuscode: 400

InvalidSampleRateException

Die angegebene Samplerate ist nicht gültig.

HTTP-Statuscode: 400

InvalidSnsTopicArnException

Der angegebene ARN für das SNS-Thema ist ungültig. Bitte geben Sie einen gültigen SNS-Thema-ARN ein und versuchen Sie es erneut.

HTTP-Statuscode: 400

InvalidSsmlException

Die von Ihnen angegebene SSML ist ungültig. Überprüfen Sie die SSML-Syntax sowie die Schreibweise der Tags und Werte, und versuchen Sie es erneut.

HTTP-Statuscode: 400

LanguageNotSupportedException

Die angegebene Sprache wird derzeit von Amazon Polly in dieser Funktion nicht unterstützt.

HTTP-Statuscode: 400

LexiconNotFoundException

Amazon Polly kann das angegebene Lexikon nicht finden. Dies kann durch ein fehlendes Lexikon, durch einen falsch geschriebenen Namen oder durch die Angabe eines Lexikons in einer anderen Region verursacht werden.

Vergewissern Sie sich, dass das Lexikon existiert, sich in der Region befindet (siehe ListLexicons) und ob Sie den Namen richtig geschrieben haben. Versuchen Sie es dann erneut.

HTTP-Statuscode: 404

MarksNotSupportedForFormatException

Sprachzeichen werden für die `OutputFormat` ausgewählten Elemente nicht unterstützt. Sprachzeichen sind nur für Inhalte im `json` Format verfügbar.

HTTP-Statuscode: 400

ServiceFailureException

Ein unbekannter Zustand hat zu einem Dienstausfall geführt.

HTTP Status Code: 500

SsmlMarksNotSupportedForTextTypeException

SSML-Sprachzeichen werden für Klartexteingaben nicht unterstützt.

HTTP-Statuscode: 400

TextLengthExceededException

Der Wert des Parameters „Text“ ist länger als die akzeptierten Grenzwerte. Für die `SynthesizeSpeech` API beträgt das Limit für Eingabetext insgesamt maximal 6000 Zeichen, von denen nicht mehr als 3000 fakturierte Zeichen sein können. Für die `StartSpeechSynthesisTask` API liegt das Maximum bei 200.000 Zeichen, wovon nicht mehr als 100.000 fakturierte Zeichen sein können. SSML-Tags werden nicht als berechnete Zeichen gezählt.

HTTP-Statuscode: 400

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einer der sprachspezifischen Sprachen finden Sie im AWS SDKs Folgenden:

- AWS Befehlszeilenschnittstelle V2
- AWS SDK for .NET V4
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK für JavaScript V3
- AWS SDK für Kotlin
- AWS SDK for PHP V3
- AWS SDK für Python
- AWS SDK for Ruby V3

SynthesizeSpeech

Synthetisiert UTF-8-Eingabe, Klartext oder SSML in einen Byte-Stream Die SSML-Eingabe muss gültiges, wohlgeformtes SSML sein. Einige Alphabete sind möglicherweise nicht mit allen Stimmen verfügbar (z. B. wird Kyrillisch möglicherweise von englischen Stimmen überhaupt nicht gelesen), es sei denn, es wird eine Phonemzuordnung verwendet. Weitere Informationen finden Sie unter So funktioniert es.

Anforderungssyntax

```
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-Anfrageparameter

Die Anforderung verwendet keine URI-Parameter.

Anforderungstext

Die Anforderung akzeptiert die folgenden Daten im JSON-Format.

Engine

Gibt die Engine (`standard`,`neural`, oder`generative`) an`long-form`, die Amazon Polly bei der Verarbeitung von Eingabetext für die Sprachsynthese verwenden soll. Stellen Sie eine Engine bereit, die von der ausgewählten Stimme unterstützt wird. Wenn Sie kein Modul angeben, ist standardmäßig das Standardmodul ausgewählt. Wenn eine gewählte Stimme von der Standard-Engine nicht unterstützt wird, führt dies zu einem Fehler. Informationen zu Amazon Polly-Stimmen

und zu den Stimmen, die für die einzelnen Engines verfügbar sind, finden Sie unter Verfügbare Stimmen.

Typ: Zeichenfolge

Zulässige Werte: `standard | neural | long-form | generative`

Erforderlich: Nein

LanguageCode

Optionaler Sprachcode für die Synthesize Speech-Anforderung. Dies ist nur erforderlich, wenn Sie eine zweisprachige Stimme wie Aditi verwenden, die entweder für indisches Englisch (en-IN) oder Hindi (Hi-in) verwendet werden kann.

Wenn eine zweisprachige Stimme verwendet wird und kein Sprachcode angegeben ist, verwendet Amazon Polly die Standardsprache der zweisprachigen Stimme. Die Standardsprache für jede Stimme ist die Sprache, die bei der Operation für den DescribeVoicesParameter zurückgegeben wurde. `LanguageCode` Wenn beispielsweise kein Sprachcode angegeben ist, verwendet Aditi indisches Englisch statt Hindi.

Typ: Zeichenfolge

Zulässige Werte: `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 | cs-CZ | de-CH | en-SG`

Erforderlich: Nein

LexiconNames

Liste mit einem oder mehreren Aussprache-Lexikonnamen, die der Dienst bei der Synthese anwenden soll. Lexika werden nur angewendet, wenn die Sprache des Lexikons mit der Sprache der Stimme übereinstimmt. Hinweise zum Speichern von Lexika finden Sie unter. PutLexicon

Typ: Zeichenfolgen-Array

Array-Mitglieder: Maximale Anzahl von 5 Elementen.

Pattern: `[0-9A-Za-z]{1,20}`

Erforderlich: Nein

OutputFormat

Das Format, in dem die zurückgegebene Ausgabe codiert wird. Für Audiostreams ist dies mp3, ogg_vorbis, ogg_opus, mu-law, a-law oder pcm. Für Sprachzeichen wird dies json sein.

Wenn pcm verwendet wird, liegt der zurückgegebene Inhalt audio/pcm in einem vorzeichenbehafteten 16-Bit-Little-Endian-Format mit 1 Kanal (mono) vor.

Typ: Zeichenfolge

Zulässige Werte: `json | mp3 | ogg_opus | ogg_vorbis | pcm | mulaw | alaw`

Erforderlich: Ja

SampleRate

Die in Hz angegebene Audiofrequenz.

Die gültigen Werte für mp3 und ogg_vorbis sind „8000", „16000", „22050", „24000", „44100" und „48000". Der Standardwert für Standardstimmen ist „22050". Der Standardwert für neuronale Stimmen ist „24000". Der Standardwert für Stimmen in Langform ist „24000". Der Standardwert für generative Stimmen ist „24000".

Gültige Werte für pcm sind „8000" und „16000". Der Standardwert ist „16000".

Der gültige Wert für ogg_opus ist „48000".

Der gültige Wert für mu-law und a-law ist „8000".

Typ: Zeichenfolge

Erforderlich: Nein

SpeechMarkTypes

Der Typ der Sprachzeichen, die für den Eingabetext zurückgegeben wurden.

Typ: Zeichenfolgen-Array

Array-Mitglieder: Maximale Anzahl von 4 Elementen.

Zulässige Werte: `sentence | ssml | viseme | word`

Erforderlich: Nein

Text

Geben Sie den zu synthetisierenden Text ein. Wenn Sie `ssml` als angeben`TextType`, folgen Sie dem SSML-Format für den Eingabetext.

Typ: Zeichenfolge

Erforderlich: Ja

TextType

Gibt an, ob der Eingabetext Klartext oder SSML ist. Der Standardwert ist Klartext. Weitere Informationen finden Sie unter Verwenden von SSML.

Typ: Zeichenfolge

Zulässige Werte: `ssml | text`

Erforderlich: Nein

VoiceId

Sprach-ID, die für die Synthese verwendet werden soll. Sie können eine Liste der verfügbaren Sprachbefehle abrufen, IDs indem Sie den DescribeVoicesVorgang aufrufen.

Typ: Zeichenfolge

Zulässige Werte: `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 | Jitka | Sabrina | Jasmine | Jihye | Ambre | Beatrice | Florian | Lennart | Lorenzo | Tiffany`

Erforderlich: Ja

Antwortsyntax

```
HTTP/1.1 200
Content-Type: ContentType
x-amzn-RequestCharacters: RequestCharacters

AudioStream
```

Antwortelemente

Wenn die Aktion erfolgreich ist, sendet der Service eine HTTP 200-Antwort zurück.

Die Antwort gibt die folgenden HTTP-Header zurück.

ContentType

Gibt den Typ des Audiostreams an. Dies sollte den `OutputFormat` Parameter in Ihrer Anfrage widerspiegeln.

- Wenn Sie `mp3` als die anfordern`OutputFormat`, wird `ContentType` audio/mpeg zurückgegeben.
- Wenn Sie `ogg_vorbis` als angeben`OutputFormat`, wird Audio/OGG `ContentType` zurückgegeben.
- Wenn Sie `ogg_opus` als angeben`OutputFormat`, ist das `ContentType` zurückgegebene Objekt Audio/OGG.
- Wenn Sie `pcm` als die anfordern`OutputFormat`, erfolgt die `ContentType` Rückgabe audio/pcm in einem vorzeichenbehafteten 16-Bit-Little-Endian-Format mit 1 Kanal (mono).
- Wenn Sie `mu-law` als angeben, ist das zurückgegebene `OutputFormat` Audio/Mulaw. `ContentType`
- Wenn Sie `a-law` als die anfordern`OutputFormat`, ist die `ContentType` zurückgegebene Antwort audio/alaw.
- Wenn Sie `json` als die anfordern`OutputFormat`, lautet die `ContentType` zurückgegebene Datei application/. x-json-stream

RequestCharacters

Anzahl der synthetisierten Zeichen.

Die Antwort gibt folgendes als HTTP-Hauptteil zurück.

AudioStream

Stream, der die synthetisierte Sprache enthält.

Fehler

EngineNotSupportedException

Diese Engine ist mit der von Ihnen angegebenen Stimme nicht kompatibel. Wählen Sie eine neue Stimme, die mit der Engine kompatibel ist, oder wechseln Sie die Engine und starten Sie den Vorgang erneut.

HTTP-Statuscode: 400

InvalidSampleRateException

Die angegebene Samplerate ist nicht gültig.

HTTP-Statuscode: 400

InvalidSsmlException

Die von Ihnen angegebene SSML ist ungültig. Überprüfen Sie die SSML-Syntax sowie die Schreibweise der Tags und Werte, und versuchen Sie es erneut.

HTTP-Statuscode: 400

LanguageNotSupportedException

Die angegebene Sprache wird derzeit von Amazon Polly in dieser Funktion nicht unterstützt.

HTTP-Statuscode: 400

LexiconNotFoundException

Amazon Polly kann das angegebene Lexikon nicht finden. Dies kann durch ein fehlendes Lexikon, durch einen falsch geschriebenen Namen oder durch die Angabe eines Lexikons in einer anderen Region verursacht werden.

Vergewissern Sie sich, dass das Lexikon existiert, sich in der Region befindet (siehe ListLexicons) und ob Sie den Namen richtig geschrieben haben. Versuchen Sie es dann erneut.

HTTP-Statuscode: 404

MarksNotSupportedForFormatException

Sprachzeichen werden für die `OutputFormat` ausgewählten Elemente nicht unterstützt. Sprachzeichen sind nur für Inhalte im `json` Format verfügbar.

HTTP-Statuscode: 400

ServiceFailureException

Ein unbekannter Zustand hat zu einem Dienstausfall geführt.

HTTP Status Code: 500

SsmlMarksNotSupportedForTextTypeException

SSML-Sprachzeichen werden für Klartexteingaben nicht unterstützt.

HTTP-Statuscode: 400

TextLengthExceededException

Der Wert des Parameters „Text“ ist länger als die akzeptierten Grenzwerte. Für die `SynthesizeSpeech` API beträgt das Limit für Eingabetext insgesamt maximal 6000 Zeichen, von denen nicht mehr als 3000 fakturierte Zeichen sein können. Für die `StartSpeechSynthesisTask` API liegt das Maximum bei 200.000 Zeichen, von denen nicht mehr als 100.000 fakturierte Zeichen sein können. SSML-Tags werden nicht als berechnete Zeichen gezählt.

HTTP-Statuscode: 400

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einer der sprachspezifischen Sprachen finden Sie im AWS SDKs Folgenden:

- AWS Befehlszeilenschnittstelle V2
- AWS SDK for .NET V4
- AWS SDK for C++
- AWS SDK for Go v2
- AWS SDK for Java V2
- AWS SDK für JavaScript V3

- AWS SDK für Kotlin
- AWS SDK for PHP V3
- AWS SDK für Python
- AWS SDK for Ruby V3

Datentypen

Die folgenden Datentypen werden unterstützt:

- AudioEvent
- CloseStreamEvent
- FlushStreamConfiguration
- Lexicon
- LexiconAttributes
- LexiconDescription
- StartSpeechSynthesisStreamActionStream
- StartSpeechSynthesisStreamEventStream
- StreamClosedEvent
- SynthesisTask
- TextEvent
- ThrottlingReason
- ValidationExceptionField
- Voice

AudioEvent

Enthält einen Teil synthetisierter Audiodaten.

Inhalt

AudioChunk

Ein Teil synthetisierter Audiodaten, die in dem durch den Parameter angegebenen Format codiert sind. `OutputFormat`

Typ: Base64-kodiertes Binärdatenobjekt

Erforderlich: Nein

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einer der sprachspezifischen Sprachen AWS SDKs finden Sie im Folgenden:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

CloseStreamEvent

Zeigt das Ende des Eingabestreams an. Nach dem Senden dieses Ereignisses wird der Eingabestream geschlossen und alle Audiodaten werden zurückgegeben.

Inhalt

Die Mitglieder dieser Ausnahmestruktur sind kontextabhängig.

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einer der sprachspezifischen Sprachen finden Sie im Folgenden AWS SDKs:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

FlushStreamConfiguration

Konfiguration, die steuert, wann synthetisierte Audiodaten an den Ausgangsstream gesendet werden.

Inhalt

Force

Gibt an, ob die Synthese-Engine gezwungen werden soll, gepufferte Audiodaten sofort in den Ausgabestrom zu schreiben.

Typ: Boolesch

Erforderlich: Nein

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einer der sprachspezifischen Sprachen finden Sie im AWS SDKs Folgenden:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

Lexicon

Stellt den Lexikonnamen und den Lexikoninhalt im Zeichenkettenformat bereit. Weitere Informationen finden Sie unter Pronunciation Lexicon Specification (PLS) Version 1.0.

Inhalt

Content

Inhalt des Lexikons im Zeichenkettenformat. Der Inhalt eines Lexikons muss im PLS-Format vorliegen.

Typ: Zeichenfolge

Erforderlich: Nein

Name

Name des Lexikons.

Typ: Zeichenfolge

Pattern: `[0-9A-Za-z]{1,20}`

Erforderlich: Nein

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einer der sprachspezifischen Sprachen finden Sie im AWS SDKs Folgenden:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

LexiconAttributes

Enthält Metadaten, die das Lexikon beschreiben, z. B. die Anzahl der Lexeme, den Sprachcode usw. Weitere Informationen finden Sie unter Lexika verwalten.

Inhalt

Alphabet

Im Lexikon verwendetes phonetisches Alphabet. Gültige Werte sind `ipa` und `x-sampa`.

Typ: Zeichenfolge

Erforderlich: Nein

LanguageCode

Sprachcode, für den das Lexikon gilt. Ein Lexikon mit einem Sprachcode wie „en“ würde auf alle englischen Sprachen (en-GB, en-US, en-AUS, en-WLS usw.) angewendet werden.

Typ: Zeichenfolge

Zulässige Werte: `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 | cs-CZ | de-CH | en-SG`

Erforderlich: Nein

LastModified

Datum, an dem das Lexikon zuletzt geändert wurde (ein Zeitstempelwert).

Typ: Zeitstempel

Erforderlich: Nein

LexemesCount

Anzahl der Lexeme im Lexikon.

Typ: Ganzzahl

Erforderlich: Nein

LexiconArn

Amazon-Ressourcenname (ARN) des Lexikons.

Typ: Zeichenfolge

Erforderlich: Nein

Size

Gesamtgröße des Lexikons in Zeichen.

Typ: Ganzzahl

Erforderlich: Nein

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einer der sprachspezifischen Sprachen finden Sie im AWS SDKs Folgenden:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

LexiconDescription

Beschreibt den Inhalt des Lexikons.

Inhalt

Attributes

Stellt Lexikon-Metadaten bereit.

Typ: LexiconAttributes Objekt

Erforderlich: Nein

Name

Name des Lexikons.

Typ: Zeichenfolge

Pattern: `[0-9A-Za-z]{1,20}`

Erforderlich: Nein

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einer der sprachspezifischen Sprachen finden Sie im AWS SDKs Folgenden:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

StartSpeechSynthesisStreamActionStream

Eingehender Eventstream zum Senden von Eingabe- und Steuerereignissen zur Verwaltung der bidirektionalen Sprachsynthese.

Inhalt

CloseStreamEvent

Ein Ereignis, das das Ende des Eingabestreams anzeigt.

Typ: CloseStreamEvent Objekt

Erforderlich: Nein

TextEvent

Ein Textereignis, das Inhalte enthält, die synthetisiert werden sollen.

Typ: TextEvent Objekt

Erforderlich: Nein

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einer der sprachspezifischen Sprachen finden Sie im AWS SDKs Folgenden:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

StartSpeechSynthesisStreamEventStream

Ausgehender Event-Stream, der synthetisierte Audiodaten und Stream-Statusereignisse enthält.

Inhalt

AudioEvent

Ein Audioereignis, das synthetisierte Sprache enthält.

Typ: AudioEvent Objekt

Erforderlich: Nein

ServiceFailureException

Ein unbekannter Zustand hat einen Dienstausfall verursacht.

Typ: Ausnahme

HTTP Status Code: 500

Erforderlich: Nein

ServiceQuotaExceededException

Eine Ausnahme, die darauf hinweist, dass ein Dienstkontingent überschritten würde.

Typ: Ausnahme

HTTP-Statuscode: 402

Erforderlich: Nein

StreamClosedEvent

Ein Ereignis mit zusammenfassenden Informationen, das darauf hinweist, dass der Stream geschlossen wurde.

Typ: StreamClosedEvent Objekt

Erforderlich: Nein

ThrottlingException

Eine Ausnahme, die darauf hinweist, dass die Anfrage gedrosselt wurde.

Typ: Ausnahme

HTTP-Statuscode: 400

Erforderlich: Nein

ValidationException

Eine Ausnahme, die darauf hinweist, dass die Eingabe nicht validiert wurde.

Typ: Ausnahme

HTTP-Statuscode: 400

Erforderlich: Nein

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einer der sprachspezifischen Sprachen AWS SDKs finden Sie im Folgenden:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

StreamClosedEvent

Zeigt an, dass der Synthesestream geschlossen ist, und bietet zusammenfassende Informationen.

Inhalt

RequestCharacters

Die Gesamtzahl der während der Streaming-Sitzung synthetisierten Zeichen.

Typ: Ganzzahl

Erforderlich: Nein

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einer der sprachspezifischen Sprachen finden Sie im AWS SDKs Folgenden:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

SynthesisTask

SynthesisTask Objekt, das Informationen über eine Sprachsyntheseaufgabe bereitstellt.

Inhalt

CreationTime

Zeitstempel für den Zeitpunkt, zu dem die Syntheseaufgabe gestartet wurde.

Typ: Zeitstempel

Erforderlich: Nein

Engine

Gibt die Engine (`standard`, `long-form` oder`generative`) an`neural`, die Amazon Polly bei der Verarbeitung von Eingabetext für die Sprachsynthese verwenden soll. Die Verwendung einer Stimme, die für die gewählte Engine nicht unterstützt wird, führt zu einem Fehler.

Typ: Zeichenfolge

Zulässige Werte: `standard | neural | long-form | generative`

Erforderlich: Nein

LanguageCode

Optionaler Sprachcode für eine Syntheseaufgabe. Dies ist nur erforderlich, wenn Sie eine zweisprachige Stimme wie Aditi verwenden, die entweder für indisches Englisch (en-IN) oder Hindi (Hi-in) verwendet werden kann.

Wenn eine zweisprachige Stimme verwendet wird und kein Sprachcode angegeben ist, verwendet Amazon Polly die Standardsprache der zweisprachigen Stimme. Die Standardsprache für jede Stimme ist die Sprache, die bei der Operation für den DescribeVoicesParameter zurückgegeben wurde. `LanguageCode` Wenn beispielsweise kein Sprachcode angegeben ist, verwendet Aditi indisches Englisch statt Hindi.

Typ: Zeichenfolge

Zulässige Werte: `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 | cs-CZ | de-CH | en-SG`

Erforderlich: Nein

LexiconNames

Liste mit einem oder mehreren Aussprache-Lexikonnamen, die der Dienst bei der Synthese anwenden soll. Lexika werden nur angewendet, wenn die Sprache des Lexikons mit der Sprache der Stimme übereinstimmt.

Typ: Zeichenfolgen-Array

Array-Mitglieder: Maximale Anzahl von 5 Elementen.

Pattern: `[0-9A-Za-z]{1,20}`

Erforderlich: Nein

OutputFormat

Das Format, in dem die zurückgegebene Ausgabe codiert wird. Für Audiostreams ist dies mp3, ogg_vorbis, ogg_opus, mu-law, a-law oder pcm. Bei Sprachzeichen ist dies json.

Typ: Zeichenfolge

Zulässige Werte: `json | mp3 | ogg_opus | ogg_vorbis | pcm | mulaw | alaw`

Erforderlich: Nein

OutputUri

Pfad für die Sprachausgabedatei.

Typ: Zeichenfolge

Erforderlich: Nein

RequestCharacters

Anzahl der synthetisierten fakturierbaren Zeichen.

Typ: Ganzzahl

Erforderlich: Nein

SampleRate

Die in Hz angegebene Audiofrequenz.

Die gültigen Werte für mp3 und ogg_vorbis sind „8000", „16000", „22050" und „24000". Der Standardwert für Standardstimmen ist „22050". Der Standardwert für neuronale Stimmen ist „24000". Der Standardwert für Stimmen in Langform ist „24000". Der Standardwert für generative Stimmen ist „24000".

Gültige Werte für pcm sind „8000" und „16000". Der Standardwert ist „16000".

Der gültige Wert für ogg_opus ist „48000".

Der gültige Wert für mu-law und a-law ist „8000".

Typ: Zeichenfolge

Erforderlich: Nein

SnsTopicArn

ARN für das SNS-Thema, das optional für die Bereitstellung von Statusbenachrichtigungen für eine Sprachsyntheseaufgabe verwendet wird.

Typ: Zeichenfolge

Pattern: `^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)$`

Erforderlich: Nein

SpeechMarkTypes

Der Typ der Sprachzeichen, die für den Eingabetext zurückgegeben wurden.

Typ: Zeichenfolgen-Array

Array-Mitglieder: Maximale Anzahl von 4 Elementen.

Zulässige Werte: `sentence | ssml | viseme | word`

Erforderlich: Nein

TaskId

Die von Amazon Polly generierte Kennung für eine Sprachsyntheseaufgabe.

Typ: Zeichenfolge

Pattern: `^[a-zA-Z0-9_-]{1,100}$`

Erforderlich: Nein

TaskStatus

Aktueller Status der einzelnen Sprachsyntheseaufgabe.

Typ: Zeichenfolge

Zulässige Werte: `scheduled | inProgress | completed | failed`

Erforderlich: Nein

TaskStatusReason

Grund für den aktuellen Status einer bestimmten Sprachsyntheseaufgabe, einschließlich Fehler, wenn die Aufgabe fehlgeschlagen ist.

Typ: Zeichenfolge

Erforderlich: Nein

TextType

Gibt an, ob der Eingabetext Klartext oder SSML ist. Der Standardwert ist Klartext.

Typ: Zeichenfolge

Zulässige Werte: `ssml | text`

Erforderlich: Nein

VoiceId

Sprach-ID, die für die Synthese verwendet werden soll.

Typ: Zeichenfolge

Zulässige Werte: `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 | Jitka | Sabrina
| Jasmine | Jihye | Ambre | Beatrice | Florian | Lennart | Lorenzo |
Tiffany
```

Erforderlich: Nein

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einer der sprachspezifischen Sprachen AWS SDKs finden Sie im Folgenden:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

TextEvent

Enthält Textinhalte, die zu Sprache synthetisiert werden sollen.

Inhalt

Text

Der Textinhalt, der synthetisiert werden soll. Wenn Sie `ssml` als angeben`TextType`, folgen Sie dem SSML-Format für den Eingabetext.

Typ: Zeichenfolge

Erforderlich: Ja

FlushStreamConfiguration

Konfiguration zur Steuerung, wann synthetisiertes Audio in den Ausgangsstream übertragen wird.

Typ: FlushStreamConfiguration Objekt

Erforderlich: Nein

TextType

Gibt an, ob der Eingabetext Klartext oder SSML ist. Standard: Klartext.

Typ: Zeichenfolge

Zulässige Werte: `ssml | text`

Erforderlich: Nein

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einer der sprachspezifischen Sprachen AWS SDKs finden Sie im Folgenden:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

ThrottlingReason

Stellt Informationen zu einem bestimmten Drosselungsgrund bereit.

Inhalt

reason

Der Ursachencode, der erklärt, warum die Anfrage gedrosselt wurde.

Typ: Zeichenfolge

Erforderlich: Nein

resource

Die Ressource, die die Drosselung verursacht hat.

Typ: Zeichenfolge

Erforderlich: Nein

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einer der sprachspezifischen Sprachen finden Sie im Folgenden AWS SDKs:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

ValidationExceptionField

Informationen zu einem Feld, bei dem die Überprüfung fehlgeschlagen ist.

Inhalt

message

Eine Meldung, in der beschrieben wird, warum das Feld nicht validiert werden konnte.

Typ: Zeichenfolge

Erforderlich: Ja

name

Der Name des Felds, bei dem die Überprüfung fehlgeschlagen ist.

Typ: Zeichenfolge

Erforderlich: Ja

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einer der sprachspezifischen Sprachen AWS SDKs finden Sie im Folgenden:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

Voice

Beschreibung der Stimme.

Inhalt

AdditionalLanguageCodes

Zusätzliche Codes für Sprachen, die für die angegebene Stimme zusätzlich zur Standardsprache verfügbar sind.

Die Standardsprache für Aditi ist beispielsweise Indisches Englisch (en-IN), weil es zuerst für diese Sprache verwendet wurde. Da Aditi zweisprachig ist und sowohl indisches Englisch als auch Hindi fließend spricht, würde dieser Parameter den Code anzeigen. `hi-IN`

Typ: Zeichenfolgen-Array

Zulässige Werte: `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 | cs-CZ | de-CH | en-SG`

Erforderlich: Nein

Gender

Geschlecht der Stimme.

Typ: Zeichenfolge

Zulässige Werte: `Female | Male`

Erforderlich: Nein

Id

Amazon Polly hat die Sprach-ID zugewiesen. Dies ist die ID, die Sie beim Aufrufen des `SynthesizeSpeech` Vorgangs angeben.

Typ: Zeichenfolge

Zulässige Werte: `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 | Jitka | Sabrina | Jasmine | Jihye | Ambre | Beatrice | Florian | Lennart | Lorenzo | Tiffany

Erforderlich: Nein

LanguageCode

Sprachcode der Stimme.

Typ: Zeichenfolge

Zulässige Werte: 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 | cs-CZ | de-CH | en-SG

Erforderlich: Nein

LanguageName

Für Menschen lesbarer Name der Sprache in Englisch.

Typ: Zeichenfolge

Erforderlich: Nein

Name

Name der Stimme (zum Beispiel Salli, Kendra usw.). Dadurch wird ein für Menschen lesbarer Sprachname bereitgestellt, den Sie möglicherweise in Ihrer Anwendung anzeigen.

Typ: Zeichenfolge

Erforderlich: Nein

SupportedEngines

Gibt an, welche Engines (`standard`neural, `long-form` odergenerative) von einer bestimmten Stimme unterstützt werden.

Typ: Zeichenfolgen-Array

Zulässige Werte: `standard | neural | long-form | generative`

Erforderlich: Nein

Weitere Informationen finden Sie unter:

Weitere Informationen zur Verwendung dieser API in einer der sprachspezifischen Sprachen AWS SDKs finden Sie im Folgenden:

- AWS SDK for C++
- AWS SDK for Java V2
- AWS SDK for Ruby V3

Häufige Fehlertypen

In diesem Abschnitt werden häufig auftretende Fehlertypen aufgeführt, die dieser AWS Dienst möglicherweise zurückgibt. Nicht alle Dienste geben alle hier aufgeführten Fehlertypen zurück. Informationen zu Fehlern, die spezifisch für eine API-Aktion für diesen Service sind, finden Sie unter dem Thema für diese API-Aktion.

AccessDeniedException

Sie sind nicht berechtigt, diese Aktion auszuführen. Stellen Sie sicher, dass Ihre IAM-Richtlinie die erforderlichen Berechtigungen enthält.

HTTP-Statuscode: 403

ExpiredTokenException

Das in der Anfrage enthaltene Sicherheitstoken ist abgelaufen. Fordern Sie ein neues Sicherheitstoken an und versuchen Sie es erneut.

HTTP-Statuscode: 403

IncompleteSignature

Die Anforderungssignatur entspricht nicht den AWS Standards. Stellen Sie sicher, dass Sie gültige AWS Anmeldeinformationen verwenden und dass Ihre Anfrage richtig formatiert ist. Wenn Sie ein SDK verwenden, stellen Sie sicher, dass es auf dem neuesten Stand ist.

HTTP-Statuscode: 403

InternalFailure

Die Anfrage kann derzeit aufgrund eines internen Serverproblems nicht bearbeitet werden. Bitte versuchen Sie es später erneut. Wenn das Problem weiterhin besteht, wenden Sie sich an den AWS Support.

HTTP Status Code: 500

MalformedHttpRequestException

Der Text der Anfrage kann nicht verarbeitet werden. Dies ist normalerweise der Fall, wenn der Anforderungstext nicht mit dem angegebenen Algorithmus zur Inhaltskodierung dekomprimiert werden kann. Stellen Sie sicher, dass der Header für die Inhaltskodierung dem verwendeten Komprimierungsformat entspricht.

HTTP-Statuscode: 400

NotAuthorized

Sie sind nicht berechtigt, diese Aktion auszuführen. Stellen Sie sicher, dass Ihre IAM-Richtlinie die erforderlichen Berechtigungen enthält.

HTTP-Statuscode: 401

OptInRequired

Ihr AWS Konto benötigt ein Abonnement für diesen Service. Stellen Sie sicher, dass Sie den Dienst in Ihrem Konto aktiviert haben.

HTTP-Statuscode: 403

RequestAbortedException

Die Anfrage wurde abgebrochen, bevor eine Antwort zurückgegeben werden konnte. Dies passiert normalerweise, wenn der Client die Verbindung schließt.

HTTP-Statuscode: 400

RequestEntityTooLargeException

Die Anforderungsentität ist zu groß. Reduzieren Sie die Größe des Anfragetexts und versuchen Sie es erneut.

HTTP-Statuscode: 413

RequestTimeoutException

Das Zeitlimit für die Anfrage wurde überschritten. Der Server hat die vollständige Anfrage nicht innerhalb des erwarteten Zeitraums erhalten. Bitte versuchen Sie es erneut.

HTTP-Statuscode: 408

ServiceUnavailable

Der Service ist vorübergehend nicht verfügbar. Bitte versuchen Sie es später erneut.

HTTP Status Code: 503

ThrottlingException

Ihre Anforderungsrate ist zu hoch. Anfragen, die diese Ausnahme erhalten, werden AWS SDKs automatisch wiederholt. Verringern Sie die Häufigkeit der Anforderungen.

HTTP-Statuscode: 400

UnknownOperationException

Die Aktion oder der Vorgang wurde nicht erkannt. Stellen Sie sicher, dass der Aktionsname richtig geschrieben ist und dass er von der von Ihnen verwendeten API-Version unterstützt wird.

HTTP-Statuscode: 404

UnrecognizedClientException

Das von Ihnen angegebene X.509-Zertifikat oder die AWS Zugangsschlüssel-ID ist in unseren Aufzeichnungen nicht vorhanden. Stellen Sie sicher, dass Sie gültige Anmeldeinformationen verwenden und dass diese nicht abgelaufen sind.

HTTP-Statuscode: 403

ValidationError

Die Eingabe entspricht nicht dem erforderlichen Format oder den erforderlichen Einschränkungen. Überprüfen Sie, ob alle erforderlichen Parameter enthalten sind und ob die Werte gültig sind.

HTTP-Statuscode: 400

Geläufige Parameter

Die folgende Liste enthält die Parameter, die alle Aktionen zum Signieren von Signature-Version-4-Anforderungen mit einer Abfragezeichenfolge verwenden. Alle aktionsspezifischen Parameter werden im Thema für diese Aktion aufgelistet. Weitere Informationen zu Signature Version 4 finden Sie unter Signieren von AWS API-Anfragen im IAM-Benutzerhandbuch.

X-Amz-Algorithm

Der Hashalgorithmus, den Sie zum Erstellen der Anforderungssignatur verwendet haben.

Bedingung: Geben Sie diesen Parameter an, wenn Sie Authentifizierungsinformationen in eine Abfragezeichenfolge anstatt in den HTTP-Autorisierungsheader aufnehmen.

Type: Zeichenkette

Zulässige Werte: `AWS4-HMAC-SHA256`

Required: Conditional

X-Amz-Credential

Der Wert des Anmeldeinformationsumfangs. Dabei handelt es sich um eine Zeichenfolge, die Ihren Zugriffsschlüssel, das Datum, die gewünschte Region und eine Zeichenfolge zur Beendigung („aws4_request“) beinhaltet. Der Wert wird im folgenden Format ausgedrückt: Zugriffsschlüssel/JJJJMMTT/Region/Service/aws4_request.

Weitere Informationen finden Sie im IAM-Benutzerhandbuch unter Erstellen einer signierten AWS API-Anfrage.

Bedingung: Geben Sie diesen Parameter an, wenn Sie Authentifizierungsinformationen in eine Abfragezeichenfolge anstatt in den HTTP-Autorisierungsheader aufnehmen.

Type: Zeichenkette

Required: Conditional

X-Amz-Date

Das Datum, das zum Erstellen der Signatur verwendet wird. Das Format muss das ISO 8601-Basisformat (JJJJMMTT'T'SSMMSS'Z') sein. Beispielsweise ist das folgende Datum und Uhrzeit ein gültiger X-Amz-Date Wert:`20120325T120000Z`.

Bedingung: X-Amz-Date ist für alle Anfragen optional; sie kann verwendet werden, um das Datum zu überschreiben, das für das Signieren von Anfragen verwendet wird. Wenn der Date-Header im ISO 8601-Standardformat angegeben ist, X-Amz-Date ist dies nicht erforderlich. Wenn verwendet X-Amz-Date wird, überschreibt er immer den Wert des Date-Headers. Weitere Informationen finden Sie unter Elemente einer AWS API-Anforderungssignatur im IAM-Benutzerhandbuch.

Type: Zeichenkette

Required: Conditional

X-Amz-Security-Token

Das temporäre Sicherheitstoken, das durch einen Aufruf von AWS -Security-Token-Service (AWS STS) abgerufen wurde. Eine Liste der Services, die temporäre Sicherheits-Anmeldeinformationen von AWS STS unterstützen, finden Sie im IAM-Benutzerhandbuch unter AWS-Services , die mit IAM funktionieren.

Bedingung: Wenn Sie temporäre Sicherheitsanmeldedaten von verwenden AWS STS, müssen Sie das Sicherheitstoken angeben.

Type: Zeichenkette

Required: Conditional

X-Amz-Signature

Gibt die hex-codierte Signatur an, die aus der zu signierenden Zeichenfolge und dem abgeleiteten Signaturschlüssel berechnet wurde.

Bedingung: Geben Sie diesen Parameter an, wenn Sie Authentifizierungsinformationen in eine Abfragezeichenfolge anstatt in den HTTP-Autorisierungsheader aufnehmen.

Type: Zeichenkette

Required: Conditional

X-Amz-SignedHeaders

Gibt alle HTTP-Header an, die als Teil der kanonischen Anforderung enthalten waren. Weitere Informationen zur Angabe signierter Header finden Sie unter Erstellen einer signierten AWS API-Anfrage im IAM-Benutzerhandbuch.

Bedingung: Geben Sie diesen Parameter an, wenn Sie Authentifizierungsinformationen in eine Abfragezeichenfolge anstatt in den HTTP-Autorisierungsheader aufnehmen.

Type: Zeichenkette

Required: Conditional

Dokumentenverlauf für Amazon Polly

In der folgenden Tabelle werden wichtige Änderungen in den einzelnen Versionen des Amazon Polly Developer Guide beschrieben. Um Benachrichtigungen über Aktualisierungen dieser Dokumentation zu erhalten, können Sie einen RSS-Feed abonnieren.

- Letzte Aktualisierung der Dokumentation: 28. Mai 2026

Änderung	Beschreibung	Datum
Neue Region für neuronale Stimmen hinzugefügt	Amazon Polly ist jetzt in der AWS Region Asien-Pazifik (Thailand) verfügbar. Diese Region unterstützt neuronale TTS (NTTS) -Stimmen. Weitere Informationen finden Sie unter Neuronale Stimmen.	28. Mai 2026
Neue Region für generative Stimmen hinzugefügt	Generative Stimmen und bidirektionales Streaming von Amazon Polly sind jetzt in einer weiteren AWS Region verfügbar: Europa (Zürich). Weitere Informationen finden Sie unter Generative Stimmen.	28. Mai 2026
Bidirektionales Streaming in neuen Regionen verfügbar	Bidirektionales Streaming mit Amazon Polly ist jetzt in zwei weiteren AWS Regionen verfügbar: Europa (London) und Kanada (Zentral). Weitere Informationen finden Sie unter Generative Stimmen.	20. April 2026
Verwaltete Richtlinie aktualisiert	Amazon Polly hat die verwaltete Richtlinie `AmazonPollyReadOnlyAccess`	19. März 2026

	aktualisiert und umfasst nun auch die Genehmigung für die bidirektionale Streaming -Sprachsynthese. Weitere Informationen finden Sie unter Amazon Polly Polly-Aktualisierungen für AWS verwaltete Richtlinien.	
Neue Stimmen für generatives Text-to-Speech hinzugefügt	Amazon Polly bietet jetzt zehn zusätzliche generative Stimmen: Brian, Aria, Jasmine, Tiffany, Ambre, Florian, Sabrina, Lennart, Beatrice, Lorenzo. Eine Liste generativer TTS-Stimmen finden Sie unter Generative Stimmen.	19. März 2026
Neue Regionen für generative Stimmen hinzugefügt	Generative Stimmen von Amazon Polly sind jetzt in zwei weiteren AWS Regionen verfügbar: Europa (London) und Kanada (Zentral). Weitere Informationen finden Sie unter Generative Stimmen.	19. März 2026
Neue Regionen für generative Stimmen hinzugefügt	Generative Stimmen von Amazon Polly sind jetzt in drei weiteren AWS Regionen verfügbar: Asien-Pazifik (Tokio), Asien-Pazifik (Seoul) und Asien-Pazifik (Singapur). Weitere Informationen finden Sie unter Generative Stimmen.	18. November 2025

Neue Stimmen für generatives Text-to-Speech hinzugefügt	Amazon Polly bietet jetzt sechs zusätzliche generative Stimmen: Seoyeon, Camila, Hannah, Niamh, Laura und Lisa. Eine Liste generativer TTS-Stimmen finden Sie unter Generative Stimmen.	14. November 2025
Neue Region für neuronale Stimmen hinzugefügt	Amazon Polly ist jetzt in der AWS Region Europa (Zürich) verfügbar. Diese Region unterstützt neuronale TTS (NTTS) -Stimmen. Weitere Informationen finden Sie unter Neuronale Stimmen.	20. Oktober 2025
Neue Stimmen für generatives Text-to-Speech hinzugefügt	Amazon Polly bietet jetzt sieben zusätzliche generative Stimmen: Salli, Isabelle, Céline, Liam, Gabrielle, Ola und Ewa. Eine Liste generativer TTS-Stimmen finden Sie unter Generative Stimmen.	26. August 2025
Neue Region für neuronale Stimmen und Standards timmen hinzugefügt	Amazon Polly ist jetzt in der AWS Region Asien-Pazifik (Malaysia) verfügbar. Diese Region unterstützt neuronales TTS (NTTS) und Standards timmen. Weitere Informationen finden Sie unter Neuronale Stimmen und Standards timmen.	27. März 2025

Neue Stimme für neuronale Text-to-Speech hinzugefügt	Amazon Polly bietet jetzt eine zusätzliche koreanische Stimme: Jihye. Eine Liste der NTTS-Stimmen finden Sie unter Neuronale Stimmen.	26. März 2025
Neue Region für neuronale Stimmen und Standardstimmen hinzugefügt	Amazon Polly ist jetzt in der AWS Region Europa (Spanien) verfügbar. Diese Region unterstützt neuronales TTS (NTTS) und Standardstimmen. Weitere Informationen finden Sie unter Neuronale Stimmen und Standardstimmen.	18. Februar 2025
Neue Stimme für neuronale Text-to-Speech hinzugefügt	Amazon Polly bietet jetzt eine zusätzliche englische (singapurische) Stimme: Jasmine. Eine Liste der NTTS-Stimmen finden Sie unter Neuronale Stimmen.	11. Februar 2025
Neue Stimmen für generatives Text-to-Speech hinzugefügt	Amazon Polly bietet jetzt sieben neue generative Stimmen: Pedro, Andrés, Sergio, Daniel, Kajal, Rémi, Bianca. Eine Liste generativer TTS-Stimmen finden Sie unter Generative Stimmen.	21. November 2024

Neue Langform-Stimmen hinzugefügt	Amazon Polly bietet jetzt mehr Langform-Stimmen. Eine weitere englische (USA) und zwei neue spanische (Spanien) Langform-Stimmen wurden hinzugefügt: Patrick, Alba und Raúl. Eine Liste aller Long-formLangform-Stimmen finden Sie unter Stimmen.	14. November 2024
Neue Stimmen für generatives Text-to-Speech hinzugefügt	Amazon Polly bietet jetzt sechs neue generative Stimmen: Ayanda, Léa, Lucia, Mía, Lupe und Vicki. Eine Liste generativer TTS-Stimmen finden Sie unter Generative Stimmen.	6. November 2024
Neue Stimmen für generatives Text-to-Speech hinzugefügt	Amazon Polly bietet jetzt vier neue generative Stimmen: Olivia, Danielle, Joanna und Stephen. Eine Liste generativer TTS-Stimmen finden Sie unter Generative Stimmen.	10. Oktober 2024
Neue Sprachen für Text-to-Speech hinzugefügt	Amazon Polly bietet jetzt zwei neue TTS-Sprachen: Tschechisch (cs-CZ) und Deutsch (Schweiz) (de-CH). Eine Liste der Sprachen finden Sie unter Unterstützte Sprachen.	26. September 2024

Neue Stimmen für NTTS hinzugefügt	Amazon Polly bietet jetzt zwei neue NTTS-Stimmen: Jitka und Sabrina. Eine Liste der NTTS-Stimmen finden Sie unter Neuronale Stimmen.	27. August 2024
Neue generative Spracheng ine hinzugefügt	Amazon Polly bietet jetzt eine generative Sprachengine für längere Inhalte mit drei englischen Stimmen in einer generativen Variante: Amy, Matthew und Ruth. Weitere Informationen finden Sie unter Generative Stimmen.	28. März 2024
Neue Stimme für NTTS hinzugefügt	Amazon Polly bietet jetzt die türkische NTTS-Stimme Burcu. Eine Liste der NTTS-Stimmen finden Sie unter Neuronale Stimmen.	14. Februar 2024
Neue Sprachengine in Langform hinzugefügt	Amazon Polly bietet jetzt eine Sprachengine in Langform, die für längere Inhalte konzipiert ist, mit drei En-US-Stimmen: Danielle, Gregory und Ruth. Weitere Informationen finden Sie unter StimmenLong-form.	16. November 2023
Neue Stimmen wurden für NTTS hinzugefügt	Amazon Polly bietet jetzt zwei neue NTTS-Stimmen für US-Englisch: Danielle und Gregory. Eine Liste der NTTS-Stimmen finden Sie unter Neuronale Stimmen.	05. Oktober 2023

Amazon Polly für Windows	Das Amazon Polly Windows Speech Application Programming Interface (SAPI) -Plugin wird nicht mehr unterstützt.	26. September 2023
Aktualisierte Kontingentrichtlinie für Amazon Polly	Aktualisierter Amazon Polly Polly-Kontingentleitfaden. Beispiele und Erläuterungen der Begriffe hinzugefügt. Die Updates finden Sie unter Kontingente in Amazon Polly.	17. August 2023
Neue Stimme für NTTS hinzugefügt	Amazon Polly bietet jetzt die NTTS-Stimme Zayd in Golf-Arabisch an. Eine Liste der NTTS-Stimmen finden Sie unter Neuronale Stimmen.	16. August 2023
Neue Stimme für NTTS hinzugefügt	Amazon Polly bietet jetzt die belgisch-französische NTTS-Stimme Isabelle. Eine Liste der NTTS-Stimmen finden Sie unter Neuronale Stimmen.	1. August 2023
Neue Stimme für NTTS hinzugefügt	Amazon Polly bietet jetzt die belgisch-niederländische (flämische) NTTS-Stimme Lisa. Eine Liste der NTTS-Stimmen finden Sie unter Neuronale Stimmen.	7. Juni 2023
Neue Stimmen wurden für NTTS hinzugefügt	Amazon Polly bietet jetzt zwei neue NTTS-Stimmen: Irisches Englisch (Niamh) und Dänisch (Sofie). Eine Liste der NTTS-Stimmen finden Sie unter Neuronale Stimmen.	30. Mai 2023

Die IAM-Leitlinien für Amazon Polly wurden aktualisiert	Aktualisierung des Leitfadens zur Ausrichtung an bewährten IAM-Methoden. Weitere Informationen finden Sie unter Bewährte Sicherheitsmethoden in IAM.	19. April 2023
WordPress update	Das Amazon Polly WordPress Polly-Plugin wird nicht mehr unterstützt.	06. April 2023
Neue Region hinzugefügt	Amazon Polly ist jetzt in der AWS Region Asien-Pazifik (Osaka) verfügbar. Diese Region unterstützt neuronales TTS (NTTS). Weitere Informationen finden Sie unter Funktions - und Regionskompatibilität für eine Liste der Regionen, die NTTS unterstützen.	5. April 2023
Neue Stimmen wurden für NTTS hinzugefügt	Amazon Polly bietet jetzt zwei neue japanische NTTS-Stimmen: Kazuha und Tomoko. Eine Liste der NTTS-Stimmen finden Sie unter Neuronale Stimmen.	07. Februar 2023
Neue Stimmen wurden für NTTS hinzugefügt	Amazon Polly bietet jetzt zwei neue NTTS-Stimmen in US-Englisch: Stephen und Ruth. Eine Liste der NTTS-Stimmen finden Sie unter Neuronale Stimmen.	31. Januar 2023

Neue Stimmen wurden für NTTS hinzugefügt	Amazon Polly bietet jetzt neue NTTS-Stimmen für: brasilianisches Portugiesisch (Thiago), kastilisches Spanisch (Sergio), Französisch (Rémi), Italienisch (Adriano) und mexikanisches Spanisch (Andrés). Eine Liste der NTTS-Stimmen finden Sie unter Neuronale Stimmen.	24. Januar 2023
Neue Stimmen wurden für NTTS hinzugefügt	Amazon Polly bietet jetzt NTTS-Stimmen für Arabisch (Hala) und Polnisch (Ola). Eine Liste der NTTS-Stimmen finden Sie unter Neuronale Stimmen.	17. November 2022
Unterstützung veröffentlichen AWS PrivateLink	Amazon Polly bietet jetzt AWS PrivateLink Support. Weitere Informationen finden Sie unter Amazon Polly mit VPC-Endpunkten verwenden.	9. November 2022
Neue Stimmen und Sprachen wurden für NTTS hinzugefügt	Amazon Polly bietet jetzt NTTS-Stimmen für Finnisch (Suvi), Norwegisch (Ida) und Schwedisch (Elin). Eine Liste der NTTS-Stimmen finden Sie unter Neuronale Stimmen.	08. November 2022
Neue Stimme für NTTS hinzugefügt	Amazon Polly bietet jetzt die niederländische NTTS-Stimme Laura. Eine Liste der NTTS-Stimmen finden Sie unter Neuronale Stimmen.	02. November 2022

Neue Region hinzugefügt	Amazon Polly ist jetzt in der AWS Region Europa (Paris) verfügbar. Diese Region unterstützt neuronales TTS (NTTS). Weitere Informationen finden Sie unter Funktions - und Regionskompatibilität für eine Liste der Regionen, die NTTS unterstützen.	22. September 2022
Neue Stimme und Sprache für NTTS hinzugefügt	Amazon Polly bietet jetzt die kantonesische NTTS-Stimme Hiujin. Eine Liste der NTTS-Stimmen finden Sie unter Neuronale Stimmen.	20. September 2022
Neue Region hinzugefügt	Amazon Polly ist jetzt in der AWS Region Asien-Pazifik (Mumbai) verfügbar. Diese Region unterstützt neuronale s TTS (NTTS). Weitere Informationen finden Sie unter Funktions - und Regionsko mpatibilität für eine Liste der Regionen, die NTTS unterstüt zen.	01. September 2022
Neue Stimme für NTTS hinzugefügt	Amazon Polly bietet jetzt die Mandarin-Stimme Zhiyu als NTTS-Stimme an. Eine Liste der NTTS-Stimmen finden Sie unter Neuronale Stimmen.	23. August 2022

Neue Stimme für NTTS hinzugefügt	Amazon Polly bietet jetzt die NTTS-Stimme Kajal auf Hindi an. Eine Liste der NTTS-Stimmen finden Sie unter Neuronale Stimmen.	27. Juli 2022
Neue Stimmen wurden für NTTS hinzugefügt	Amazon Polly bietet jetzt NTTS-Stimmen für US-Spanisch (Pedro), Deutsch (Daniel), kanadisches Französisch (Liam) und britisches Englisch (Arthur). Eine Liste der NTTS-Stimmen finden Sie unter Neuronale Stimmen.	28. Juni 2022
Neue Stimme für NTTS hinzugefügt	Amazon Polly bietet jetzt die portugiesische (brasilianische) Stimme Vitória als NTTS-Stimme an. Eine Liste der NTTS-Stimmen finden Sie unter Neuronale Stimmen.	27. April 2022
Neue Stimme für NTTS hinzugefügt	Amazon Polly bietet jetzt die portugiesische (europäische) Stimme Inês als NTTS-Stimme an. Eine Liste der NTTS-Stimmen finden Sie unter Neuronale Stimmen.	26. April 2022
Neue Stimme und Sprache für NTTS hinzugefügt	Amazon Polly bietet jetzt die deutsche (österreichische) Sprache und die NTTS-Stimme Hannah. Eine Liste der NTTS-Stimmen finden Sie unter Neuronale Stimmen.	19. April 2022

Neue Stimmen und Sprachen wurden für NTTS hinzugefügt	Amazon Polly bietet jetzt die spanische (mexikanische) Stimme Mia als NTTS-Stimme an. Eine neue Sprache, Katalanisch, wurde zusammen mit der NTTS-Stimme Arlet hinzugefügt. Eine Liste der NTTS-Stimmen finden Sie unter Neuronale Stimmen.	22. März 2022
Neue Stimme für NTTS hinzugefügt	Amazon Polly bietet jetzt die japanische Stimme Takumi als NTTS-Stimme an. Eine Liste der NTTS-Stimmen finden Sie unter Neuronale Stimmen.	6. Dezember 2021
Neue Stimme für NTTS hinzugefügt	Amazon Polly bietet jetzt die französische Stimme Léa als NTTS-Stimme an. Eine Liste der NTTS-Stimmen finden Sie unter Neuronale Stimmen.	18. November 2021
Neue Stimmen wurden für NTTS hinzugefügt	Amazon Polly bietet jetzt die italienische Stimme Bianca und die europäische spanische Stimme Lucia als NTTS-Stimmen an. Eine Liste der NTTS-Stimmen finden Sie unter Neuronale Stimmen.	8. November 2021
Neue Stimme für NTTS hinzugefügt	Amazon Polly bietet jetzt eine neue südafrikanische englische Stimme, Ayanda. Die Stimme ist nur als NTTS-Stimme verfügbar. Eine Liste der NTTS-Stimmen finden Sie unter Neuronale Stimmen.	1. September 2021

Neue Region hinzugefügt	Amazon Polly ist jetzt in der AWS Region Afrika (Kapstadt) verfügbar. Diese Region unterstützt neuronales TTS (NTTS). Weitere Informationen finden Sie unter Funktions - und Regionskompatibilität für eine Liste der Regionen, die NTTS unterstützen.	1. September 2021
Neue Sprache und Stimme hinzugefügt	Amazon Polly unterstützt jetzt Neuseeland Englisch (en-NZ). Eine neue NTTS-Stimme, Aria, spricht Neuseeland Englisch und eine Auswahl von Maori-Wörtern.	24. August 2021
Neues Feature	Amazon Polly macht den Konversations-Sprechstil zur Standardversion für die neuronalen Matthew- und Joanna-Stimmen. Wir haben Verweise auf den Konversat ions-Sprechstil entfernt.	28. Juni 2021
Neue Stimme für NTTS hinzugefügt	Amazon Polly bietet jetzt die deutsche Stimme Vicki als NTTS-Stimme an.	15. Juni 2021

Neue Stimme hinzugefügt	Eine neue weibliche Stimme, Gabrielle, wurde dem Gebietsschema Französisch (Kanada) (fr-CA) hinzugefügt. Die Stimme ist von hoher Qualität und nur als NTTS-Stimme verfügbar. Wie alle neuronalen Stimmen ist sie nur in bestimmten Regionen verfügbar. Eine Liste der Regionen finden Sie unter Kompatibilität von Funktionen und Regionen.	1. Juni 2021
Neue Stimme für NTTS hinzugefügt	Amazon Polly bietet jetzt die koreanische Stimme Seoyeon als NTTS-Stimme an.	11. Mai 2021
Neue Region für NTTS hinzugefügt	Amazon Polly unterstützt jetzt neuronales TTS (NTTS) in der Region Kanada (Zentral) . AWS Weitere Informationen finden Sie unter Funktions - und Regionalkompatibilität für NTTS.	17. März 2021

Neue Stimme im Newscaster-Stil verfügbar	Zusätzlich zu den Stimmen von Matthew, Joanna und Lupe für den Newscaster-Sprechstil bietet Amazon Polly jetzt eine zusätzliche Option für diesen Sprechstil. Mithilfe der Neural Engine können Sie die Amy-Stimme in britische m Englisch für den Newscaste r-Stil verwenden. Weitere Informationen finden Sie unter NTTS-Sprechstile.	10. November 2020
Neue Regionen für NTTS hinzugefügt	Zusätzlich zu den bestehend en Regionen für NTTS (us-east-1, us-west-2, eu-west-1 und ap-southeast-2) werden neuronale Stimmen nun in vier weiteren Regionen unterstüt zt: (ap-northeast-1 (Tokio), ap-southeast-1 (Singapur), eu-central-1 (Frankfurt) und eu-west-2 (London). Weitere Informationen finden Sie unter Funktions- und Regionalk ompatibilität für NTTS.	3. September 2020

Neue Stimme hinzugefügt	Neben den Kinderstimmen „Ivy“ und „Justin“ wurde dem US-Englisch (de-US) als neue männliche Kindesstimme „Kevin“ hinzugefügt. Diese neue Stimme ist sehr hochwertig und nur als NTTS-Stimme verfügbar. Wie alle neuronalen Stimmen wird sie nur in vier Regionen unterstützt: us-east-1 (Nord-Virginia), us-west-2 (Oregon), eu-west-1 (Irland) und ap-southeast-2 (Sydney). Weitere Informationen finden Sie unter NTTS-Stimmen.	16. Juni 2020
Neue Stimme für Nachrichtensprecher verfügbar	Zusätzlich zu den Stimmen von Matthew und Joanna für den Newscaster-Sprechstil bietet Amazon Polly jetzt eine zusätzliche Option für diesen Sprechstil. Mit der neuronalen Engine können Sie die Stimme Lupe auf Spanisch (Amerikanisch) für den Newscaster-Stil verwenden. Weitere Informationen finden Sie unter NTTS-Sprechstile.	16. April 2020

Neues Feature	Zusätzlich zum Newscaster-Sprechstil bietet Amazon Polly jetzt einen zweiten NTTS-Sprechstil, mit dem Sie noch bessere Text-to-Speech-Passagen synthetisieren können. Der Conversational-Stil verwendet das neuronale System, um Sprache in einem freundlicheren und expressiveren Konversationsstil zu generieren, der in vielen Anwendungsfällen verwendet werden kann. Weitere Informationen finden Sie unter NTTS-Sprechstile.	25. November 2019
Neue Stimmen wurden hinzugefügt	Zwei neue Stimmen wurden hinzugefügt: Camila (weiblich, Portuguese-Brazil) und Lupe (weiblich, Spanish-US).	23. Oktober 2019
Neue Funktion hinzugefügt	Hinzufügen des Amazon Polly for Windows-Plug-ins, um die gesamte Palette der Amazon Polly Polly-Stimmen in SAPI-compliant Windows-Anwendungen zu integrieren.	26. September 2019

Wichtiges neues Feature	Zusätzlich zu den standardmäßigen Text-to-Speech-Stimmen (TTS), die Amazon Polly seit seiner Markteinführung unterstützt, bietet Amazon Polly jetzt ein verbessertes neuronales TTS-System (NTTS), das Stimmen von noch höherer Qualität liefern kann und Ihnen so natürliche und menschenähnliche Text-to-Speech-Stimmen bietet. Weitere Informationen finden Sie unter Neural. Text-to-Speech	30. Juli 2019
Neue Stimmen wurden hinzugefügt	Neue Stimmen hinzugefügt: Lucia (weiblich, spanisch) und Bianca (weiblich, Italienisch).	2. August 2018
Neue Sprache hinzugefügt	Neue Sprache hinzugefügt: mexikanisches Spanisch (es-MX). Diese Sprache verwendet die weibliche Stimme von Mia.	2. August 2018
Neue Sprache hinzugefügt	Neue Sprache hinzugefügt: Hindi (hi-IN). Diese Stimme verwendet die weibliche Stimme von Aditi, die auch für indisches Englisch verwendet wird. Damit ist Aditi Amazon Pollys erste zweisprachige Stimme.	2. August 2018

Neue Funktion hinzugefügt	Hinzufügen von Sprachsynthese von langen Textpassagen (bis zu 100.000 kostenpflichtige Zeichen).	17. Juli 2018
Neue SSML-Funktion hinzugefügt	Hinzufügen von Maximale Dauer der generierten Sprachausgabe.	17. Juli 2018
Neue Stimme hinzugefügt	Neue Stimme wurde hinzugefügt: Léa (weiblich, Französisch).	5. Juni 2018
Regionale Erweiterung	Ausweitung des Amazon Polly Polly-Service auf alle Handelsregionen.	4. Juni 2018
Neue Sprache hinzugefügt	Neue Sprache hinzugefügt: Koreanisch (ko-KR).	4. Juni 2018
Erweiterte Funktion	Die Amazon Polly WordPress Polly-Plug-in-Funktion, einschließlich zusätzlicher Amazon Translate Translate-Funktionen.	4. Juni 2018
Neue Stimmen hinzugefügt	Zwei neue Stimmen wurden hinzugefügt: Aditi (weiblich, indisches Englisch) und Seoyeon (weiblich, koreanisch).	15. November 2017
Neues Feature	Hinzufügen einer neuen Sprachmarkierungs-Funktion sowie Erweitern der SSML-Funktionen.	19. April 2017

Neues Handbuch	Dies ist die erste Version des Amazon Polly Developer Guide.	30. November 2016

Die vorliegende Übersetzung wurde maschinell erstellt. Im Falle eines Konflikts oder eines Widerspruchs zwischen dieser übersetzten Fassung und der englischen Fassung (einschließlich infolge von Verzögerungen bei der Übersetzung) ist die englische Fassung maßgeblich.