Amazon Polly

개발자 안내서

Amazon Polly: 개발자 안내서

Table of Contents

Amazon Polly란 무엇입니까?

Amazon Polly는 텍스트를 생생한 음성으로 변환하는 클라우드 서비스입니다. Amazon Polly을 사용하여 참여와 접근성을 높여주는 애플리케이션을 개발할 수 있습니다. Amazon Polly은 여러 언어를 지원하며 다양한 실제 음성이 포함되어 있어서 여러 위치에서 고객에게 적합한 음성을 사용하는 음성 지원 애플리케이션을 개발할 수 있습니다. Amazon Polly에서는 합성한 텍스트에 대해서만 비용을 지불합니다. 또한 Amazon Polly에서 생성된 음성을 캐시하여 추가 요금 없이 재생할 수 있습니다.

Amazon Polly의 일반적인 사용 사례에는 뉴스리더, 게임, 전자 학습 플랫폼, 시각 장애가 있는 사람을 위한 접근성 애플리케이션, 빠르게 성장하는 사물 인터넷(IoT) 세그먼트 등과 같은 모바일 애플리케이션이 포함되나 이에 국한되지 않습니다.

Amazon Polly는 HIPAA(Health Insurance Portability and Accountability Act of 1996) 및 GovCloud 규제 워크로드에서 사용하도록 인증되었습니다.

Amazon Polly는 신용카드 업계의 데이터 보안 표준(PCI DSS) 또는 FedRAMP에서 사용하도록 인증되지 않았습니다.

Amazon Polly를 사용하면 다음과 같은 이점이 있습니다.

- 우수한 품질 – Amazon Polly에서는 동급 최고의 TTS(Text-to-Speech) 기술을 사용하여 자연 음성을 정확한 발음으로 합성합니다(약어, 두문자어 확장, 날짜/시간 해석, 명확한 동음이의어 구별 포함).

- 짧은 지연시간 – Amazon Polly에서는 빠른 응답 시간을 보장하므로 대화 시스템과 같은 짧은 지연시간을 요구하는 사용 사례에 실행 가능한 옵션입니다.

- 다양한 언어 및 음성 포트폴리오 지원 – Amazon Polly에서는 수십 가지 음성과 다양한 언어를 지원하며 대부분의 언어에서 남성 및 여성 음성 옵션을 제공합니다.

- 비용 효율성 – Amazon Polly의 사용별 지불 모델은 설정 비용이 없습니다. 따라서 소규모로 시작하여 애플리케이션의 성장에 따라 규모를 확장할 수 있습니다.

- 클라우드 기반 솔루션 – 디바이스 기반 Text-to-Speech 솔루션을 사용하려면 충분한 컴퓨팅 리소스, 매우 높은 CPU 전력, RAM 및 디스크 공간이 필요합니다. 따라서 개발 비용이 높고 태블릿, 스마트폰 등과 같은 디바이스에서 전력 소비가 높아집니다. 반면에 클라우드에서 수행되는 Text-to-Speech 변환은 로컬 리소스 요구사항을 획기적으로 줄여줍니다. 따라서 모든 사용 가능한 언어와 음성을 최상의 품질로 지원할 수 있습니다. 또한 음성 개선 사항이 모든 최종 사용자에게 즉시 제공되므로 디바이스에 대한 추가 업데이트가 필요하지 않습니다.

Amazon Polly를 처음 사용하십니까?

Amazon Polly를 처음 사용하는 경우, 먼저 다음 단원을 나열된 순서대로 읽어보십시오.

1. Amazon Polly: 사용 방법 (p. 3) - 이 섹션에서는 종단 간 경험 생성에 사용 가능한 다양한 Amazon Polly 입력 및 옵션을 소개합니다.
2. Amazon Polly로 시작하기 (p. 6) - 이 섹션에서는 계정을 설정하고 Amazon Polly 스피치 합성을 테스트합니다.

3. 예제 애플리케이션 (p. 103) – 이 섹션에서는 Amazon Polly 탐색에 사용할 수 있는 추가 예제를 제공합니다.

Amazon Polly: 사용 방법

Amazon Polly는 입력 텍스트를 실제 같은 음성으로 변환합니다. `SynthesizeSpeech` 메서드를 호출하고, 합성하려는 텍스트를 제공한 다음 사용 가능한 TTS(Text-to-Speech) 음성 중 하나를 선택하고 오디오 출력 형식만 선택하면 됩니다. Amazon Polly에서 제공된 텍스트를 고품질 음성 오디오 스트림으로 합성합니다.

- 입력 텍스트 – 합성하려는 텍스트를 제공하고 Amazon Polly에서 오디오 스트림을 반환합니다. 입력은 일반 텍스트 또는 SSML(Speech Synthesis Markup Language) 형식으로 제공할 수 있습니다. SSML을 사용하면 발음, 볼륨, 피치 및 말하기 속도와 같은 다양한 음성 측면을 제어 할 수 있습니다. 자세한 내용은 SSML 사용 (p. 62) 섹션을 참조하십시오.

- 사용가능한 음성 – Amazon Polly에서는 다양한 언어의 포트폴리오와 다양한 음성을 제공합니다. 대부분의 언어에서 남성과 여성을 포함한 여러 가지 음성 중에서 선택할 수 있습니다. `SynthesizeSpeech` 작업을 호출 할 때 음성 이름만 지정하면 서비스에서 이 음성을 사용하여 텍스트를 음성으로 변환합니다. Amazon Polly는 번역 서비스는 아닙니다. 합성된 스피치는 텍스트 언어입니다. 숫자를 사용하는 수(예: 오십삼이 아닌 53)는 음성 언어로 합성됩니다.

- 출력 형식 – Amazon Polly는 여러 가지 형식으로 합성된 스피지를 전달할 수 있습니다. 사용자의 요구 사항에 맞는 음성 형식을 선택할 수 있습니다. 예를 들어, 웹 및 모바일 애플리케이션에서 사용하도록 MP3 또는 Ogg Vorbis 형식의 스피치를 요청할 수 있습니다. 또는 AWS IoT 장치 및 전화 솔루션에 대한 PCM 출력 형식을 요청할 수도 있습니다.

다음 단계

Amazon Polly을 처음 사용하는 경우, 먼저 다음 항목을 순서대로 읽어보십시오.

- Amazon Polly로 시작하기 (p. 6)
- 예제 애플리케이션 (p. 103)
- Amazon Polly 제한 (p. 130)

일반적인 질문

이 항목에서는 Amazon Polly에 대해 자주 묻는 질문의 답변을 제공합니다.

항목

일반 질문

문: 음성 합성을 저장할 수 있습니까?

합성 출력을 로컬 시스템에 저장할 수 있습니다. Amazon Polly를 호출하여 암호화 키로 파일을 암호화한 후 Amazon Simple Storage Service(Amazon S3)나 다른 보안 스토리지에 저장할 수도 있습니다. Amazon Polly

`SynthesizeSpeech` 호출은 상태를 저장하지 않으며 고객 ID와 연결되지 않습니다. 나중에 Amazon Polly에서 검색할 수 없습니다.

콘텐츠 렌더링

문: Amazon Polly에서 음성으로 재생할 때 일부 텍스트에서 잘못된 음절에 강세가 옵니다. 양음 악센트(U+0301)를 사용하여 강세를 표시했지만 여전히 잘못된 음절에 강세가 옵니다. 어떻게 해결할 수 있나요?

Amazon Polly는 현재 양음 악센트(U+0301)를 단어의 음절 강세로 인식하지 못합니다. 하지만 두 가지 방법을 사용하여 단어의 강세를 변경할 수 있습니다. IPA 폰과 ssml 태그를 사용하여 단어 발음을 변경할 수 있습니다. 자세한 내용은 Amazon Polly의 SSML 태그 (p. 68) 단원을 참조하십시오. 일부 언어에서는 음절 바로 뒤에 아포스트로피를 사용하여 강세 변화를 나타낼 수도 있습니다. 예를 들어 러시아어에서 стра́ны 단어와 стра́ны 단어는 여기서 양음 악센트로 표시한 대로 강세가 다릅니다. 하지만 철자가 동일하기 때문에 Amazon Polly는 표준 언어 사용에 따라 둘 다 마지막 음절에 강세를 주어 발음합니다. стран'ы와 같이 아포스트로피를 사용하여 강세 음절을 표시하면 Amazon Polly가 해당 음절에 강세를 줍니다.

문: 텍스트에 글머리표를 사용할 경우 Amazon Polly가 제대로 옮기지 못합니다. 글머리표가 나올 때마다 "마이너스"라고 합니다. 어떻게 해야 합니까?

글머리표 대신 "-"(하이픈)을 사용하면 일부 언어에서는 Amazon Polly가 이 하이픈을 빼기 부호로 옮깁니다. 글머리표로 하이픈을 사용하려면 어휘 항목을 사용합니다. 자세한 내용은 어휘 관리 (p. 84) 단원을 참조하십시오.

문: "and/or" 그리고 "yes/no" 등과 같이 "/"(슬래시) 기호를 텍스트에 자주 사용합니다. 이 경우 Amazon Polly가 어떻게 옮깁니까?

영어에서 Amazon Polly는 "and/or"를 "and or"로 옮깁니다. 현재 이 규칙은 다른 언어에서는 사용할 수 없습니다. 영어 외 언어에서는 Amazon Polly가 "yes/no"를 "yes 슬래시 no"로 옮깁니다. 이렇게 바뀌지 않게 하려면 어휘 항목을 사용하면 됩니다. 자세한 내용은 어휘 관리 (p. 84) 단원을 참조하십시오.

문: 기존 소스의 텍스트를 사용하여 Linux 머신에서 AWS CLI를 통해 음성 합성을 하는 경우 일부 UTF-8 문자가 콘솔을 사용하면 제대로 처리되는 것 같은데 Amazon Polly에서는 제대로 처리되지 않는 것 같습니다. 무슨 이유입니까?

Unix 셸에서 유니코드를 처리하는 방식 때문이며 Amazon Polly의 문제가 아닙니다. 두 가지 방법을 사용할 수 있습니다. 문제의 문자를 찾아서 입력 텍스트에서 바꾸거나, PHP 인터페이스와 같이 이 문제 없이 Amazon Polly를 사용할 수 있는 다른 방법을 이용합니다. 이 문제는 현재 해결 중인 알려진 문제이며, 자주 사용하지 않는 극히 일부 유니코드 문자에서만 발생합니다.

문: 국제 음성 기호(IPA)를 포함하는 소스의 텍스트를 합성하려는 경우 Amazon Polly가 이를 인식하지 못하고 일부 기호를 발음합니다. 어떻게 해결하나요?

SSML(Speech Synthesis Markup Language)을 사용하여 정의하지 않는 한 Amazon Polly는 IPA 기호를 인식하지 못합니다. 하지만 일반적으로 IPA 기호에서 판독기의 발음 지침을 나타내는 섹션은 적으므로, 대부분의 경우 단순 삭제하여 입력 텍스트에서 안전하게 제거할 수 있습니다. 어휘 목록을 사용하여 Amazon Polly에서 이를 렌더링하는 방식을 변경할 수도 있습니다. 자세한 내용은 SSML 사용 (p. 62) 및 어휘 관리 (p. 84) 단원을 참조하십시오.

음성

문: 사용 여부를 확인하기 위해 음성을 들어볼 수 있습니까?

Amazon Polly 콘솔을 통해 짧은 텍스트를 합성하거나, AWS CLI에서 `synthesize-speech` API를 사용하여 Amazon Polly가 지원하는 음성을 들을 수 있습니다. 콘솔에서 지원되는 음성을 선택하고, 샘플 텍스트를 실행하여 음성을 들을 수 있습니다. 자세한 내용은 연습 1: 스피치 합성 빠른 시작(콘솔) (p. 8) 단원을 참조하십시오. AWS CLI에서 `synthesize-speech` API를 실행할 때 들으려는 음성의 `voice-id`를 사용할 수 있습니다. 자세한 내용은 3.2 단계: AWS CLI 사용 연습 시작하기 (p. 10) 단원을 참조하십시오.

문: 음성은 얼마 동안 사용할 수 있습니까? 지금 사용하기 위해 음성을 선택하는 경우 5년 후에도 사용할 수 있습니까?

지속적인 고객 지원을 위해 음성을 폐기할 계획은 없습니다 현재 사용 가능한 음성과 앞으로의 음성 모두에 대해 폐기 계획은 없습니다.

데이터 보안 및 비밀 유지

문: 요청 API를 사용한 요청 로깅을 옵트아웃할 수 있습니까?

예. AWS Support에 문의하여 요청할 수 있습니다.

문: 특정 데이터 필드가 저장되지 않도록 숨길 수 있습니까? 예를 들어 일부 기밀 데이터가 포함된 텍스트를 변환할 경우 AWS 시스템에 저장되지 않게 하기 위해 숨길 수 있습니까?

아니요. Amazon Polly는 이 기능을 현재로서는 지원하지 않습니다.

문: Amazon Polly에서 사용하려는 텍스트가 기밀 사항입니다. 내 데이터를 보호할 수 있습니까?

제출된 모든 텍스트는 전송되고 저장될 때 다음을 사용하여 SSL(Secure Sockets Layer)로 보호됩니다. RSA 암호화. 서비스 로그와 텍스트를 별도로 관리하므로 콘텐츠를 고객 ID와 연관시키는 것을 불가능합니다. 따라서 Amazon Polly는 제출된 텍스트를 고객 ID와 연관짓지 않습니다.

문: 데이터는 얼마 동안 유지됩니까?

Amazon Polly는 데이터를 14일 동안 유지합니다. 14일이 지나면 시스템에서 자동으로 삭제됩니다.

문: 데이터를 원래 보존 기간보다 일찍 지우도록 요청할 수 있습니까?

예. AWS Support에 문의하여 요청할 수 있습니다.

Amazon Polly로 시작하기

Amazon Polly는 기존 애플리케이션과 쉽게 통합할 수 있는 간단한 API 작업을 제공합니다. 지원되는 작업 목록은 Actions (p. 137) 항목을 참조하십시오. 다음 두 가지 옵션 중 하나를 사용할 수 있습니다.

- AWS SDK – SDK를 사용할 때 Amazon Polly에 대한 요청은 사용자가 제공한 자격 증명을 사용하여 자동으로 서명되고 인증됩니다. 이것은 애플리케이션을 구축할 때 권장되는 선택 사항입니다.
- AWS CLI – AWS CLI를 사용하여 코드를 작성하지 않고 Amazon Polly 기능에 액세스할 수 있습니다.

다음 섹션에서는 설정 및 입문용 실습을 제공하는 방법에 대해 설명합니다.

항목

1단계: AWS 계정 설정 및 사용자 만들기

Amazon Polly을 처음 사용하기 전에 다음 작업을 완료해야 합니다.

1. 1.1단계: AWS에 가입 (p. 6)
2. 2단계: IAM 사용자 생성 (p. 7)

1.1단계: AWS에 가입

Amazon Web Services(AWS)에 가입하면 Amazon Polly을 포함해 AWS의 모든 서비스에 AWS 계정이 자동으로 등록됩니다. 사용한 서비스에 대해서만 청구됩니다.

Amazon Polly에서는 사용한 리소스에 대해서만 비용을 지불합니다. AWS를 처음 사용하는 고객인 경우 Amazon Polly를 무료로 시작할 수 있습니다. 자세한 내용은 AWS 프리 티어를 참조하십시오.

이미 AWS 계정이 있다면 다음 단계로 건너뛰십시오. AWS 계정이 없는 경우에는 다음 절차의 단계를 수행하여 계정을 만듭니다.

AWS 계정을 만들려면 다음을 수행합니다.

1. https://aws.amazon.com/을 열고 [Create an AWS Account]를 선택합니다.

   Note

   이전에 AWS Management 콘솔에 로그인한 적이 있을 경우 이를 브라우저에서 사용하지 못할 수 있습니다. 그와 같은 경우 [Sign in to a different account]를 선택한 다음, [Create a new AWS account]를 선택합니다.
2. 온라인 지시 사항을 따릅니다.

등록 절차 중 전화를 받고 전화 키패드를 사용하여 PIN을 입력하는 과정이 있습니다.

다음 단계에 필요하므로 AWS 계정 ID를 기록합니다.

2단계: IAM 사용자 생성

Amazon Polly 등의 AWS 서비스에 액세스하려면 해당 서비스가 소유한 리소스에 대한 액세스 권한이 있는지 확인하기 위해 자격 증명을 제공해야 합니다. 콘솔은 암호를 요구합니다. AWS 계정에 대한 액세스 키를 생성하면 AWS CLI 또는 API에 액세스할 수 있습니다. 그러나 AWS 계정용 자격 증명을 사용하여 AWS에 액세스하지 않는 것이 좋습니다. 대신 AWS Identity and Access Management(IAM)을 사용하는 것이 좋습니다. 관리자 권한을 사용하여 IAM 사용자를 생성하고 IAM 그룹에 추가한 다음 생성한 IAM 사용자에게 관리자 권한을 부여합니다. 그러면 특정 URL과 해당 IAM 사용자의 자격 증명을 사용하여 AWS에 액세스할 수 있습니다.

AWS에 가입했지만 IAM 사용자를 만들지 않은 경우, IAM 콘솔을 사용하여 사용자를 만들 수 있습니다.

이 가이드의 시작하기 연습에서는 관리자 권한이 있는 사용자(`adminuser`)가 있다고 가정합니다. 절차에 따라 계정에서 `adminuser`를 만듭니다.

관리자 사용자를 만들고 콘솔에 로그인하려면

1. AWS 계정에서 `adminuser`라는 관리자 사용자를 만듭니다. 관련 지침은 IAM 사용 설명서의 IAM 사용자와 관리자 그룹 처음 만들기를 참조하십시오.
2. 사용자는 특정 URL을 사용하여 AWS Management 콘솔에 로그인할 수 있습니다. 자세한 내용은 IAM 사용 설명서의 사용자의 계정 로그인 방법을 참조하십시오.

   Important

   관리자 자격증명을 사용하는 연습 시작하기 보안 강화를 위해 프로덕션 애플리케이션을 구축하고 테스팅할 때 Amazon Polly 작업에 대해서만 권한이 있는 서비스 관련 관리자를 생성하는 것이 좋습니다. Amazon Polly 특정 권한을 부여하는 정책에 대한 예제는 예제 1: 모든 Amazon Polly 작업 허용 (p. 164) 항목을 참조하십시오.

IAM에 대한 자세한 내용은 다음을 참조하십시오.

- AWS Identity and Access Management(IAM)
- 시작하기
- IAM 사용 설명서

다음 단계

3단계: 콘솔 사용 시작하기 (p. 7)

3단계: 콘솔 사용 시작하기

Amazon Polly 콘솔은 Amazon Polly의 스피치 합성을 테스트하고 사용하기 위한 가장 간단한 방법입니다. Amazon Polly 콘솔은 일반 텍스트 또는 SSML 입력에서 스피치 합성을 지원합니다.

항목

연습 1: 스피치 합성 빠른 시작(콘솔)

빠른 시작은 스피치 품질을 위한 Amazon Polly 스피치 합성을 테스트하는 가장 빠른 방법을 안내합니다. [Text-to-Speech] 탭을 선택하면 텍스트를 입력하기 위한 텍스트 필드에 예제 텍스트가 미리 로드되어 Amazon Polly를 신속하게 사용해 볼 수 있습니다.

Amazon Polly를 빠르게 테스트하려면

1. AWS Management Console에 로그인한 후 https://console.aws.amazon.com/polly/에서 Amazon Polly 콘솔을 엽니다.
2. [Text-to-Speech] 탭을 선택합니다.
3. (옵션)[SSML]을 선택합니다.
4. 언어 및 리전을 선택한 다음 음성을 선택합니다.
5. [Listen to speech]을 선택합니다.

심층 테스트에 대한 자세한 내용은 다음 항목을 참조하십시오.

- 연습 2: 스피치 합성(일반 텍스트 입력) (p. 8)
- Amazon Polly 콘솔과 함께 SSML 사용 (p. 62)
- 콘솔을 사용하여 어휘 적용(스피치 합성) (p. 86)

연습 2: 스피치 합성(일반 텍스트 입력)

다음 절차에서는 일반 텍스트를 입력하여 스피치를 합성합니다. "W3C"와 날짜 "10/3"(10월 3일)이 어떻게 합성되는지 확인하십시오.

일반 텍스트를 입력하여 스피치를 합성하려면

1. Amazon Polly 콘솔에 로그인하고 [Get started]를 선택한 다음 [Text-to-Speech] 탭을 선택합니다.
2. [Plain text] 탭을 선택합니다.
3. 이 텍스트를 입력란에 입력하거나 붙여 넣습니다.

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

4. [Choose a language and region]에서 [English US]를 선택한 다음 이 텍스트에 사용할 음성을 선택합니다.
5. 스피치를 즉시 들으려면 [Listen to speech]를 선택합니다.
6. 스피치를 파일로 저장하려면 다음 중 한 가지를 실행합니다.
   a. [Save speech to MP3]를 선택합니다.
   b. 다른 파일 형식으로 변경하려면 [Change file format]을 선택하여 원하는 파일 형식을 선택한 다음 [Change]를 선택합니다.

심층 예제에 대한 자세한 내용은 다음 항목을 참조하십시오.

- 콘솔을 사용하여 어휘 적용(스피치 합성) (p. 86)
- Amazon Polly 콘솔과 함께 SSML 사용 (p. 62)

다음 단계

3단계: AWS CLI 사용 시작하기 (p. 9)

3단계: AWS CLI 사용 시작하기

AWS CLI를 사용하면 Amazon Polly 콘솔을 사용하여 수행할 수 있는 거의 모든 Amazon Polly 작업을 수행할 수 있습니다. AWS CLI를 사용하여 합성된 스피치를 들을 수 없습니다. 대신 파일에 저장한 다음 파일을 재생할 수 있는 애플리케이션에서 파일을 열어야 합니다.

항목

- 3.1단계: AWS Command Line Interface(AWS CLI) 설정 (p. 9)
- 3.2 단계: AWS CLI 사용 연습 시작하기 (p. 10)

3.1단계: AWS Command Line Interface(AWS CLI) 설정

단계에 따라 AWS Command Line Interface(AWS CLI)를 다운로드하고 구성합니다.

> **Important**
>
> 이 시작하기 연습에 있는 단계를 수행하는 데 AWS CLI가 필요하지 않습니다. 하지만 이 가이드의 일부 연습에서는 AWS CLI를 사용합니다. 이 단계를 건너뛰어 3.2 단계: AWS CLI 사용 연습 시작하기 (p. 10)으로 이동한 다음 나중에 필요할 때 AWS CLI를 설정할 수 있습니다.

AWS CLI를 설정하려면

1. AWS CLI를 다운로드하고 구성합니다. 관련 지침은 AWS Command Line Interface 사용 설명서에서 다음 항목을 참조하십시오.
   - AWS Command Line Interface 설정
   - AWS Command Line Interface 구성
2. AWS CLI 구성 파일에서 관리자 사용자의 명명된 프로필을 추가합니다. 이 프로필은 AWS CLI 명령을 실행할 때 사용합니다. 명명된 프로필에 대한 자세한 내용은 AWS Command Line Interface 사용 설명서의 명명된 프로필을 참조하십시오.

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

   Amazon Polly가 지원하는 사용 가능한 AWS 리전 목록은 Amazon Web Services 일반 참조의 리전 및 엔드포인트를 참조하십시오.

   AWS CLI를 구성할 때 Amazon Polly 지원 리전 중 하나를 지정하면 AWS CLI 코드 예제에서 다음 줄을 생략 할 수 있습니다. AWS CLI 구성에서 Amazon Polly에서 지원하지 않는 리전을 지정하면 예를 들어

Amazon Polly를 지원하지 않는 리전의 다른 서비스를 사용하는 기존 AWS 고객인 경우 다음 행을 포함해야 합니다.

```
--region polly-supported-aws-region
```

3. 명령 프롬프트에서 다음 help 명령을 입력하여 설정을 확인합니다.

```
aws help
```

유효한 AWS 명령 목록이 AWS CLI 창에 나타납니다.

AWS CLI에서 Amazon Polly를 활성화하려면(선택 사항)

이전에 AWS CLI를 다운로드하여 구성한 경우 Amazon Polly에서는 AWS CLI를 재구성하지 않고 사용할 수 없습니다. 이 절차는 해당 사항이 필요한지 여부를 확인하고 Amazon Polly가 자동으로 사용할 수 없는 경우 지침을 제공합니다.

1. 명령 프롬프트에서 다음 help 명령을 입력하여 Amazon Polly의 사용 가능 여부를 확인하십시오.

```
aws polly help
```

Amazon Polly에 대한 설명과 유효한 명령 목록이 표시되면 AWS CLI 창에 나타납니다. Amazon Polly가 AWS CLI에서 사용 가능하며 바로 사용될 수 있습니다. 이 경우 이 절차의 나머지 부분은 건너뛸 수 있습니다. 이 메시지가 표시되지 않으면 2단계로 진행합니다.

2. Amazon Polly를 사용하려면 다음 두 옵션 중 하나를 사용하십시오.

   a. AWS CLI를 제거하고 다시 설치하십시오.

      지침은 AWS Command Line Interface 사용 설명서: AWS Command Line Interface 설치의 다음 항목을 참조하십시오.

      또는

   b. Download the file service-2.json 파일 다운로드

      프롬프트에서 다음 명령을 실행합니다.

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

3. Amazon Polly의 가용성 재확인

```
aws polly help
```

Amazon Polly에 대한 설명이 표시되어야 합니다.

다음 단계

3.2 단계: AWS CLI 사용 연습 시작하기 (p. 10)

3.2 단계: AWS CLI 사용 연습 시작하기

이제 Amazon Polly에서 제공하는 스피치 합성을 테스트할 수 있습니다. 이 연습에서는 샘플 텍스트를 전달하여 `SynthesizeSpeech` 작업을 호출합니다. 결과 오디오를 파일로 저장하고 콘텐츠를 확인할 수 있습니다.

AWS CLI를 구성할 때 Amazon Polly 지원 리전 중 하나를 지정하면 AWS CLI 코드 예제에서 다음 줄을 생략할 수 있습니다. AWS CLI 구성에서 Amazon Polly에서 지원하지 않는 리전을 지정하면 예를 들어 Amazon Polly를 지원하지 않는 리전의 다른 서비스를 사용하는 기존 AWS 고객인 경우 다음 행을 포함해야 합니다.

```
--region polly-supported-aws-region
```

1. `synthesize-speech` AWS CLI 명령을 실행하여 샘플 텍스트를 오디오 파일 (`hello.mp3`)에 합성합니다.

   다음은 Unix, Linux, macOS용 형식으로 지정된 AWS CLI 예제입니다. Windows의 경우 각 줄 끝에 있는 백슬래시(\) Unix 연속 문자를 캐럿(^)으로 바꿉니다. 입력 텍스트는 큰 따옴표(")로 감싸고 내부 태그에는 작은 따옴표(')를 사용합니다.

   ```
   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
   ```

   `synthesize-speech` 호출에서 합성을 위한 샘플 텍스트, 사용할 음성(다음 3단계에서 설명하는 음성 ID 제공)과 출력 형식을 제공합니다. 이 명령은 결과 오디오를 `hello.mp3` 파일에 저장합니다.

   위의 작업은 MP3 파일 외에도 콘솔에 다음과 같은 출력을 생성합니다.

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

2. `hello.mp3` 결과 파일을 재생하여 합성된 스피치를 확인합니다.
3. `DescribeVoices` 작업을 사용해 사용 가능한 음성 목록을 가져올 수 있습니다. `describe-voices` AWS CLI 명령을 실행합니다.

   ```
   aws polly describe-voices
   ```

   이에 대한 응답으로 Amazon Polly에서는 사용 가능한 모든 음성 목록을 반환합니다. 각 음성에 대해 응답으로 음성 ID, 언어 코드, 언어 이름 및 성별과 같은 메타데이터를 제공합니다. 다음은 응답 예입니다.

   ```
   {
       "Voices": [
           {
               "Gender": "Female",
               "Name": "Salli",
               "LanguageName": "US English",
               "Id": "Salli",
               "LanguageCode": "en-US"
           },
           {
               "Gender": "Female",
               "Name": "Joanna",
               "LanguageName": "US English",
               "Id": "Kendra",
               "LanguageCode": "en-US"
           }
       ]
   }
   ```

필요에 따라 언어 코드를 지정하여 특정 언어에 사용 가능한 음성을 찾을 수 있습니다. Amazon Polly는 수십 가지의 음성을 지원합니다. 다음 예에서는 브라질 포르투갈어에 대한 모든 음성이 나열됩니다.

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

언어 코드의 목록은 DescribeVoices (p. 140)를 참조하십시오. 이러한 언어 코드는 W3C 언어 식별 태그(*## ### ## ISO 639 ##-ISO 3166 ## ##*)입니다. 예로, en-US(미국 영어), en-GB(영국 영어) 및 es-ES(스페인어) 등입니다.

AWS CLI에서 `help` 옵션을 사용하여 언어 코드 목록을 얻을 수도 있습니다.

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

다음 단계

이 가이드는 추가 예제를 제공하며 일부는 AWS SDK for Python (Boto)를 사용하여 Amazon Polly에 API를 호출하는 Python 코드 예제입니다. Python을 설정하고 다음 섹션에서 제공하는 예제 코드 테스트를 권장합니다. 추가 예제는 다음(예제 애플리케이션 (p. 103))을 참조하십시오.

Python 설정 및 예제 테스트

Python 예제 코드를 테스트하려면 AWS SDK for Python (Boto)이 필요합니다. 자세한 방법은 Python용 AWS SDK(Boto3)를 참조하십시오.

예제 Python 코드를 테스트하려면

다음 Python 코드 예제에서는 다음과 같이 작업을 수행합니다.

- AWS SDK for Python (Boto)를 사용하여 간단한 텍스트를 입력으로 제공하는 Amazon Polly에 `SynthesizeSpeech` 요청을 보냅니다.
- 응답에서 결과 오디오 스트림에 액세스하고 오디오를 로컬 디스크의 파일(`speech.mp3`)에 저장합니다.
- 로컬 시스템의 기본 오디오 플레이어로 오디오 파일을 재생합니다.

코드를 파일(example.py)에 저장하고 실행합니다.

```
"""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 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(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 Mac and Linux. (Darwin = mac, xdg-open = linux).
    opener = "open" if sys.platform == "darwin" else "xdg-open"
    subprocess.call([opener, output])
```

예제 애플리케이션을 포함한 추가 예제에 대해서는 예제 애플리케이션 (p. 103)을 참고하십시오.

스피치 마크

스피치 마크는 문장이나 단어가 오디오 스트림에서 시작과 끝의 위치에서 합성하는 스피치를 설명하는 메타데이터입니다. 텍스트에 스피치 마크를 요청하면 Amazon Polly에서 합성된 스피치 대신 이 메타데이터를 반환합니다. 합성 스피치 오디오 스트림과 함께 스피치 마크를 사용하면 애플리케이션에 향상된 시각적 경험을 제공할 수 있습니다.

예를 들어, 메타데이터를 텍스트의 오디오 스트림과 결합하면 스피치를 얼굴 애니메이션(립싱크)과 동기화하거나 쓰여진 단어를 강조해서 표시할 수 있습니다.

항목

스피치 마크 형식

`synthesize-speech`명령에 `speech-mark-types` 옵션을 사용하여 AWS CLI에서 스피치 마크를 요청합니다. 입력 텍스트에서 반환할 메타데이터 요소를 지정합니다. 최대 네 가지 형식의 메타데이터를 요청할 수 있지만, 요청당 반드시 하나 이상을 지정해야 합니다. 요청 시 오디오 출력이 생성되지 않습니다.

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

Amazon Polly에서는 다음 요소를 사용하여 스피치 마크를 생성합니다.

- sentence – 입력 텍스트의 문장 요소를 나타냅니다.
- word – 텍스트의 단어 요소를 나타냅니다.
- viseme – 말하는 각 음소에 대응하는 얼굴과 입의 움직임을 설명합니다. 자세한 내용은 Viseme 및 Amazon Polly (p. 14) 섹션을 참조하십시오.
- ssml – SSML 입력 텍스트에서 <mark> 요소를 설명합니다. 자세한 내용은 SSML 사용 (p. 62) 섹션을 참조하십시오.

Viseme 및 Amazon Polly

viseme은 단어를 말할 때 얼굴과 입의 위치를 나타냅니다. 이것은 단어가 구성되는 기본적인 음향 단위인 음소에 대한 시각적인 효과와 동등한 것을 나타냅니다. Viseme은 스피치의 기본 시각적 빌딩 블록입니다.

각 언어에는 특정 음소에 해당하는 viseme 집합이 있습니다. 언어에서 각 음소는 소리를 형성할 때 입이 만드는 모양을 나타내는 해당 viseme을 가지고 있습니다. 그러나 소리가 달라도 수많은 음소가 동일하게 나타나므로 모든 viseme을 특정 음소에 매핑할 수 있는 것은 아닙니다. 예를 들어 영어에서는 "pet"과 "bet"이라는 단어가 음향적으로 다릅니다. 그러나 소리없이 시각적으로 관찰했을 때 두 단어는 동일한 것처럼 보입니다.

다음 차트에서는 미국 영어 음성에 대한 IPA(International Phonetic Alphabet) 음소와 X-SAMPA(Extended Speech Assessment Methods Phonetic Alphabet) 기호 세트 및 이에 해당하는 viseme이 나와 있습니다.

IPA	X-SAMPA	설명	예	Viseme
자음				
b	b	유성 양순 파열음	bed	p
d	d	유성 치경 파열음	dig	t
ʤ	dZ	유성 후치경 파찰음	jump	S
ð	D	유성 치 마찰음	then	T
f	f	무성 순치 마찰음	five	f
g	g	유성 연구개 파열음	game	k
h	h	무성 성문 마찰음	house	k
j	j	경구개 접근음	yes	i
k	k	무성 연구개 파열음	cat	k
l	l	치경 설측 접근음	lay	t
m	m	양순 비음	mouse	p
n	n	치경 비음	nap	t
ŋ	N	연구개 비음	thing	k
p	p	무성 양순 파열음	speak	p
ɹ	r\	치경 접근음	red	r
s	s	무성 치경 마찰음	seem	s
ʃ	S	무성 후치경 마찰음	ship	S
t	t	무성 치경 파열음	trap	t
ʧ	tS	무성 후치경 파찰음	chart	S
Θ	T	무성 치 마찰음	thin	T
v	v	유성 순치 마찰음	vest	f
w	w	양순 연구개 접근음	west	u
z	z	유성 치경 마찰음	zero	s
ʒ	Z	유성 후치경 마찰음	vision	S
모음				
ə	@	중설 중모음	arena	@
ɚ	@'	중설 치경 접근 수반음	reader	@
æ	{	전설 비원순 근저모음	trap	a
aɪ	aI	이중 모음	price	a

IPA	X-SAMPA	설명	예	Viseme
aʊ	aU	이중 모음	mouth	a
ɑ	A	후설 비원순 저장모음	father	a
eɪ	eI	이중 모음	face	e
ɝ	3'	중설 비원순 중저 치경 접근 수반음	nurse	E
ɛ	E	전설 비원순 중저모음	dress	E
i:	i	전설 비원순 고장모음	fleece	i
ɪ	I	근전설 비원순 근고모음	kit	i
oʊ	oU	이중 모음	goat	o
ɔ	O	후설 원순 중저장모음	thought	O
ɔɪ	OI	이중 모음	choice	O
u	u	후설 원순 고장모음	goose	u
ʊ	U	근후설 원순 근고모음	foot	u
ʌ	V	후설 평순 중저모음	strut	E

사용 가능한 모든 언어를 보려면 지원되는 언어의 음소/Viseme 표 (p. 20) 단원을 참조하십시오.

스피치 마크 사용

스피치 마크 요청

입력 텍스트의 스피치 마크를 요청하려면 `synthesize-speech` 명령을 사용하십시오. 입력 텍스트 외에 이 메타데이터를 반환하려면 다음 요소가 필요합니다.

- `output-format`

  Amazon Polly에서는 스피치 마크를 반환할 때 JSON 형식만 지원합니다.

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

  지원되지 않는 출력 형식을 사용하면 Amazon Polly에서 예외가 발생합니다.

- `voice-id`

  메타데이터가 관련 오디오 스트림과 일치하는지 확인하려면 합성 스피치 오디오 스트림을 생성하는 데 사용되는 것과 동일한 음성을 지정하십시오. 사용 가능한 음성은 동일한 스피치 속도를 아닙니다. 스피치를 생성하는 데 사용된 것과 다른 음성을 사용하는 경우 메타데이터가 오디오 스트림과 일치하지 않습니다.

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

- `speech-mark-types`

원하는 스피치 마크 유형을 지정하십시오. 스피치 마크 형식의 일부 또는 전부를 요청할 수 있지만 적어도 하나의 형식을 지정해야 합니다.

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

- `text-type`

  일반 텍스트는 Amazon Polly의 기본 입력 텍스트이므로 SSML 스피치 마크를 반환하려면 `text-type ssml`을 사용해야 합니다.
- `outfile`

  메타데이터가 기록되는 출력 파일을 지정하십시오.

```
MaryLamb.txt
```

다음은 Unix, Linux, macOS용 형식으로 지정된 AWS CLI 예제입니다. Windows의 경우 각 줄 끝에 있는 백슬래시(\) Unix 연속 문자를 캐럿(^)으로 바꿉니다. 입력 텍스트는 큰 따옴표(")로 감싸고 내부 태그에는 작은 따옴표(')를 사용합니다.

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

스피치 마크 출력

Amazon Polly에서는 줄로 구분된 JSON 스트림에서 스피치 마크 객체를 반환합니다. 스피치 마크 객체는 다음과 같은 필드를 포함합니다.

- time – 해당하는 오디오 스트림의 시작 부분의 타임스탬프(밀리초)
- type – 스피치 마크의 형식(sentence, word, viseme, ssml)
- start – 입력 텍스트에서 객체 시작의 오프셋(바이트)(viseme 마크 포함하지 않음)
- end – 입력 텍스트에서 객체 끝의 오프셋(바이트)(viseme 마크 포함하지 않음)
- value – 스피치 마크 형식에 따라 다양
  - SSML: <mark> SSML 태그
  - viseme: viseme 이름
  - word 또는 sentence: 시작 및 끝 필드로 구분된 입력 텍스트의 하위 문자열

예를 들어 Amazon Polly에서 "Mary have little lamb" 텍스트에서 다음 `word` 스피치 마크 객체를 생성합니다.

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

설명된 단어 "had"는 오디오 스트림이 시작된 후 373밀리초부터 시작하여 입력 텍스트의 5바이트에서 시작하여 8바이트에서 끝납니다.

> **Note**
>
> 이 메타데이터는`Joanna` 음성 ID용입니다. 동일한 입력 텍스트로 다른 음성을 사용하는 경우 메타데이터가 다를 수 있습니다.

스피치 마크 예제

다음과 같은 스피치 마크 요청의 예제는 일반적인 요청을 만드는 방법과 요청이 생성하는 출력을 보여줍니다.

예제 1 : SSML이 없는 스피치 마크

다음 예제는 "Mary have little lamb."이라는 간단한 문장에 대해 화면에서 요청된 메타데이터가 어떻게 표시되는지 보여줍니다. 이 예제에서는 간소화를 위해 SSML 스피치 마크를 포함하지 않습니다.

다음은 Unix, Linux, macOS용 형식으로 지정된 AWS CLI 예제입니다. Windows의 경우 각 줄 끝에 있는 백슬래시(\) Unix 연속 문자를 캐럿(^)으로 바꿉니다. 입력 텍스트는 큰 따옴표(")로 감싸고 내부 태그에는 작은 따옴표(')를 사용합니다.

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

이 요청을 하면 Amazon Polly에서는 .txt 파일에 다음을 반환합니다.

```
{"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"}
```

이 출력에서 텍스트의 각 부분은 스피치 마크 용어로 구분됩니다.

- sentence는 "Mary had a little lamb."입니다.
- 텍스트의 각 word는 "Mary", "had", "a", "little", and "lamb."입니다.
- 해당하는 오디오 스트림에서 각 사운드에 대한 viseme는 "p", "E", "r", "i" 등입니다. viseme에 대한 자세한 내용은 Viseme 및 Amazon Polly (p. 14) 항목을 참조하십시오.

예제 2 : SSML이 있는 스피치 마크

SSML로 확장된 텍스트에서 스피치 마크를 생성하는 과정은 SSML이 없는 경우와 유사합니다. `synthesize-speech` 명령을 사용하여 다음 예제와 같이 원하는 SSML 확장 텍스트와 스피치 마크 형식을 지정하십시오. 예제를 더 쉽게 읽도록 viseme 스피치 마크는 포함하지 않았지만 포함할 수도 있습니다.

다음은 Unix, Linux, macOS용 형식으로 지정된 AWS CLI 예제입니다. Windows의 경우 각 줄 끝에 있는 백슬래시(\) Unix 연속 문자를 캐럿(^)으로 바꿉니다. 입력 텍스트는 큰 따옴표(")로 감싸고 내부 태그에는 작은 따옴표(')를 사용합니다.

```
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
```

이 요청을 하면 Amazon Polly에서는 .txt 파일에 다음을 반환합니다.

```
{"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"}
```

Amazon Polly 콘솔을 사용하여 스피치 마크 요청

콘솔을 사용하여 Amazon Polly의 스피치 마크를 요청할 수 있습니다. 그런 다음 메타데이터를 보거나 파일에 저장할 수 있습니다.

콘솔에서 스피치 마크를 생성하려면

1. AWS Management Console에 로그인한 후 https://console.aws.amazon.com/polly/에서 Amazon Polly 콘솔을 엽니다.
2. [Text-to-Speech] 탭을 선택합니다.
3. [Plain Text] 탭을 계속 사용하거나 [SSML] 탭을 선택합니다.
4. 텍스트를 입력란에 입력하거나 붙여 넣습니다.
5. [Language and region]에서 텍스트 언어를 선택합니다.
6. [Voice]에서 텍스트에 사용하려는 음성을 선택합니다.
7. 텍스트 발음을 변경하려면 [Customize Pronunciation]를 선택하고 [Apply Lexicon]에서 원하는 어휘를 선택합니다.
8. 스피치가 최종인지 확인하려면 [Listen to speech]를 선택하십시오.
9. [Change File Format]을 선택합니다.

   Note

   MP3, OGG, PCM 형식을 다운로드하면 스피치 마크가 생성되지 않습니다.

10. [File Format]에서 [Speech Marks]를 선택합니다.
11. [Speech Mark Types]에서 생성할 스피치 마크 유형을 선택합니다. [SSML] 메타데이터를 선택하는 옵션은 SSML 탭에서만 사용할 수 있습니다. Amazon Polly에서 SSML 사용에 대한 자세한 내용은 SSML 사용 (p. 62)을 참조하십시오.
12. [Change]를 선택합니다.
13. [Download Speech Marks]를 선택합니다.

지원되는 언어의 음소/Viseme 표

이 단원은 Amazon Polly이 지원하는 언어의 음소/viseme 표를 제공합니다.

항목

덴마크어(da-DK)

다음 표에는 Amazon Polly에서 지원하는 덴마크어 음성에 대해 IPA(International Phonetic Alphabet) 음소와 X-SAMPA(Extended Speech Assessment Methods Phonetic Alphabet) 기호 및 해당하는 viseme이 나와 있습니다.

음소/Viseme 표

IPA	X-SAMPA	설명	예	Viseme
자음				
b	b	유성 양순 파열음	bat	p
d	d	유성 치경 파열음	da	t
ð	D	유성 치 마찰음	mad, thriller	T
f	f	무성 순치 마찰음	fat	f
g	g	유성 연구개 파열음	gat	k
h	h	무성 성문 마찰음	hat	k
j	j	경구개 접근음	jo	i
k	k	무성 연구개 파열음	kat	k
l	l	치경 설측 접근음	ladt	t
m	m	양순 비음	mat	p
n	n	치경 비음	nay	t
ŋ	N	연구개 비음	lang	k
p	p	무성 양순 파열음	pande	p
r	r	치경 전동음	thriller, story	r
ʁ	R	유성 구개수 마찰음	rat	k
s	s	무성 치경 마찰음	sat	s
t	t	무성 치경 파열음	tal	t
v	v	유성 순치 마찰음	vat	f
w	w	양순 연구개 접근음	hav, weekend	X
모음				
ø	2	전설 원순 중고모음	øst	o
ø:	2:	전설 원순 중고장모음	øse	o
ɐ	6	중설 근저모음	mor	a
œ	9	전설 원순 중저모음	skøn, grønt	O
œ:	9:	전설 원순 중저장모음	høne, gøre	O
ə	@	중설 중모음	ane	@
æ:	{:	전설 비원순 근저장모음	male	a
a	a	전설 비원순 저모음	man	a
æ	{	전설 비원순 근저모음	adresse	a

IPA	X-SAMPA	설명	예	Viseme
ɑ	A	후설 비원순 저모음	lak, tak	a
ɑː	A:	후설 비원순 저장모음	rase	a

네덜란드어(nl-NL)

다음 표에는 Amazon Polly에서 지원하는 네덜란드어 음성에 대해 IPA(International Phonetic Alphabet) 음소와 X-SAMPA(Extended Speech Assessment Methods Phonetic Alphabet) 기호 및 해당하는 viseme이 나와 있습니다.

음소/Viseme 표

IPA	X-SAMPA	설명	예	Viseme
자음				
b	b	유성 양순 파열음	bak	p
d	d	유성 치경 파열음	dak	t
ʤ	dZ	유성 후치경 파찰음	manager	S
f	f	무성 순치 마찰음	fel	f
g	g	유성 연구개 파열음	goal	k
ɣ	G	유성 연구개 마찰음	hoed	k
ɦ	h\	유성 성문 마찰음	hand	k
j	j	경구개 접근음	ja	i
k	k	무성 연구개 파열음	kap	k
l	l	치경 설측 접근음	land	t
m	m	양순 비음	met	p
n	n	치경 비음	net	t
ŋ	N	연구개 비음	bang	k
p	p	무성 양순 파열음	pak	p
r	r	치경 전동음	rand	r
s	s	무성 치경 마찰음	sein	s
ʃ	S	무성 후치경 마찰음	show	S
t	t	무성 치경 파열음	tak	t
v	v	유성 순치 마찰음	vel	f
ʋ	v\	순치 접근음	wit	f
x	x	무성 연구개 마찰음	toch	k
z	z	유성 치경 마찰음	ziin	s

IPA	X-SAMPA	설명	예	Viseme
ʒ	Z	유성 후치경 마찰음	bagage	S
모음				
øː	2:	전설 원순 중고장모음	neus	o
œy	9y	이중 모음	buit	O
ə	@	중설 중모음	de	@
aː	a:	전설 비원순 저장모음	baad	a
ɑː	A	후설 비원순 저모음	bad	a
eː	e:	전설 비원순 중고장모음	beet	e
ɜː	3:	중설 비원순 중저장모음	barrière	E
ɛ	E	전설 비원순 중저모음	bed	E
ɛi	Ei	이중 모음	beet	E
i	i	전설 비원순 고모음	vier	i
ɪ	I	근전설 비원순 근고모음	pit	i
oː	o:	후설 원순 중고장모음	boot	o
ɔ	O	후설 원순 중저모음	pot	O
u	u	후설 원순 고모음	hoed	u
ʌu	Vu	이중 모음	fout	E
yː	y:	전설 원순 고장모음	fuut	u
ʏ	Y	근전설 원순 근고모음	hut	u

영어, 호주(en-AU)

다음 표에는 Amazon Polly에서 지원하는 호주 영어 음성에 대해 IPA(International Phonetic Alphabet) 음소와 X-SAMPA(Extended Speech Assessment Methods Phonetic Alphabet) 기호 및 해당하는 viseme이 나와 있습니다.

음소/Viseme 표

IPA	X-SAMPA	설명	예	Viseme
자음				
b	b	유성 양순 파열음	bed	p
d	d	유성 치경 파열음	dig	t
ʤ	dZ	유성 후치경 파찰음	jump	S

IPA	X-SAMPA	설명	예	Viseme
ð	D	유성 치 마찰음	then	T
f	f	무성 순치 마찰음	five	f
g	g	유성 연구개 파열음	game	k
h	h	무성 성문 마찰음	house	k
j	j	경구개 접근음	yes	i
k	k	무성 연구개 파열음	cat	k
l	l	치경 설측 접근음	lay	t
l̩	l=	성절 치경 설측 접근음	battle	t
m	m	양순 비음	mouse	p
m̩	m=	성절 양순 비음	anthem	p
n	n	치경 비음	nap	t
n̩	n=	성절 치경 비음	nap	t
ŋ	N	연구개 비음	thing	k
p	p	무성 양순 파열음	pin	p
ɹ	r\	치경 접근음	red	r
s	s	무성 치경 마찰음	seem	s
ʃ	S	무성 후치경 마찰음	ship	S
t	t	무성 치경 파열음	task	t
t͡ʃ	tS	무성 후치경 파찰음	chart	S
Θ	T	무성 치 마찰음	thin	T
v	v	유성 순치 마찰음	vest	f
w	w	양순 연구개 접근음	west	u
z	z	유성 치경 마찰음	zero	s
ʒ	Z	유성 후치경 마찰음	vision	S
모음				
ə	@	중설 중모음	arena	@
əʊ	@U	이중 모음	goat	@
æ	{	전설 비원순 근저모음	trap	a
aɪ	aI	이중 모음	price	a
aʊ	aU	이중 모음	mouth	a

IPA	X-SAMPA	설명	예	Viseme
ɑː	A:	후설 비원순 저장모음	father	a
eɪ	eI	이중 모음	face	e
ɜː	3:	중설 비원순 중저장모음	nurse	E
ɛ	E	전설 비원순 중저모음	dress	E
ɛə	E@	이중 모음	square	E
iː	i	전설 비원순 고장모음	fleece	i
ɪ	I	근전설 비원순 근고모음	kit	i
ɪə	I@	이중 모음	near	i
ɔː	OI	후설 원순 중저장모음	thought	O
ɔɪ	OI	이중 모음	choice	O
ɒ	Q	후설 원순 저모음	lot	O
uː	u:	후설 원순 고장모음	goose	u
ʊ	U	근후설 원순 근고모음	foot	u
ʊə	U@	이중 모음	cure	u
ʌ	V	후설 평순 중저모음	strut	E

영어, 인도(en-IN)

다음 표에는 Amazon Polly에서 지원하는 인도 영어 음성에 대해 IPA(International Phonetic Alphabet) 음소와 X-SAMPA(Extended Speech Assessment Methods Phonetic Alphabet) 기호 및 해당하는 viseme이 나와 있습니다.

음소/Viseme 표

IPA	X-SAMPA	설명	예	Viseme
자음				
b	b	유성 양순 파열음	bed	p
d	d	유성 치경 파열음	dig	t
ʤ	dZ	유성 후치경 파찰음	jump	S
ð	D	유성 치 마찰음	then	T
f	f	무성 순치 마찰음	five	f
g	g	유성 연구개 파열음	game	k
h	h	무성 성문 마찰음	house	k

IPA	X-SAMPA	설명	예	Viseme
j	j	경구개 접근음	yes	i
k	k	무성 연구개 파열음	cat	k
l	l	치경 설측 접근음	lay	t
ɫ	l=	성절 치경 설측 접근음	battle	t
m	m	양순 비음	mouse	p
m̩	m=	성절 양순 비음	anthem	p
n	n	치경 비음	nap	t
n̩	n=	성절 치경 비음	nap	t
ŋ	N	연구개 비음	thing	k
p	p	무성 양순 파열음	pin	p
ɹ	r\	치경 접근음	red	r
s	s	무성 치경 마찰음	seem	s
ʃ	S	무성 후치경 마찰음	ship	S
t	t	무성 치경 파열음	task	t
t͡ʃ	tS	무성 후치경 파찰음	chart	S
Θ	T	무성 치 마찰음	thin	T
v	v	유성 순치 마찰음	vest	f
w	w	양순 연구개 접근음	west	u
z	z	유성 치경 마찰음	zero	s
ʒ	Z	유성 후치경 마찰음	vision	S
모음				
ə	@	중설 중모음	arena	@
əʊ	@U	이중 모음	goat	@
æ	{	전설 비원순 근저모음	trap	a
aɪ	aI	이중 모음	price	a
aʊ	aU	이중 모음	mouth	a
ɑː	A:	후설 비원순 저장모음	father	a
eɪ	eI	이중 모음	face	e
ɜː	3:	중설 비원순 중저장모음	nurse	E
ɛ	E	전설 비원순 중저모음	dress	E

IPA	X-SAMPA	설명	예	Viseme
ɛə	E@	이중 모음	square	E
iː	i	전설 비원순 고장모음	fleece	i
ɪ	I	근전설 비원순 근고모음	kit	i
ɪə	I@	이중 모음	near	i
ɔː	OI	후설 원순 중저장모음	thought	O
ɔɪ	OI	이중 모음	choice	O
ɒ	Q	후설 원순 저모음	lot	O
uː	u:	후설 원순 고장모음	goose	u
ʊ	U	근후설 원순 근고모음	foot	u
ʊə	U@	이중 모음	cure	u
ʌ	V	후설 평순 중저모음	strut	E

영어, 영국(en-GB)

다음 표에는 Amazon Polly에서 지원하는 영국 영어 음성에 대해 IPA(International Phonetic Alphabet) 음소와 X-SAMPA(Extended Speech Assessment Methods Phonetic Alphabet) 기호 및 해당하는 viseme이 나와 있습니다.

음소/Viseme 표

IPA	X-SAMPA	설명	예	Viseme
자음				
b	b	유성 양순 파열음	bed	p
d	d	유성 치경 파열음	dig	t
ʤ	dZ	유성 후치경 파찰음	jump	S
ð	D	유성 치 마찰음	then	T
f	f	무성 순치 마찰음	five	f
g	g	유성 연구개 파열음	game	k
h	h	무성 성문 마찰음	house	k
j	j	경구개 접근음	yes	i
k	k	무성 연구개 파열음	cat	k
l	l	치경 설측 접근음	lay	t
l̩	l=	성절 치경 설측 접근음	battle	t

IPA	X-SAMPA	설명	예	Viseme
m	m	양순 비음	mouse	p
m̩	m=	성절 양순 비음	anthem	p
n	n	치경 비음	nap	t
n̩	n=	성절 치경 비음	button	t
ŋ	N	연구개 비음	thing	k
p	p	무성 양순 파열음	pin	p
ɹ	r\	치경 접근음	red	r
s	s	무성 치경 마찰음	seem	s
ʃ	S	무성 후치경 마찰음	ship	S
t	t	무성 치경 파열음	task	t
t͡ʃ	tS	무성 후치경 파찰음	chart	S
Θ	T	무성 치 마찰음	thin	T
v	v	유성 순치 마찰음	vest	f
w	w	양순 연구개 접근음	west	u
z	z	유성 치경 마찰음	zero	s
ʒ	Z	유성 후치경 마찰음	vision	S
모음				
ə	@	중설 중모음	arena	@
əʊ	@U	이중 모음	goat	@
æ	{	전설 비원순 근저모음	trap	a
aɪ	aI	이중 모음	price	a
aʊ	aU	이중 모음	mouth	a
ɑː	A:	후설 비원순 저장모음	father	a
eɪ	eI	이중 모음	face	e
ɜː	3:	중설 비원순 중저장모음	nurse	E
ɛ	E	전설 비원순 중저모음	dress	E
ɛə	E@	이중 모음	square	E
iː	i	전설 비원순 고장모음	fleece	i
ɪ	I	근전설 비원순 근고모음	kit	i
ɪə	I@	이중 모음	near	i

IPA	X-SAMPA	설명	예	Viseme
ɔː	O:	후설 원순 중저장모음	thought	O
ɔɪ	OI	이중 모음	choice	O
ɒ	Q	후설 원순 저모음	lot	O
uː	u:	후설 원순 고장모음	goose	u
ʊ	U	근후설 원순 근고모음	foot	u
ʊə	U@	이중 모음	cure	u
ʌ	V	후설 평순 중저모음	strut	E

영어, 미국(en-US)

다음 표에는 Amazon Polly에서 지원하는 미국 영어 음성에 대해 IPA(International Phonetic Alphabet) 음소와 X-SAMPA(Extended Speech Assessment Methods Phonetic Alphabet) 기호 및 해당하는 viseme이 나와 있습니다.

음소/Viseme 표

IPA	X-SAMPA	설명	예	Viseme
자음				
b	b	유성 양순 파열음	bed	p
d	d	유성 치경 파열음	dig	t
ʤ	dZ	유성 후치경 파찰음	jump	S
ð	D	유성 치 마찰음	then	T
f	f	무성 순치 마찰음	five	f
g	g	유성 연구개 파열음	game	k
h	h	무성 성문 마찰음	house	k
j	j	경구개 접근음	yes	i
k	k	무성 연구개 파열음	cat	k
l	l	치경 설측 접근음	lay	t
m	m	양순 비음	mouse	p
n	n	치경 비음	nap	t
ŋ	N	연구개 비음	thing	k
p	p	무성 양순 파열음	speak	p
ɹ	r\	치경 접근음	red	r
s	s	무성 치경 마찰음	seem	s
ʃ	S	무성 후치경 마찰음	ship	S

IPA	X-SAMPA	설명	예	Viseme
t	t	무성 치경 파열음	trap	t
t͡ʃ	tS	무성 후치경 파찰음	chart	S
θ	T	무성 치 마찰음	thin	T
v	v	유성 순치 마찰음	vest	f
w	w	양순 연구개 접근음	west	u
z	z	유성 치경 마찰음	zero	s
ʒ	Z	유성 후치경 마찰음	vision	S
모음				
ə	@	중설 중모음	arena	@
ɚ	@'	중설 치경 접근 수반음	reader	@
æ	{	전설 비원순 근저모음	trap	a
aɪ	aI	이중 모음	price	a
aʊ	aU	이중 모음	mouth	a
ɑ	A	후설 비원순 저장모음	father	a
eɪ	eI	이중 모음	face	e
ɝ	3'	중설 비원순 중저 치경 접근 수반음	nurse	E
ɛ	E	전설 비원순 중저모음	dress	E
i:	i	전설 비원순 고장모음	fleece	i
ɪ	I	근전설 비원순 근고모음	kit	i
oʊ	oU	이중 모음	goat	o
ɔ	O	후설 원순 중저장모음	thought	O
ɔɪ	OI	이중 모음	choice	O
u	u	후설 원순 고장모음	goose	u
ʊ	U	근후설 원순 근고모음	foot	u
ʌ	V	후설 평순 중저모음	strut	E

영어, 웨일스(en-GB-WSL)

다음 표에는 Amazon Polly에서 지원하는 웨일스 영어 음성에 대해 IPA(International Phonetic Alphabet) 음소와 X-SAMPA(Extended Speech Assessment Methods Phonetic Alphabet) 기호 및 해당하는 viseme이 나와 있습니다.

음소/Viseme 표

IPA	X-SAMPA	설명	예	Viseme
자음				
b	b	유성 양순 파열음	bed	p
d	d	유성 치경 파열음	dig	t
ʤ	dZ	유성 후치경 파찰음	jump	S
ð	D	유성 치 마찰음	then	T
f	f	무성 순치 마찰음	five	f
g	g	유성 연구개 파열음	game	k
h	h	무성 성문 마찰음	house	k
j	j	경구개 접근음	yes	i
k	k	무성 연구개 파열음	cat	k
l	l	치경 설측 접근음	lay	t
l̩	l=	성절 치경 설측 접근음	battle	t
m	m	양순 비음	mouse	p
m̩	m=	성절 양순 비음	anthem	p
n	n	치경 비음	nap	t
n̩	n=	성절 치경 비음	nap	t
ŋ	N	연구개 비음	thing	k
p	p	무성 양순 파열음	pin	p
ɹ	r\	치경 접근음	red	r
s	s	무성 치경 마찰음	seem	s
ʃ	S	무성 후치경 마찰음	ship	S
t	t	무성 치경 파열음	task	t
ʧ	tS	무성 후치경 파찰음	chart	S
Θ	T	무성 치 마찰음	thin	T
v	v	유성 순치 마찰음	vest	f
w	w	양순 연구개 접근음	west	u
z	z	유성 치경 마찰음	zero	s
ʒ	Z	유성 후치경 마찰음	vision	S
모음				
ə	@	중설 중모음	arena	@

IPA	X-SAMPA	설명	예	Viseme
əʊ	@U	이중 모음	goat	@
æ	{	전설 비원순 근저모음	trap	a
aɪ	aI	이중 모음	price	a
aʊ	aU	이중 모음	mouth	a
ɑː	A:	후설 비원순 저장모음	father	a
eɪ	eI	이중 모음	face	e
ɜː	3:	중설 비원순 중저장모음	nurse	E
ɛ	E	전설 비원순 중저모음	dress	E
ɛə	E@	이중 모음	square	E
iː	i	전설 비원순 고장모음	fleece	i
ɪ	I	근전설 비원순 근고모음	kit	i
ɪə	I@	이중 모음	near	i
ɔː	OI	후설 원순 중저장모음	thought	O
ɔɪ	OI	이중 모음	choice	O
ɒ	Q	후설 원순 저모음	lot	O
uː	u:	후설 원순 고장모음	goose	u
ʊ	U	근후설 원순 근고모음	foot	u
ʊə	U@	이중 모음	cure	u
ʌ	V	후설 평순 중저모음	strut	E

프랑스어(fr-FR)

다음 표에는 Amazon Polly에서 지원하는 프랑스어 음성에 대해 IPA(International Phonetic Alphabet) 음소와 X-SAMPA(Extended Speech Assessment Methods Phonetic Alphabet) 기호 및 해당하는 viseme이 나와 있습니다.

음소/Viseme 표

IPA	X-SAMPA	설명	예	Viseme
자음				
b	b	유성 양순 파열음	boire	p
d	d	유성 치경 파열음	madame	t
f	f	무성 순치 마찰음	femme	f

IPA	X-SAMPA	설명	예	Viseme
g	g	유성 연구개 파열음	grand	k
ɥ	H	양순 경구개 접근음	bruit	u
j	j	경구개 접근음	meilleur	i
k	k	무성 연구개 파열음	quatre	k
l	l	치경 설측 접근음	malade	t
m	m	양순 비음	maison	p
n	n	치경 비음	astronome	t
ɲ	J	경구개 비음	baigner	J
ŋ	N	연구개 비음	parking	k
p	p	무성 양순 파열음	pomme	p
ʁ	R	유성 구개수 마찰음	amoureux	k
s	s	무성 치경 마찰음	santé	s
ʃ	S	무성 후치경 마찰음	chat	S
t	t	무성 치경 파열음	téléphone	t
v	v	유성 순치 마찰음	vrai	f
w	w	양순 연구개 접근음	soir	u
z	z	유성 치경 마찰음	raison	s
ʒ	Z	유성 후치경 마찰음	aubergine	S
모음				
ø	2	전설 원순 중고모음	deux	o
œ	9	전설 원순 중저모음	neuf	O
œ̃	9~	전설 원순 비(nasal) 중저모음	brun	O
ə	@	중설 중모음	je	@
a	a	전설 비원순 저모음	table	a
ɑ̃	A~	후설 비원순 비(nasal) 저모음	camembert	a
e	e	전설 비원순 중고모음	marché	e
ɛ	E	전설 비원순 중저모음	neige	E
ɛ̃	E~	전설 비원순 비(nasal) 중저모음	sapin	E
i	i	전설 비원순 고모음	mille	i

IPA	X-SAMPA	설명	예	Viseme
o	o	후설 원순 중고모음	hôpital	o
ɔ	O	후설 원순 중저모음	homme	O
ɔ̃	O~	후설 원순 비(nasal) 중저모음	bon	O
u	u	후설 원순 고모음	sous	u
y	y	전설 원순 고모음	dur	u

프랑스어, 캐나다(fr-CA)

다음 표에는 Amazon Polly에서 지원하는 캐나다 프랑스어 음성에 대해 IPA(International Phonetic Alphabet) 음소와 X-SAMPA(Extended Speech Assessment Methods Phonetic Alphabet) 기호 및 해당하는 viseme이 나와 있습니다.

음소/Viseme 표

IPA	X-SAMPA	설명	예	Viseme
자음				
b	b	유성 양순 파열음	boire	p
d	d	유성 치경 파열음	madame	t
f	f	무성 순치 마찰음	femme	f
g	g	유성 연구개 파열음	grand	k
ɥ	H	양순 경구개 접근음	bruit	u
j	j	경구개 접근음	meilleur	i
k	k	무성 연구개 파열음	quatre	k
l	l	치경 설측 접근음	malade	t
m	m	양순 비음	maison	p
n	n	치경 비음	astronome	t
ɲ	J	경구개 비음	baigner	J
ŋ	N	연구개 비음	parking	k
p	p	무성 양순 파열음	pomme	p
ʁ	R	유성 구개수 마찰음	amoureux	k
s	s	무성 치경 마찰음	santé	s
ʃ	S	무성 후치경 마찰음	chat	S
t	t	무성 치경 파열음	téléphone	t
v	v	유성 순치 마찰음	vrai	f

IPA	X-SAMPA	설명	예	Viseme
w	w	양순 연구개 접근음	soir	u
z	z	유성 치경 마찰음	raison	s
ʒ	Z	유성 후치경 마찰음	aubergine	S
모음				
ø	2	전설 원순 중고모음	deux	o
œ	9	전설 원순 중저모음	neuf	O
œ̃	9~	전설 원순 비(nasal) 중저모음	brun	O
ə	@	중설 중모음	je	@
a	a	전설 비원순 저모음	table	a
ɑ̃	A~	후설 비원순 비(nasal) 저모음	camembert	a
e	e	전설 비원순 중고모음	marché	e
ɛ	E	전설 비원순 중저모음	neige	E
ɛ̃	E~	전설 비원순 비(nasal) 중저모음	sapin	E
i	i	전설 비원순 고모음	mille	i
o	o	후설 원순 중고모음	hôpital	o
ɔ	O	후설 원순 중저모음	homme	O
ɔ̃	O~	후설 원순 비(nasal) 중저모음	bon	O
u	u	후설 원순 고모음	sous	u
y	y	전설 원순 고모음	dur	u

독일어(de-DE)

다음 표에는 Amazon Polly에서 지원하는 독일어 음성에 대해 IPA(International Phonetic Alphabet) 음소와 X-SAMPA(Extended Speech Assessment Methods Phonetic Alphabet) 기호 및 해당하는 viseme이 나와 있습니다.

음소/Viseme 표

IPA	X-SAMPA	설명	예	Viseme
자음				
ʔ	?	성문 폐쇄음		
b	b	유성 양순 파열음	Bier	p

IPA	X-SAMPA	설명	예	Viseme
d	d	유성 치경 파열음	Dach	t
ç	C	무성 경구개 마찰음	ich	k
d͡ʒ	dZ	유성 후치경 파찰음	Dschungel	S
f	f	무성 순치 마찰음	Vogel	f
g	g	유성 연구개 파열음	Gabel	k
h	h	무성 성문 마찰음	Haus	k
j	j	무성 성문 마찰음	jemand	i
k	k	무성 연구개 파열음	Kleid	k
l	l	치경 설측 접근음	Loch	t
m	m	양순 비음	Milch	p
n	n	치경 비음	Natur	t
ŋ	N	연구개 비음	klingen	k
p	p	무성 양순 파열음	Park	p
p͡f	pf	무성 순치 파찰음	Apfel	
ʁ	R	구개수 전동음	Regen	
s	s	무성 치경 마찰음	Messer	s
ʃ	S	무성 후치경 마찰음	Fischer	S
t	t	무성 치경 파열음	Topf	T
t͡s	Ts	무성 치경 파찰음	Zahl	
t͡ʃ	tS	무성 후치경 파찰음	deutsch	S
v	v	유성 순치 마찰음	Wasser	f
x	x	무성 연구개 마찰음	kochen	k
z	z	유성 치경 마찰음	See	s
ʒ	Z	유성 후치경 마찰음	Orange	S
모음				
øː	2:	전설 원순 중고장모음	böse	o
ɐ	6	중설 근저모음	besser	a
ɐ̯	6_^	비성절 중설 근저모음	Klar	a
œ	9	전설 원순 중저모음	können	O
ə	@	중설 중모음	Rede	@
a	a	전설 비원순 저모음	Salz	a

IPA	X-SAMPA	설명	예	Viseme
a:	a:	전설 비원순 저장모음	Sahne	a
aɪ	aI	이중 모음	nein	a
aʊ	aU	이중 모음	Augen	a
ɑ̃	A~	후설 비원순 비(nasal) 저모음	Restaurant	a
e:	e:	전설 비원순 중고장모음	Rede	e
ɛ	E	전설 비원순 중저모음	Keller	E
ɛ̃	E~	전설 비원순 비(nasal) 중저모음	Terrain	E
i:	i:	전설 비원순 고장모음	Lied	i
ɪ	I	근전설 비원순 근고모음	bitte	i
o:	o:	후설 원순 중고장모음	Kohl	o
ɔ	O	후설 원순 중저모음	Koffer	O
ɔ̃	O~	후설 원순 비(nasal) 중저모음	Annonce	O
ɔʏ	OY	이중 모음	neu	O
u:	u:	후설 원순 고장모음	Bruder	u
ʊ	U	근후설 원순 근고모음	Wunder	u
y:	y:	전설 원순 고장모음	kühl	u
ʏ	Y	근전설 원순 근고모음	Küche	u

아이슬란드어(is-IS)

다음 표에는 Amazon Polly에서 지원하는 아이슬란드어 음성에 대해 IPA(International Phonetic Alphabet) 음소와 X-SAMPA(Extended Speech Assessment Methods Phonetic Alphabet) 기호 및 해당하는 viseme이 나와 있습니다.

음소/Viseme 표

IPA	X-SAMPA	설명	예	Viseme
자음				
b	b	유성 양순 파열음	grasbakkanum	0
c	c	무성 경구개 파열음	pakkin	k
cʰ	c_h	기식 무성 경구개 파열음	anarkistai	k

IPA	X-SAMPA	설명	예	Viseme
ç	C	무성 경구개 마찰음	héðan	k
d	d	유성 치경 파열음	bóndi	t
ð	D	유성 치 마찰음	borð	T
f	f	무성 순치 마찰음	duft	f
g	g	유성 연구개 파열음	holgóma	k
ɣ	G	유성 연구개 마찰음	hugur	k
h	h	무성 성문 마찰음	heili	k
j	j	경구개 접근음	jökull	i
kʰ	k_h	기식 무성 연구개 파열음	ósköpunum	k
l	l	치경 설측 접근음	gólf	t
l̥	l_0	무성 치경 설측 접근음	fólk	t
m	m	양순 비음	september	p
m̥	m_0	무성 양순 비음	kompa	p
n	n	치경 비음	númer	t
n̥	n_0	무성 치경 비음	pöntun	t
ɲ	J	경구개 비음	pælingar	J
ŋ	N	연구개 비음	söngvarann	k
ŋ̊	N_0	무성 연구개 비음	frænka	k
pʰ	p_h	기식 무성 양순 파열음	afplánun	p
r	r	치경 전동음	afskrifta	r
r̥	r_0	무성 치경 전동음	andvörpum	r
s	s	무성 치경 마찰음	baðhús	s
tʰ	t_h	기식 무성 치경 파열음	tanki	t
θ	T	무성 치 마찰음	þeldökki	T
v	v	유성 순치 마찰음	silfur	f
w	w	양순 연구개 접근음		u
x	x	무성 연구개 마찰음	samfélags	k
모음				
œ	9	전설 원순 중저모음	þröskuldinum	O

IPA	X-SAMPA	설명	예	Viseme
œ:	9:	전설 원순 중저장모음	tvö	O
a	a	전설 비원순 저모음	nefna	a
a:	a:	전설 비원순 저장모음	fara	a
au	au	이중 모음	átta	a
au:	au:	이중 모음	átján	a
ɛ	E	전설 비원순 중저모음	kennari	E
ɛ:	E:	전설 비원순 중저장모음	dreka	E
i	i	전설 비원순 고모음	Gúlíver	i
i:	i:	전설 비원순 고장모음	þrír	i
ɪ	I	근전설 비원순 근고모음	samspil	i
ɪ:	I:	근전설 비원순 근고장모음	stig	i
ɔ	O	후설 원순 중저모음	regndropar	O
ɔ:	O:	후설 원순 중저장모음	ullarbolur	O
ɔu	Ou	이중 모음	tólf	O
ɔu:	Ou:	이중 모음	fjórir	O
u	u	후설 원순 고모음	stúlkan	u
u:	u:	후설 원순 고장모음	frú	u
ʏ	Y	근전설 원순 근고모음	tíu	u
ʏ:	Y	근전설 원순 근고장모음	gruninn	u

이탈리아어(it-IT)

다음 표에는 Amazon Polly에서 지원하는 이탈리아어 음성에 대해 IPA(International Phonetic Alphabet) 음소와 X-SAMPA(Extended Speech Assessment Methods Phonetic Alphabet) 기호 및 해당하는 viseme이 나와 있습니다.

음소/Viseme 표

IPA	X-SAMPA	설명	예	Viseme
자음				
b	b	유성 양순 파열음	bacca	p
d	d	유성 치경 파열음	dama	t

IPA	X-SAMPA	설명	예	Viseme
ʣ	dz	유성 치경 파찰음	zero	s
ʤ	dZ	유성 후치경 파찰음	giro	S
f	f	무성 순치 마찰음	famiglia	f
g	g	유성 연구개 파열음	gatto	k
h	h	무성 성문 마찰음	horror	k
j	j	경구개 접근음	dieci	i
k	k	무성 연구개 파열음	campo	k
l	l	치경 설측 접근음	lido	t
ʎ	L	경구개 설측 접근음	aglio	J
m	m	양순 비음	mille	p
n	n	치경 비음	nove	t
ɲ	J	경구개 비음	lasagne	J
p	p	무성 양순 파열음	pizza	p
r	r	치경 전동음	risata	r
s	s	무성 치경 마찰음	sei	s
ʃ	S	무성 후치경 마찰음	scienza	S
t	t	무성 치경 파열음	tavola	t
ʦ	ts	무성 치경 파찰음	forza	s
ʧ	tS	무성 후치경 파찰음	cielo	S
v	v	유성 순치 마찰음	venti	f
w	w	양순 연구개 접근음	quattro	u
z	z	유성 치경 마찰음	bisogno	s
ʒ	Z	유성 후치경 마찰음	bijou	S
모음				
a	a	전설 비원순 저모음	arco	a
e	e	전설 비원순 중고모음	tre	e
ɛ	E	전설 비원순 중저모음	ettaro	E
i	i	전설 비원순 고모음	impero	i
o	o	후설 원순 중고모음	cento	o
ɔ	O	후설 원순 중저모음	otto	O
u	u	후설 원순 고모음	uno	u

일본어(ja-JP)

다음 표에는 Amazon Polly에서 지원하는 일본어 음성에 대해 IPA(International Phonetic Alphabet) 음소와 X-SAMPA(Extended Speech Assessment Methods Phonetic Alphabet) 기호 및 해당하는 viseme이 나와 있습니다.

IPA	X-SAMPA	설명	예	Viseme
자음				
ɾ	4	치경 탄음	練習, renshuu	t
ʔ	?	성문 폐쇄음	あつっ, atsu'	
b	b	유성 양순 파열음	舞踊, buyou	p
β	B	유성 양순 마찰음	ヴィンテージ, vinteeji	B
c	c	무성 경구개 파열음	ききょう, kikyou	k
ç	C	무성 경구개 마찰음	人, hito	k
d	d	유성 치경 파열음	濁点, dakuten	t
ʥ	dz\	유성 치경구개 파찰음	純, jun	J
g	g	유성 연구개 파열음	ご飯, gohan	k
h	h	무성 성문 마찰음	本, hon	k
j	j	경구개 접근음	屋根, yane	i
ɟ	J\	유성 경구개 파열음	行儀, gyougi	J
k	k	무성 연구개 파열음	漢字, kanji	k
ɺ	l\	치경 설측 탄음	釣り, tsuri	r
ɺj	l\j	치경 설측 탄음, 경구개 접근음	流行, ryuukou	r
m	m	양순 비음	飯, meshi	p
n	n	치경 비음	猫, neko	t
ɲ	J	경구개 비음	日本, nippon	J
ɴ	N\	구개수 비음	缶, kan	k
p	p	무성 양순 파열음	パン, pan	p
ɸ	p\	무성 양순 마찰음	福, huku	f
s	s	무성 치경 마찰음	層, sou	s
ɕ	s\	무성 치경구개 마찰음	書簡, shokan	J
t	t	무성 치경 파열음	手紙, tegami	t
ʦ	ts	무성 치경 파찰음	釣り, tsuri	s

IPA	X-SAMPA	설명	예	Viseme
t͡ɕ	ts\	무성 치경구개 파찰음	吉, kichi	J
w	w	양순 연구개 접근음	電話, denwa	u
z	z	유성 치경 마찰음	座敷, zashiki	s
모음				
äː	a:_"	중설 비원순 저장모음	羽蟻, haari	a
ä	a_"	중설 비원순 저모음	仮名, kana	a
eː	e:_o	전설 비원순 중장모음	学生, gakusei	@
e	e_o	전설 비원순 중모음	歴, reki	@
i	i	전설 비원순 고모음	気, ki	i
iː	i:	전설 비원순 고장모음	詩歌, shiika	i
ɯ	M USD	후설 비원순 고모음	運, un	i
ɯː	M USD:	후설 비원순 고장모음	宗教, shuukyou	i
oː	o:_o	후설 원순 중장모음	購読, koodoku	o
o	o_o	후설 원순 중모음	読者, dokusha	o

한국어(ko-KR)

다음 표에는 Amazon Polly에서 지원하는 한국어 음성에 대해 IPA(International Phonetic Alphabet) 음소와 X-SAMPA(Extended Speech Assessment Methods Phonetic Alphabet) 기호 및 해당하는 viseme이 나와 있습니다.

IPA	X-SAMPA	설명	예	Viseme
자음				
k	k	무성 연구개 파열음	강, [g]ang	k
k#	k_t	무성 연구개 경파열음	깨, [kk]e	k
n	n	치경 비음	남, [n]am	t
t	t	무성 치경 파열음	도, [d]o	t
t#	t_t	무성 치경 경파열음	때, [tt]e	t
ɾ	4	치경 탄음	사랑, sa[r]ang	t
l	l	치경 설측 접근음	돌, do[l]	t
m	m	양순 비음	무, [m]u	p
p	p	무성 양순 파열음	봄, [b]om	p
p#	p_t	무성 양순 경파열음	뻘, [pp]eol	p

IPA	X-SAMPA	설명	예	Viseme
s	s	무성 치경 마찰음	새, [s]e	s
s#	s_t	무성 치경 경마찰음	씨, [ss]i	s
ŋ	N	연구개 비음	방, ba[ng]	k
t͡ɕ	ts\	무성 치경구개 파찰음	조, [j]o	J
t͡#ɕ	ts_t	무성 치경구개 경파찰음	찌, [jj]i	J
t͡ɕʰ	ts_h	기식 무성 치경구개 경파찰음	차, [ch]a	J
kʰ	k_h	기식 무성 연구개 파열음	코, [k]o	k
tʰ	t_h	기식 무성 치경 파열음	통, [t]ong	t
pʰ	p_h	기식 무성 양순 파열음	패, [p]e	p
h	h	무성 성문 마찰음	힘, [h]im	k
j	j	경구개 접근음	양, [y]ang	i
w	w	양순 연구개 접근음	왕, [w]ang	u
ɯ̟	M\	연구개 접근음>	의, [wj]i	i
모음				
a	a	전설 비원순 저모음	밥, b[a]b	a
ʌ	V	후설 평순 중저모음	정, j[eo]ng	E
ɛ	E	전설 비원순 중저모음	배, b[e]	E
o	o	후설 원순 중고모음	노, n[o]	o
u	u	후설 원순 고모음	둘, d[u]l	u
ɯ	M	후설 비원순 고모음	은, [eu]n	i
i	i	전설 비원순 고모음	김, k[i]m	i

노르웨이어(nb-NO)

다음 차트에는 노르웨이어 음성에 대해 Amazon Polly에서 지원하는 IPA(International Phonetic Alphabet) 음소와 X-SAMPA(Extended Speech Assessment Methods Phonetic Alphabet) 기호 세트 및 해당하는 viseme이 나와 있습니다.

IPA	X-SAMPA	설명	예	Viseme
자음				

IPA	X-SAMPA	설명	예	Viseme
ɾ	4	치경 탄음	prøv	t
b	b	유성 양순 파열음	labb	p
ç	C	무성 경구개 마찰음	kino	k
d	d	유성 치경 파열음	ladd	t
ɖ	d`	유성 권설 파열음	verdi	t
f	f	무성 순치 마찰음	fot	f
g	g	유성 연구개 파열음	tagg	k
h	h	무성 성문 마찰음	ha	k
j	j	경구개 접근음	gi	i
k	k	무성 연구개 파열음	takk	k
l	l	치경 설측 접근음	fall, ball	t
ɭ	l`	권설 설측 접근음	ærlig	t
m	m	양순 비음	lam	p
n	n	치경 비음	vann	t
ɳ	n`	권설 비음	garn	t
ŋ	N	연구개 비음	sang	k
p	p	무성 양순 파열음	hopp	p
s	s	무성 치경 마찰음	lass	s
ʂ	s`	무성 권설 마찰음	års	S
ʃ	S	무성 후치경 마찰음	skyt	S
t	t	무성 치경 파열음	lat	t
ʈ	t`	무성 권설 파열음	hardt	t
ʊ	v\	순치 접근음	vin	f
w	w	양순 연구개 접근음	will	x
모음				
øː	2:	전설 원순 중고장모음	søt	o
œ	9	전설 원순 중저모음	søtt	O
ə	@	중설 중모음	ape	@
æː	{:	전설 비원순 근저장모음	vær	a
ʉ	}	중설 원순 고모음	lund	u

IPA	X-SAMPA	설명	예	Viseme
ʉː	}:	중설 원순 고장모음	lun	u
æ	{	전설 비원순 근저모음	vært	a
ɑ	A	후설 비원순 저모음	hatt	a
ɑː	A:	후설 비원순 저장모음	hat	a
eː	e:	전설 비원순 중고장모음	sen	e
ɛ	E	전설 비원순 중저모음	send	E
iː	i:	전설 비원순 고장모음	vin	i
ɪ	I	근전설 비원순 근고모음	vind	i
oː	o:	후설 원순 중고장모음	våt	o
ɔ	O	후설 원순 중저모음	vått	O
uː	u:	후설 원순 고장모음	bok	u
ʊ	U	근후설 원순 근고모음	bukk	u
yː	y:	전설 원순 고장모음	lyn	u
ʏ	Y	근전설 원순 근고모음	lynne	u

폴란드어(pl-PL)

다음 표에는 Amazon Polly에서 지원하는 폴란드어 음성에 대해 IPA(International Phonetic Alphabet) 음소와 X-SAMPA(Extended Speech Assessment Methods Phonetic Alphabet) 기호 및 해당하는 viseme이 나와 있습니다.

음소/Viseme 표

IPA	X-SAMPA	설명	예	Viseme
자음				
b	b	유성 양순 파열음	bobas, belka	p
d	d	유성 치경 파열음	dar, do	t
d͡z	dz	유성 치경 파찰음	dzwon, widzowie	s
d͡ʑ	dz\	유성 치경구개 파찰음	dźwięk	J
d͡ʐ	dz`	유성 권설 파찰음	dżem, dżungla	S
f	f	무성 순치 마찰음	furtka, film	f
g	g	유성 연구개 파열음	gazeta, waga	k
h	h	무성 성문 마찰음	chleb, handel	k

IPA	X-SAMPA	설명	예	Viseme
j	j	경구개 접근음	jak, maja	i
k	k	무성 연구개 파열음	kura, marek	k
l	l	치경 설측 접근음	lipa, alicja	t
m	m	양순 비음	matka, molo	p
n	n	치경 비음	norka	t
ɲ	J	경구개 비음	koń, toruń	J
p	p	무성 양순 파열음	pora, stop	p
r	r	치경 전동음	rok, park	r
s	s	무성 치경 마찰음	sum, pas	s
ɕ	s\	무성 치경구개 마찰음	śruba, śnieg	J
ʂ	s`	무성 권설 마찰음	szum, masz	S
t	t	무성 치경 파열음	tok, stół	t
t͡s	ts	무성 치경 파찰음	car, co	s
t͡ɕ	ts\	무성 치경구개 파찰음	ćma, mieć	J
t͡ʂ	ts`	무성 권설 파찰음	czas, raczej	S
v	v	유성 순치 마찰음	worek, mewa	f
w	w	양순 연구개 접근음	łaska, mało	u
z	z	유성 치경 마찰음	zero	s
ʑ	z\	유성 치경구개 마찰음	źrebię, bieliźnie	J
ʐ	z`	유성 권설 마찰음	żar, żona	S
모음				
a	a	전설 비원순 저모음	ja	a
ɛ	E	전설 비원순 중저모음	echo	E
ɛ̃	E~	전설 비원순 비(nasal) 중저모음	węże	E
i	i	전설 비원순 고모음	ile	i
ɔ	O	후설 원순 중저모음	oczy	O
ɔ̃	O~	후설 원순 비(nasal) 중저모음	wąż	O
u	u	후설 원순 고모음	uczta	u
ɨ	1	중설 비원순 고모음	byk	i

포르투갈어(pt-PT)

다음 표에는 Amazon Polly에서 지원하는 포르투갈어 음성에 대해 IPA(International Phonetic Alphabet) 음소와 X-SAMPA(Extended Speech Assessment Methods Phonetic Alphabet) 기호 및 해당하는 viseme이 나와 있습니다.

음소/Viseme 표

IPA	X-SAMPA	설명	예	Viseme
자음				
ɾ	4	치경 탄음	pira	t
b	b	유성 양순 파열음	dato	p
d	d	유성 치경 파열음	dato	t
f	f	무성 순치 마찰음	facto	f
g	g	유성 연구개 파열음	gato	k
j	j	경구개 접근음	paraguay	i
k	k	무성 연구개 파열음	cacto	k
l	l	치경 설측 접근음	galo	t
ʎ	L	경구개 설측 접근음	galho	J
m	m	양순 비음	mato	p
n	n	치경 비음	nato	t
ɲ	J	경구개 비음	pinha	J
p	p	무성 양순 파열음	pato	p
ʁ	R\	구개수 전동음	barroso	k
s	s	무성 치경 마찰음	saca	s
ʃ	S	무성 후치경 마찰음	chato	S
t	t	무성 치경 파열음	tacto	t
v	v	유성 순치 마찰음	vaca	f
w	w	양순 연구개 접근음	mau	u
z	z	유성 치경 마찰음	zaca	s
ʒ	Z	유성 후치경 마찰음	jacto	S
모음				
a	a	전설 비원순 저모음	parto	a
ã	a~	전설 비원순 비(nasal) 저모음	pega	a
e	e	전설 비원순 중고모음	pega	e

IPA	X-SAMPA	설명	예	Viseme
ẽ	e~	전설 비원순 비(nasal) 중고모음	movem	e
ɛ	E	전설 비원순 중저모음	café	E
i	i	전설 비원순 고모음	lingueta	i
ĩ	i~	전설 비원순 비(nasal) 고모음	cinto	i
o	o	후설 원순 중고모음	poder	o
õ	o~	후설 원순 비(nasal) 중고모음	compra	o
ɔ	O	후설 원순 중저모음	cotó	O
u	u	후설 원순 고모음	fui	u
ũ	u~	후설 원순 비(nasal) 고모음	sunto	u

포르투갈어, 브라질(pt-BR)

다음 표에는 Amazon Polly에서 지원하는 브라질어 음성에 대해 IPA(International Phonetic Alphabet) 음소와 X-SAMPA(Extended Speech Assessment Methods Phonetic Alphabet) 기호 및 해당하는 viseme이 나와 있습니다.

음소/Viseme 표

IPA	X-SAMPA	설명	예	Viseme
자음				
ɾ	4	치경 탄음	pira	t
b	b	유성 양순 파열음	bato	p
d	d	유성 치경 파열음	dato	t
͡dʒ	dZ	유성 후치경 파찰음	idade	S
f	f	무성 순치 마찰음	facto	f
g	g	유성 연구개 파열음	gato	k
j	j	경구개 접근음	paraguay	i
k	k	무성 연구개 파열음	cacto	k
l	l	치경 설측 접근음	galo	t
ʎ	L	경구개 설측 접근음	galho	J
m	m	양순 비음	mato	p
n	n	치경 비음	nato	t

IPA	X-SAMPA	설명	예	Viseme
ɲ	J	경구개 비음	pinha	J
p	p	무성 양순 파열음	pato	p
s	s	무성 치경 마찰음	saca	s
ʃ	S	무성 후치경 마찰음	chato	S
t	t	무성 치경 파열음	tacto	t
t͡ʃ	tS	무성 후치경 파찰음	noite	S
v	v	유성 순치 마찰음	vaca	f
w	w	양순 연구개 접근음	mau	u
χ	X	무성 구개수 마찰음	carro	k
z	z	유성 치경 마찰음	zaca	s
ʒ	Z	유성 후치경 마찰음	jacto	S
모음				
a	a	전설 비원순 저모음	parto	a
ã	a~	전설 비원순 비(nasal) 저모음	pensamos	a
e	e	전설 비원순 중고모음	pega	e
ẽ	e~	전설 비원순 비(nasal) 중고모음	movem	e
ɛ	E	전설 비원순 중저모음	café	E
i	i	전설 비원순 고모음	lingueta	i
ĩ	i~	전설 비원순 비(nasal) 고모음	cinto	i
o	o	후설 원순 중고모음	poder	o
õ	o~	후설 원순 비(nasal) 중고모음	compra	o
ɔ	O	후설 원순 중저모음	cotó	O
u	u	후설 원순 고모음	fui	u
ũ	u~	후설 원순 비(nasal) 고모음	sunto	u

루마니아어(ro-RO)

다음 표에는 Amazon Polly에서 지원하는 루마니아어 음성에 대해 IPA(International Phonetic Alphabet) 음소와 X-SAMPA(Extended Speech Assessment Methods Phonetic Alphabet) 기호 및 해당하는 viseme이 나와 있습니다.

음소/Viseme 표

IPA	X-SAMPA	설명	예	Viseme
자음				
b	b	유성 양순 파열음	bubă	p
d	d	유성 치경 파열음	după	t
d͡ʒ	dZ	유성 후치경 파찰음	george	S
f	f	무성 순치 마찰음	afacere	f
g	g	유성 연구개 파열음	agri#	k
h	h	무성 성문 마찰음	harpă	k
j	j	경구개 접근음	baie	i
k	k	무성 연구개 파열음	co#	k
l	l	치경 설측 접근음	lampa	t
m	m	양순 비음	mama	p
n	n	치경 비음	nor	t
p	p	무성 양순 파열음	pilă	p
r	r	치경 전동음	rampă	r
s	s	무성 치경 마찰음	soare	s
ʃ	S	무성 후치경 마찰음	ma#ină	S
t	t	무성 치경 파열음	tata	t
t͡s	ts	무성 치경 파찰음	#ară	s
t͡ʃ	tS	무성 후치경 파찰음	ceai	S
v	v	유성 순치 마찰음	via#ă	f
w	w	양순 연구개 접근음	beau	u
z	z	유성 치경 마찰음	mozol	s
ʒ	Z	유성 후치경 마찰음	joacă	S
모음				
ə	@	중설 중모음	babă	@
a	a	전설 비원순 저모음	casa	a
e	e	전설 비원순 중고모음	elan	e
e̯	e_^	비성절 전설 비원순 중고모음	beau	e
i	i	전설 비원순 고모음	mie	i
o	o	후설 원순 중고모음	oră	o

IPA	X-SAMPA	설명	예	Viseme
oa	o_^a	이중 모음	oare	o
u	u	후설 원순 고모음	unde	u
ɨ	1	중설 비원순 고모음	România	i

러시아어(ru-RU)

다음 표에는 Amazon Polly에서 지원하는 러시아어 음성에 대해 IPA(International Phonetic Alphabet) 음소와 X-SAMPA(Extended Speech Assessment Methods Phonetic Alphabet) 기호 및 해당하는 viseme이 나와 있습니다.

음소/Viseme 표

IPA	X-SAMPA	설명	예	Viseme
자음				
b	b	유성 양순 파열음	борт	p
bʲ	b'	경구개음화 유성 양순 파열음	бюро	p
d	d	유성 치경 파열음	дом	t
dʲ	d'	경구개음화 유성 치경 파열음	дядя	t
f	f	무성 순치 마찰음	флаг	f
fʲ	f'	경구개음화 무성 순치 마찰음	февраль	f
g	g	유성 연구개 파열음	нога	k
gʲ	g'	경구개음화 유성 연구개 파열음	герой	k
j	j	경구개 접근음	дизайн, ящик	i
k	k	무성 연구개 파열음	кот	k
kʲ	k'	경구개음화 무성 연구개 파열음	кино	k
l	l	치경 설측 접근음	лампа	t
lʲ	l'	경구개음화 치경 설측 접근음	лес	t
m	m	양순 비음	мама	p
mʲ	m'	경구개음화 양순 비음	мяч	p
n	n	치경 비음	нос	t
nʲ	n'	경구개음화 치경 비음	няня	t

IPA	X-SAMPA	설명	예	Viseme
p	p	무성 양순 파열음	папа	p
pʲ	p'	경구개음화 무성 양순 파열음	перо	p
r	r	치경 전동음	роза	r
rʲ	r'	경구개음화 치경 전동음	рюмка	r
s	s	무성 치경 마찰음	сыр	s
sʲ	s'	경구개음화 무성 치경 마찰음	сердце, русь	s
ɕː	s\:	무성 치경구개 마찰장음	щека	J
ʂ	s`	무성 권설 마찰음	шум	S
t	t	무성 치경 파열음	точка	t
tʲ	t'	경구개음화 무성 치경 파열음	тётя	t
t͡s	ts	무성 치경 파찰음	царь	s
t͡ɕ	ts\	무성 치경구개 파찰음	час	J
v	v	유성 순치 마찰음	вор	f
vʲ	v'	경구개음화 유성 순치 마찰음	верфь	f
x	x	무성 연구개 마찰음	хор	k
xʲ	x'	경구개음화 무성 연구개 마찰음	химия	k
z	z	유성 치경 마찰음	зуб	s
zʲ	z'	경구개음화 유성 치경 마찰음	зима	s
ʑː	z\:	유성 치경구개 마찰장음	уезжать	J
ʐ	z`	유성 권설 마찰음	жена	S
모음				
ə	@	중설 중모음	канарейка	@
a	a	전설 비원순 저모음	два, яблоко	a
e	e	전설 비원순 중고모음	печь	e
ɛ	E	전설 비원순 중저모음	это	E
i	i	전설 비원순 고모음	один, четыре	i

IPA	X-SAMPA	설명	예	Viseme
o	o	후설 원순 중고모음	кот	o
u	u	후설 원순 고모음	муж, вьюга	u
ɨ	1	중설 비원순 고모음	мышь	i

스페인어(es-ES)

다음 표에는 Amazon Polly에서 지원하는 스페인어 음성에 대해 IPA(International Phonetic Alphabet) 음소와 X-SAMPA(Extended Speech Assessment Methods Phonetic Alphabet) 기호 및 해당하는 viseme을 보여줍니다.

음소/Viseme 표

IPA	X-SAMPA	설명	예	Viseme
자음				
ɾ	4	치경 탄음	pero, bravo, amor, eterno	t
b	b	유성 양순 파열음	bestia	p
β	B	유성 양순 마찰음	bebé	B
d	d	유성 치경 파열음	cuando	t
ð	D	유성 치 마찰음	arder	T
f	f	무성 순치 마찰음	fase, café	f
g	g	유성 연구개 파열음	gato, lengua, guerra	k
ɣ	G	유성 연구개 마찰음	trigo, Argos	k
j	j	경구개 접근음	hacia, tierra, radio, viuda	i
ʝ	j\	유성 경구개 마찰음	enhielar, sayo, inyectado, desyerba	J
k	k	무성 연구개 파열음	caña, laca, quisimos	k
l	l	치경 설측 접근음	lino, calor, principal	t
ʎ	L	경구개 설측 접근음	llave, pollo	J
m	m	양순 비음	madre, comer, anfibio	p
n	n	치경 비음	nido, anillo, sin	t
ɲ	J	경구개 비음	cabaña, ñoquis	J
ŋ	N	연구개 비음	cinco, venga	k
p	p	무성 양순 파열음	pozo, topo	p

IPA	X-SAMPA	설명	예	Viseme
r	r	치경 전동음	perro, enrachado	r
s	s	무성 치경 마찰음	saco, casa, puertas	s
t	t	무성 치경 파열음	tamiz, átomo	t
t͡ʃ	tS	무성 후치경 파찰음	chubasco	S
θ	T	무성 치 마찰음	cereza, zorro, lacero, paz	T
w	w	양순 연구개 접근음	fuego, fuimos, cuota, cuadro	u
x	x	무성 연구개 마찰음	jamón, general, suje, reloj	k
z	z	유성 치경 마찰음	rasgo, mismo	s
모음				
a	a	전설 비원순 저모음	tanque	a
e	e	전설 비원순 중고모음	peso	e
i	i	전설 비원순 고모음	cinco	i
o	o	후설 원순 중고모음	bosque	o
u	u	전설 비원순 중고모음	publicar	u
e	e	전설 비원순 중고모음	keçi	e
ɛ	E	전설 비원순 중저모음	dede	e
i	i	전설 비원순 고모음	bir	i
i:	i:	전설 비원순 고장모음	izah	i
ɪ	I	근전설 비원순 근고모음	keçi	i
ɯ	M USD	후설 비원순 고모음	kıl	i
o	o	후설 원순 중고장모음	kol	o
o:	o:	후설 원순 중고장모음	dolar	o
u	u	후설 원순 고모음	durum	u
u:	u:	후설 원순 고장모음	ruhum	u
ʊ	U	근후설 원순 근고모음	dolu	u
Y	y	전설 원순 고모음	güvenlik	u
ʏ	Y	근전설 원순 근고모음	aşı	u

스페인어, 미국(es-US)

다음 표에는 Amazon Polly에서 지원하는 미국 스페인어 음성에 대해 IPA(International Phonetic Alphabet) 음소와 X-SAMPA(Extended Speech Assessment Methods Phonetic Alphabet) 기호 및 해당하는 viseme이 나와 있습니다.

IPA	X-SAMPA	설명	예	Viseme
자음				
b	b	유성 양순 파열음	bed	p
d	d	유성 치경 파열음	dig	t
ʤ	dZ	유성 후치경 파찰음	jump	S
ð	D	유성 치 마찰음	then	T
f	f	무성 순치 마찰음	five	f
g	g	유성 연구개 파열음	game	k
h	h	무성 성문 마찰음	house	k
j	j	경구개 접근음	yes	i
k	k	무성 연구개 파열음	cat	k
l	l	치경 설측 접근음	lay	t
m	m	양순 비음	mouse	p
n	n	치경 비음	nap	t
ŋ	N	연구개 비음	thing	k
p	p	무성 양순 파열음	speak	p
ɹ	r\	치경 접근음	red	r
s	s	무성 치경 마찰음	seem	s
ʃ	S	무성 후치경 마찰음	ship	S
t	t	무성 치경 파열음	trap	t
ʧ	tS	무성 후치경 파찰음	chart	S
Θ	T	무성 치 마찰음	thin	T
v	v	유성 순치 마찰음	vest	f
w	w	양순 연구개 접근음	west	u
z	z	유성 치경 마찰음	zero	s
ʒ	Z	유성 후치경 마찰음	vision	S
모음				
ə	@	중설 중모음	arena	@

IPA	X-SAMPA	설명	예	Viseme
ɚ	@'	중설 치경 접근 수반음	reader	@
æ	{	전설 비원순 근저모음	trap	a
aɪ	aI	이중 모음	price	a
aʊ	aU	이중 모음	mouth	a
ɑ	A	후설 비원순 저장모음	father	a
eɪ	eI	이중 모음	face	e
ɝ	3'	중설 비원순 중저 치경 접근 수반음	nurse	E
ɛ	E	전설 비원순 중저모음	dress	E
iː	i	전설 비원순 고장모음	fleece	i
ɪ	I	근전설 비원순 근고모음	kit	i
oʊ	oU	이중 모음	goat	o
ɔ	O	후설 원순 중저장모음	thought	O
ɔɪ	OI	이중 모음	choice	O
u	u	후설 원순 고장모음	goose	u
ʊ	U	근후설 원순 근고모음	foot	u
ʌ	V	후설 평순 중저모음	strut	E

스웨덴어(sv-SE)

다음 표에는 Amazon Polly에서 지원하는 스웨덴어 음성에 대해 IPA(International Phonetic Alphabet) 음소와 X-SAMPA(Extended Speech Assessment Methods Phonetic Alphabet) 기호 및 해당하는 viseme이 나와 있습니다.

음소/Viseme 표

IPA	X-SAMPA	설명	예	Viseme
자음				
b	b	유성 양순 파열음	bil	p
d	d	유성 치경 파열음	dal	t
ɖ	d`	유성 권설 파열음	bord	t
f	f	무성 순치 마찰음	fil	f
g	g	유성 연구개 파열음	gås	k
h	h	무성 성문 마찰음	hal	k

IPA	X-SAMPA	설명	예	Viseme
j	j	경구개 접근음	jag	i
k	k	무성 연구개 파열음	kal	k
l	l	치경 설측 접근음	lös	t
ɭ	l`	권설 설측 접근음	härlig	t
m	m	양순 비음	mil	p
n	n	치경 비음	nålar	t
ɳ	n`	권설 비음	barn	t
ŋ	N	연구개 비음	ring	k
p	p	무성 양순 파열음	pil	p
r	r	치경 전동음	ris	r
s	s	무성 치경 마찰음	sil	s
ɕ	s\	무성 치경구개 마찰음	tjock	J
ʂ	s`	무성 권설 마찰음	fors, schlager	S
t	t	무성 치경 파열음	tal	t
ʈ	t`	무성 권설 파열음	hjort	t
v	v	유성 순치 마찰음	vår	f
w	w	양순 연구개 접근음	aula, airways	u
ɧ	x\	무성 경구개-연구개 마찰음	sjuk	k
모음				
ø	2	전설 원순 중고모음	föll, förr	o
ø	2:	전설 원순 중고장모음	föl, nöt, för	o
ɵ	8	중설 원순 중고모음	buss, full	o
ə	@	중설 중모음	pojken	@
ʉː	}:	중설 원순 고장모음	hus, ful	u
a	a	전설 비원순 저모음	hall, matt	a
æ	{	전설 비원순 근저모음	herr	a
ɑː	A:	후설 비원순 저장모음	hal, mat	a
eː	e:	전설 비원순 중고장모음	vet, hel	e
ɛ	E	전설 비원순 중저모음	vett, rätt, hetta, häll	E

IPA	X-SAMPA	설명	예	Viseme
ɛ:	E:	전설 비원순 중저장모음	säl, häl, här	E:
i:	i:	전설 비원순 고장모음	vit, sil	i:
ɪ	I	근전설 비원순 근고모음	vitt, sill	i
o:	o:	후설 원순 중고장모음	hål, mål	o
ɔ	O	후설 원순 중저모음	håll, moll	O
u:	u:	후설 원순 고장모음	sol, bot	u
ʊ	U	근후설 원순 근고모음	bott	u
y	y	전설 원순 고모음	bytt	u
y:	y:	전설 원순 고장모음	syl, syl	u

터키어(tr-TR)

다음 표에는 Amazon Polly에서 지원하는 터키어 음성에 대해 IPA(International Phonetic Alphabet) 음소와 X-SAMPA(Extended Speech Assessment Methods Phonetic Alphabet) 기호 및 해당하는 viseme이 나와 있습니다.

음소/Viseme 표

IPA	X-SAMPA	설명	예	Viseme
자음				
ɾ	4	치경 탄음	durum	t
ɾ̥	4_0_r	무성 마찰 치경 탄음	bir	t
ɾ̞	4_r	마찰 치경 탄음	raf	t
b	b	유성 양순 파열음	raf	p
c	c	무성 경구개 파열음	kedi	k
d	d	유성 치경 파열음	dede	t
d͡ʒ	dZ	유성 후치경 파찰음	cam	S
f	f	무성 순치 마찰음	fare	f
g	g	유성 연구개 파열음	galibi	k
h	h	무성 성문 마찰음	hasta	k
j	j	경구개 접근음	yat	i
ɟ	J\	유성 경구개 파열음	genç	J
k	k	무성 연구개 파열음	akıl	k

IPA	X-SAMPA	설명	예	Viseme
l	l	치경 설측 접근음	lale	t
ɬ	5	연구개음화 치경 설측 접근음	labirent	t
m	m	양순 비음	maaş	p
n	n	치경 비음	anı	t
p	p	무성 양순 파열음	ip	p
s	s	무성 치경 마찰음	ses	s
ʃ	S	무성 후치경 마찰음	aşı	S
t	t	무성 치경 파열음	ütü	t
t͡ʃ	tS	무성 후치경 파찰음	çaba	S
v	v	유성 순치 마찰음	ekvator, kahveci, akvaryum, isveçli, teşviki, cetvel	f
z	z	유성 치경 마찰음	ver	s
ʒ	Z	유성 후치경 마찰음	azık	S
모음				
ø	2	전설 원순 중고모음	göl	0
œ	9	전설 원순 중저모음	banliyö	O
a	a	전설 비원순 저모음	kal	a
a:	a:	전설 비원순 저장모음	davacı	a
æ	{	전설 비원순 근저모음	özlem, güvenlik, gürel, somersault	a
e	e	전설 비원순 중고모음	keçi	e
ɛ	E	전설 비원순 중저모음	dede	E
i	i	전설 비원순 고모음	bir	i
i:	i:	전설 비원순 고장모음	izah	i
ɪ	I	근전설 비원순 근고모음	keçi	i
ɯ	M USD	후설 비원순 고모음	kıl	i
o	o	후설 원순 중고모음	kol	o
o:	o:	후설 원순 중고장모음	dolar	o
u	u	후설 원순 고모음	durum	u
u:	u:	후설 원순 고장모음	ruhum	u

IPA	X-SAMPA	설명	예	Viseme
ʊ	U	근후설 원순 근고모음	dolu	u
y	y	전설 원순 고모음	güvenlik	u
ʏ	Y	근전설 원순 근고모음	aşı	u

웨일스어(cy-GB)

다음 표에는 Amazon Polly에서 지원하는 웨일스어 음성에 대해 IPA(International Phonetic Alphabet) 음소와 X-SAMPA(Extended Speech Assessment Methods Phonetic Alphabet) 기호 및 해당하는 viseme이 나와 있습니다.

음소/Viseme 표

IPA	X-SAMPA	설명	예	Viseme
자음				
b	b	유성 양순 파열음	baban	p
d	d	유성 치경 파열음	deg	t
ʤ	dZ	유성 후치경 파찰음	garej	S
ð	D	유성 치 마찰음	deuddeg	T
f	f	무성 순치 마찰음	ffacs	f
g	g	유성 연구개 파열음	gadael	k
h	h	무성 성문 마찰음	haearn	k
j	j	경구개 접근음	astudio	i
k	k	무성 연구개 파열음	cant	k
l	l	치경 설측 접근음	lan	t
ɬ	K USD	무성 치경 설측 마찰음	llan	t
m	m	양순 비음	mae	p
m̥	m_0	무성 양순 비음	ymhen	p
n	n	치경 비음	naw	t
n̥	n_0	무성 치경 비음	anhawster	t
ŋ	N	연구개 비음	argyfwng	k
ŋ̊	N_0	무성 연구개 비음	anghenion	k
p	p	무성 양순 파열음	pump	p
r	r	치경 전동음	rhoi	r
r̥	r_0	무성 치경 전동음	garw	r

IPA	X-SAMPA	설명	예	Viseme
s	s	무성 치경 마찰음	saith	s
ʃ	S	무성 후치경 마찰음	siawns	S
t	t	무성 치경 파열음	tegan	t
t͡ʃ	tS	무성 후치경 파찰음	cytsain	S
θ	T	무성 치 마찰음	aberth	T
v	v	유성 순치 마찰음	prawf	f
w	w	양순 연구개 접근음	rhagweld	u
χ	X	무성 구개수 마찰음	chwech	k
z	z	유성 치경 마찰음	aids	s
ʒ	Z	유성 후치경 마찰음	rouge	S
모음				
ə	@	중설 중모음	ychwanega	@
a	a	전설 비원순 저모음	acen	a
ai	ai	이중 모음	dau	a
au	au	이중 모음	awdur	a
ɑ:	A:	후설 비원순 저장모음	mab	a
ɑːɨ	A:1	이중 모음	aelod	a
e:	e:	전설 비원순 중고장모음	peth	e
ɛ	E	전설 비원순 중저모음	pedwar	E
ɛi	Ei	이중 모음	beic	E
i:	i:	전설 비원순 고장모음	tri	i
ɪ	I	근전설 비원순 근고모음	miliwn	i
ɨu	1u	이중 모음	unigryw	i
o:	o:	후설 원순 중고장모음	oddi	o
ɔ	O	후설 원순 중저모음	oddieithr	O
ɔi	Oi	이중 모음	troi	O
ɔu	Ou	이중 모음	rownd	O
u:	u:	후설 원순 고장모음	cwch	u
ʊ	U	근후설 원순 근고모음	acwstig	u
ʊi	Ui	이중 모음	wyth	u

SSML 사용

Amazon Polly는 SSML 버전 1.1을 준수하는 일반 텍스트 입력과 SSML(Speech Synthesis Markup Language) 문서 모두에서 스피치를 생성합니다. SSML 태그를 사용하여 발음, 볼륨 및 스피치 속도와 같은 스피치 측면을 사용자 정의하고 제어할 수 있습니다.

Amazon Polly는 다음 W3C 권장 사항에서 정의한 SSML 1.1을 지원합니다.

- Speech Synthesis Markup Language (SSML) Version 1.1, W3C Recommendation 7 September 2010

SSML W3C 권장 사항의 일부 요소는 지원되지 않습니다. 자세한 내용은 Amazon Polly 제한 (p. 130) 섹션을 참조하십시오.

이번 섹션에서는 스피치 출력 생성 및 제어에 사용할 수 있는 SSML의 간단한 예를 제공합니다. 또한 이러한 예를 테스트할 수 있는 `synthesize-speech` AWS CLI 명령을 제공합니다.

항목

Amazon Polly 콘솔과 함께 SSML 사용

Amazon Polly는 W3C에서 정의한 버전 1.1 SSML을 지원합니다. 이 섹션에서는 Amazon Polly 콘솔에서 스피치 합성을 위해 SSML 입력을 사용하는 방법에 대해 설명합니다.

Amazon Polly 콘솔과 함께 SSML 사용

다음 절차에서는 SSML 입력을 사용하여 스피치를 합성합니다. 아래의 3단계와 4단계를 제외한 이 예의 단계는 연습 2: 스피치 합성(일반 텍스트 입력) (p. 8)의 단계와 동일합니다.

Amazon Polly 콘솔을 사용하여 스피치를 합성하려면(SSML 입력)

이 예에서는 SSML 태깅을 사용하여 "World Wide Web Consortium"을 "W3C"로 대체합니다. 이 연습의 결과를 미국 영어 및 다른 언어의 콘솔을 사용하여 어휘 적용(스피치 합성) (p. 86) 결과와 모두 비교하십시오.

1. AWS Management Console에 로그인한 후 https://console.aws.amazon.com/polly/에서 Amazon Polly 콘솔을 엽니다.
2. 필요한 경우 [Text-to-Speech] 탭을 선택합니다.
3. [SSML] 탭을 선택합니다.
4. 다음 텍스트를 텍스트 상자에 입력하거나 붙여 넣습니다.

```
<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, "Score!" quite loudly. When his boss stared at him, he repeated
     <amazon:effect name="whispered">"Score"</amazon:effect> in a whisper.
```

```
</speak>
```

SSML 태그는 텍스트가 지정된 방식으로 렌더링하도록 Amazon Polly에 알립니다.

- `<break time="1s"/>`는 처음 두 문장 사이에서 1초 동안 일시 중지하도록 Amazon Polly에 지시합니다.
- `<sub alias="World Wide Web Consortium">W3C</sub>`는 "World Wide Web Consortium"을 두문자어 "W3C"로 대체하도록 Amazon Polly에 지시합니다.
- `<amazon:effect name="whispered">Score</amazon:effect>`는 속삭이는 음성으로 두 번째 "점수"를 말하도록 Amazon Polly에 지시합니다.

예에 사용된 SSML에 대한 자세한 내용은 지원되는 SSML 태그 (p. 69) 섹션을 참조하십시오.

5. [Choose a language and region]에서 [English US]를 선택한 다음 원하는 음성을 선택합니다.
6. 스피치를 즉시 들으려면 [Listen to speech]를 선택합니다.
7. 음성 파일을 저장하려면 음성이 원하는 형식인 경우 [Download [format]]를 선택합니다. 음성이 원하는 형식이 아니라면 [Change file format]을 선택하여 원하는 파일 형식을 선택한 후 [Change]를 선택합니다. [Download [format]]를 선택합니다.

관련 콘솔 예

- 연습 2: 스피치 합성(일반 텍스트 입력) (p. 8)
- 콘솔을 사용하여 어휘 적용(스피치 합성) (p. 86)

Note

AWS CLI에서 입력 텍스트를 입력하면 입력 텍스트 주위에 따옴표가 사용되어 주변 코드와 구별됩니다. Amazon Polly 콘솔에는 코드가 표시되지 않으므로 콘솔의 입력 텍스트 주위에 따옴표가 사용되지 않습니다.

다음 단계

콘솔을 사용하여 어휘 적용(스피치 합성) (p. 86)

AWS CLI와 함께 SSML 사용

다음 항목에서는 Amazon Polly의 AWS CLI와 함께 SSML 입력을 사용하는 방법을 보여줍니다.

예 1: Synthesize-Speech 명령을 통해 SSML 전달

다음 `synthesize-speech` 명령에서는 필수 여는 태그와 닫는 태그인 `<speak></speak>`만 있고 따옴표로 둘러싸인 간단한 SSML 문자열을 지정합니다. (필요한 경우 전체 문서 헤더도 지정할 수 있습니다). 일반 텍스트가 기본이므로, 이 명령은 입력 텍스트가 SSML임을 나타내기 위해 `--text-type` 매개 변수도 지정합니다. SSML 문자열에만 필요한 요소는 입력 텍스트(생성된 스피치로 지정됨), `output-format` 및 `voice-id`입니다.

Important

Amazon Polly 콘솔에서는 입력 텍스트 주위에 따옴표를 사용하지 않지만 AWS CLI 또는 다른 코드를 사용할 때는 따옴표가 필요합니다. 이 사항은 일반 텍스트와 SSML에 모두 해당됩니다. 전체 입력 텍스트 주위의 따옴표와 텍스트 내의 따옴표를 구별해야 합니다.

예를 들어, 전체 입력 텍스트 주위에는 작은 따옴표(')를 사용하고 내부 태그 내에는 표준 따옴표(")를 사용할 수 있습니다. 그 반대로도 사용할 수 있습니다. (AWS CLI를 사용할 때는 두 옵션 모두 Unix, Linux 및 macOS에 사용됩니다. Windows 명령 프롬프트를 사용하면 입력 텍스트용 표준 따옴표와 내부 태그용 작은 따옴표가 필요합니다.
따라서 다음 형식 중 하나를 사용할 수 있습니다.

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

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

다음은 Unix, Linux, macOS용 형식으로 지정된 AWS CLI 예제입니다. Windows의 경우 각 줄 끝에 있는 백슬래시(\) Unix 연속 문자를 캐럿(^)으로 바꿉니다. 입력 텍스트는 큰 따옴표(")로 감싸고 내부 태그에는 작은 따옴표(')를 사용합니다.

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

`speech.mp3` 결과 파일을 재생하여 합성된 스피치를 확인합니다.

예 2: 전체 SSML 문서 합성

이 예에서는 SSML 콘텐츠를 파일에 저장하고 `synthesize-speech` 명령에 파일 이름을 지정합니다. 이 예에서는 다음 SSML을 사용합니다:

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

`xml:lang` 속성은 `en-US`(미국 영어)를 입력 텍스트의 언어로 지정합니다. 입력 텍스트의 언어와 선택한 음성의 언어가 `SynthesizeSpeech` 작업에 미치는 영향에 대한 자세한 내용은 xml:lang 속성 사용 (p. 68)을 참조하십시오.

SSML을 테스트하려면

1. SSML을 파일(`example.xml`)에 저장합니다.
2. XML 파일이 저장된 경로에서 다음 `synthesize-speech` 명령을 실행하고 입력으로 SSML을 지정합니다. 이 명령은 실제 입력 텍스트를 포함하는 대신 파일을 가리키므로 따옴표가 필요하지 않습니다.

   다음은 Unix, Linux, macOS용 형식으로 지정된 AWS CLI 예제입니다. Windows의 경우 각 줄 끝에 있는 백슬래시(\) Unix 연속 문자를 캐럿(^)으로 바꿉니다. 입력 텍스트는 큰 따옴표(")로 감싸고 내부 태그에는 작은 따옴표(')를 사용합니다.

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

3. `speech.mp3` 파일을 재생하여 합성된 스피치를 확인합니다.

예 3: 공통 SSML 태그 사용

이 섹션에서는 특정 결과를 보관하기 위해 일부 공통 SSML 태그를 사용하는 방법에 대해 설명합니다. 더 많은 예를 보려면 Speech Synthesis Markup Language (SSML) Version 1.1을 참조하십시오.

이 섹션에서는 `synthesize-speech` 명령을 사용하여 예를 테스트할 수 있습니다.

<break> 요소 사용

다음 SSML `synthesize-speech` 명령은 `<break>` 요소를 사용하여 결과 음성에서 "Hello"와 "World" 단어 사이에 300밀리초 지연을 추가합니다.

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

다음은 Unix, Linux, macOS용 형식으로 지정된 AWS CLI 예제입니다. Windows의 경우 각 줄 끝에 있는 백슬래시(\) Unix 연속 문자를 캐럿(^)으로 바꿉니다. 입력 텍스트는 큰 따옴표(")로 감싸고 내부 태그에는 작은 따옴표(')를 사용합니다.

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

`speech.mp3` 결과 파일을 재생하여 합성된 스피치를 확인합니다.

<prosody> 요소 사용

`<prosody>` 요소를 사용하여 음색, 말하기 속도 및 음성 볼륨을 제어할 수 있습니다.

- 다음 SSML은 `<prosody>` 요소를 사용하여 볼륨을 제어합니다.

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

  다음은 Unix, Linux, macOS용 형식으로 지정된 AWS CLI 예제입니다. Windows의 경우 각 줄 끝에 있는 백슬래시(\) Unix 연속 문자를 캐럿(^)으로 바꿉니다. 입력 텍스트는 큰 따옴표(")로 감싸고 내부 태그에는 작은 따옴표(')를 사용합니다.

  ```
  aws polly synthesize-speech \
  --text-type ssml \
  --text '<speak><prosody volume="+20dB">Hello world</prosody></speak>' \
  --output-format mp3 \
  --voice-id Joanna \
  speech.mp3
  ```

- 다음 SSML은 `<prosody>` 요소를 사용하여 음색을 제어합니다.

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

```
</speak>
```

다음은 Unix, Linux, macOS용 형식으로 지정된 AWS CLI 예제입니다. Windows의 경우 각 줄 끝에 있는 백슬래시(\) Unix 연속 문자를 캐럿(^)으로 바꿉니다. 입력 텍스트는 큰 따옴표(")로 감싸고 내부 태그에는 작은 따옴표(')를 사용합니다.

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

- 다음 SSML은 <prosody> 요소를 사용하여 스피치 속도를 지정합니다.

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

다음은 Unix, Linux, macOS용 형식으로 지정된 AWS CLI 예제입니다. Windows의 경우 각 줄 끝에 있는 백슬래시(\) Unix 연속 문자를 캐럿(^)으로 바꿉니다. 입력 텍스트는 큰 따옴표(")로 감싸고 내부 태그에는 작은 따옴표(')를 사용합니다.

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

- 다음 예와 같이 <prosody> 요소에서 속성을 여러 개 지정할 수 있습니다.

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

다음은 Unix, Linux, macOS용 형식으로 지정된 AWS CLI 예제입니다. Windows의 경우 각 줄 끝에 있는 백슬래시(\) Unix 연속 문자를 캐럿(^)으로 바꿉니다. 입력 텍스트는 큰 따옴표(")로 감싸고 내부 태그에는 작은 따옴표(')를 사용합니다.

```
aws polly synthesize-speech \
--text-type ssml \
--text '<speak><prosody volume="x-loud" pitch="x-high" rate="x-fast">Hello world</
prosody></speak>' \
--output-format mp3 \
--voice-id Joanna \
speech.mp3
```

speech.mp3 결과 파일을 재생하여 합성된 스피치를 확인합니다.

속삭이는 음성 사용

다음 synthesize-speech 명령은 <amazon:effect name="whispered"> 요소를 사용하여 결과 음성에서 "little lamb" 단어를 속삭이는 음성으로 말하도록 합니다.

```
<speak>
```

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

<prosody> 요소를 사용하여 속삭이는 스피치를 약간 느리게 하면 이 효과를 향상시킬 수 있습니다.

다음은 Unix, Linux, macOS용 형식으로 지정된 AWS CLI 예제입니다. Windows의 경우 각 줄 끝에 있는 백슬래시(\) Unix 연속 문자를 캐럿(^)으로 바꿉니다. 입력 텍스트는 큰 따옴표(")로 감싸고 내부 태그에는 작은 따옴표(')를 사용합니다.

```
aws polly synthesize-speech \
--text-type ssml \
--text '<speak> Mary has a <prosody rate="-10%"><amazon:effect name="whispered"> \
little lamb.</amazon:effect></prosody></speak>' \
--output-format mp3 \
--voice-id Joanna \
speech.mp3
```

`speech.mp3` 결과 파일을 재생하여 합성된 스피치를 확인합니다.

<emphasis> 요소 사용

<emphasis> 요소를 사용하면 특정 단어나 구를 말할 때 적용할 강세 또는 강조 부분을 지정할 수 있습니다.

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

다음은 Unix, Linux, macOS용 형식으로 지정된 AWS CLI 예제입니다. Windows의 경우 각 줄 끝에 있는 백슬래시(\) Unix 연속 문자를 캐럿(^)으로 바꿉니다. 입력 텍스트는 큰 따옴표(")로 감싸고 내부 태그에는 작은 따옴표(')를 사용합니다.

```
aws polly synthesize-speech \
--text-type ssml \
--text '<speak><emphasis level="strong">Hello</emphasis> world how are you?</speak>' \
--output-format mp3 \
--voice-id Joanna \
speech.mp3
```

`speech.mp3` 결과 파일을 재생하여 합성된 스피치를 확인합니다.

예 4: 단어를 말하는 방식 제어

이 예에서는 Amazon Polly에서 특정 단어를 말하는 방식을 제어하는 데 사용할 수 있는 일부 공통 SSML 태그에 대해 설명합니다. Amazon Polly는 특정한 방식으로 또는 단어가 발음되는 방식에 따라 단어를 해석합니다.

<say-as> 요소 사용

<say-as> 요소를 사용하면 해당 요소 내에 포함된 텍스트 유형에 대한 정보를 제공할 수 있습니다.

예를 들어 다음 SSML에서, `<say-as>`는 4/6 텍스트를 특정 방식으로 해석하도록 지정합니다. `interpret-as="date" format="dm"` 속성은 이 텍스트를 월/일 형식의 날짜 값으로 말하도록 지정합니다.

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

```
</speak>
```

이 요소를 사용하여 분수, 전화번호, 측정 단위 등을 음성으로 변환할 수도 있습니다.

다음은 Unix, Linux, macOS용 형식으로 지정된 AWS CLI 예제입니다. Windows의 경우 각 줄 끝에 있는 백슬래시(\) Unix 연속 문자를 캐럿(^)으로 바꿉니다. 입력 텍스트는 큰 따옴표(")로 감싸고 내부 태그에는 작은 따옴표(')를 사용합니다.

```
aws polly synthesize-speech \
--text-type ssml \
--text '<speak>Today is <say-as interpret-as="date" format="md" >4/6</say-as></speak>' \
--output-format mp3 \
--voice-id Joanna \
speech.mp3
```

결과 스피치는 "Today is June 4th"라고 말합니다. `<say-as>` 태그는 `interpret-as` 속성을 통해 추가 컨텍스트를 제공하여 텍스트가 해석되는 방법에 대해 설명합니다.

`speech.mp3` 결과 파일을 재생하여 합성된 스피치를 확인합니다.

이 요소에 대한 자세한 내용은 <say-as> (p. 73) 단원을 참조하십시오.

xml:lang 속성 사용

`xml:lang` 속성을 사용하여 대상 언어를 지정하면 입력 텍스트 언어에서 외국어의 단어 발음을 향상할 수 있습니다. 이렇게 하면 TTS 엔진이 대상 언어의 특정 단어에 대해 다른 발음 규칙을 적용합니다. 아래의 예에서는 `synthesize-speech` 호출에서 지정할 수 있는 입력 텍스트와 음성에 다른 언어와의 조합을 보여줍니다. `synthesize-speech` 명령으로 이러한 예를 테스트하여 결과를 확인할 수 있습니다.

사용 가능한 전체 언어 목록은 Amazon Polly에서 지원하는 언어 (p. 128) 단원을 참조하십시오.

이 예에서는 선택한 음성이 미국 영어 음성입니다. Amazon Polly는 입력 텍스트가 선택한 음성과 동일한 언어라고 가정합니다. 특정 단어에 대한 스페인어 발음을 얻으려면 대상 단어를 스페인어로 표시해야 합니다.

```
aws polly synthesize-speech \
--text-type ssml \
--text '<speak>That restaurant is terrific. <lang xml:lang="es-ES">Mucho gusto.</lang></
speak>' \
--output-format mp3 \
--voice-id Joanna \
speech.mp3
```

입력 텍스트의 언어가 지정되었으므로 Amazon Polly는 스페인어 결과 음소를 음향 거리가 가장 짧은 영어 음소로 매핑합니다. 결과적으로, Salli는 정확한 스페인어 발음을 알고 있지만 미국 영어 억양을 아는 미국 원어민으로서 텍스트를 읽습니다.

> **Note**
>
> 이 연습은 Amazon Polly 언어 포트폴리오에서 사용할 수 있는 언어 쌍으로 제한됩니다. 일부 언어 쌍은 언어의 음운 구조 때문에 다른 언어보다 효과적으로 작동합니다.

`speech.mp3` 결과 파일을 재생하여 합성된 스피치를 듣습니다.

Amazon Polly의 SSML 태그

SSML로 향상된 입력 텍스트를 사용하면 Amazon Polly가 제공한 텍스트에서 음성을 생성하는 방법을 추가로 제어할 수 있습니다.

예를 들어, 텍스트에 일시 중지를 길게 포함하거나 스피치 속도 또는 음색 변경과 같이 다른 방법으로 텍스트를 변경할 수 있습니다. Amazon Polly는 Speech Synthesis Markup Language (SSML) Version 1.1, W3C Recommendation에 정의된 SSML 마크업 태그의 하위 집합과 함께 이러한 유형의 제어 기능을 제공합니다.

지원되는 SSML 태그

Amazon Polly는 다음 SSML을 지원합니다.

루트

- <speak> (p. 69)

표준 태그

- <break> (p. 69)
- <emphasis> (p. 70)
- <lang> (p. 70)
- <mark> (p. 71)
- <p> (p. 72)
- <phoneme> (p. 72)
- <prosody> (p. 72)
- <s> (p. 73)
- <say-as> (p. 73)
- <sub> (p. 75)
- <w> (p. 75)

Amazon Polly 전용 태그

- <amazon:breath> 및 <amazon:auto-breaths> (p. 75)
- <amazon:effect name="drc"> (p. 79)
- <amazon:effect phonation="soft"> (p. 80)
- <amazon:effect vocal-tract-length> (p. 80)
- <amazon:effect name="whispered"> (p. 81)

Amazon Polly은 텍스트를 처리할 때 입력 텍스트에서 지원되지 않는 SSML 태그를 무시합니다.

<speak>

`<speak>` 태그는 모든 Amazon Polly SSML 텍스트의 루트 요소입니다. 말하려는 모든 SSML로 향상된 텍스트를 이 태그 내에 포함시켜야 합니다.

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

<break>

이 태그는 스피치에서 일시 중지를 나타냅니다. 일시 중지의 길이는 `strength` 또는 `time` 속성으로 설정할 수 있습니다. 속성에 `break` 태그가 사용되지 않으면 기본 `<break strength="medium">`이 사용됩니다.

다음 값은 `strength` 속성과 함께 사용할 수 있습니다.

- none: 일시 중지가 발생하지 않습니다. 일반적으로 발생하는 일시 중지(예 : 마침표 다음)를 제거하는 데 사용할 수 있습니다.
- x-weak: none과 동일합니다.
- weak: 인접한 단어를 쉼표 하나로 구분하여 처리합니다.
- medium: weak와 동일합니다.
- strong: 인접한 단어를 문장 나누기로 구분한 것처럼 처리합니다(<s> 태그와 동일함).
- x-strong: 인접한 단어를 단락 나누기로 구분한 것처럼 처리합니다(<p> 태그와 동일함).

다음 값은 time 속성과 함께 사용할 수 있습니다.

- s: 일시 중시 기간(초)입니다. 일시 중지 기간은 최대 10s입니다.
- ms: 일시 중시 기간(밀리초)입니다. 일시 중지 기간은 최대 10000ms입니다.

예:

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

<emphasis>

<emphasis> 태그는 태그가 지정된 단어나 구를 강조합니다. 텍스트를 강조하면 스피치의 속도와 볼륨이 변경됩니다. 많이 강조할수록 텍스트를 더 크고 느리게 읽습니다. 덜 강조하면 더 작고 빠르게 읽습니다. level 속성은 텍스트에 적용할 강조 수준을 나타냅니다.

세 가지 level 옵션을 사용할 수 있습니다.

- Strong: 볼륨을 높이고 말하기 속도가 느려지므로 스피치가 더 크고 느려집니다.
- Moderate: 볼륨을 높이고 말하기 속도가 느려지지만 strong으로 설정할 때보다 덜합니다. 기본값이며 수준이 제공되지 않는 경우 이 옵션이 사용됩니다.
- Reduced: 볼륨을 낮추고 말하기 속도를 높입니다. 스피치가 부드럽고 빨라집니다.

> **Note**
>
> 지정된 음성의 정상적인 말하기 속도 및 볼륨은 moderate 수준과 reduced 수준 사이입니다.

예:

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

<lang>

이 태그는 특정 단어나 구의 언어를 나타냅니다. 일반적으로 외국어(단어 및 구)는 <lang> 태그에 포함되면 더 잘 들리도록 렌더링됩니다. 발음을 렌더링할 언어에 xml:lang 속성을 사용하여 지정해야 합니다.

사용 가능한 전체 언어 목록은 Amazon Polly에서 지원하는 언어 (p. 128) 단원을 참조하십시오.

단어나 구가 렌더링되는 실제 언어는 <lang> 태그의 언어가 아니라 voice-id의 언어가 사용됩니다. 이 태그는 voice-id의 언어와 다른 경우에만 텍스트의 발음을 설정하는 데 사용됩니다.

예:

`voice-id`가 Joanna(미국 영어)이면 다음을 실행합니다.

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

Amazon Polly는 프랑스어 억양을 사용하지 않고 Joanna 음성으로 이 텍스트를 말합니다.

동일한 음성이 다음 `<lang>` 태그과 함께 사용되는 경우:

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

Amazon Polly는 미국식 억양의 프랑스어를 사용하여 Joanna 음성으로 이 텍스트를 발음합니다.

그러나 Joanna의 음성은 원어민 프랑스어가 아니기 때문에 발음은 일반 음성의 언어를 따릅니다. 예를 들면, 완벽한 프랑스어 발음에서는 단어 adore에 목젖 울림 /R/이 있지만, Joanna의 미국식 영어 음성은 이 음소를 해당하는 /r/ 소리로 발음합니다.

또 다른 예로 Giorgio(이탈리아인)의 `voice-id`는 다음 텍스트에 사용됩니다.

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

Amazon Polly는 비 이탈리아어 발음을 사용하지 않고 Giorgio 음성으로 이 텍스트를 말합니다.

동일한 음성이 다음 `<lang>` 태그과 함께 사용되는 경우:

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

Amazon Polly는 이탈리아식 억양의 영어를 사용하여 Giorgio 음성으로 이 텍스트를 발음합니다.

<mark>

이 태그는 사용자가 텍스트 내에 사용자 정의 태그를 배치할 수 있는 기능을 제공합니다. Amazon Polly에서 태그에 수행되는 작업은 없지만, SSML 메타데이터가 반환되면 이 태그의 위치도 반환됩니다.

이 태그는 다음과 같은 형식으로 사용자가 지정하면 무엇이든 될 수 있습니다.

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

예를 들어, 태그 이름이 "animal"인 경우 입력 텍스트는 다음과 같습니다.

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

다음 SSML 메타데이터는 Amazon Polly에서 반환될 수 있습니다.

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

<p>

이 태그는 텍스트에서 단락을 나타냅니다. 이는 <break strength="x-strong"/>으로 일시 중지를 지정하는 것과 동일합니다. <s> 태그와 같이 해당 문장을 이 태그 안에 묶어야 합니다.

```
<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>
```

<phoneme>

이 태그는 표시된 텍스트에 대한 음성 발음(음소)을 제공합니다.

두 속성에는 phoneme 태그가 필요합니다.

- alphabet
  - ipa - IPA(International Phonetic Alphabet) 시스템을 사용하는 음소를 나타냅니다.
  - x-sampa - X-SAMPA(Extended Speech Assessment Methods Phonetic Alphabet) 시스템을 사용하는 음소를 나타냅니다.
- ph
  - 발음에 사용되는 발음 기호를 나타냅니다. Amazon Polly는 표준 IPA 및 X-SAMPA 발음 기호를 지원합니다. 자세한 내용은 Amazon Polly에서 사용하는 발음 표 (p. 82) 단원을 참조하십시오.

phoneme 태그를 사용하면 Amazon Polly는 태그 내에 포함된 텍스트에 기본적으로 연결된 발음 대신 ph 속성에서 제공된 발음을 사용합니다. 그러나 태그 내에 사람이 읽을 수 있는 텍스트를 계속 제공해야 합니다.

예를 들어, 단어 "pecan"은 두 개의 다른 방식으로 발음될 수 있습니다. 다음 예에서는 단어 "pecan"에 대해 각 행마다 서로 다른 사용자 정의 발음이 지정됩니다. 따라서 Amazon Polly는 기본 발음 대신 ph 속성에 제공된 발음을 사용합니다.

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

> **Note**
>
> 경우에 따라 음성 기호에 작은따옴표 또는 큰따옴표가 포함될 수 있으며, 이 기호는 말하는 문자열을 표시하는 데 사용되는 따옴표와 충돌합니다. 이 경우 문자열 기호와 충돌하지 않도록 기호를 사용하려면 백슬래시를 사용합니다.
> 예를 들어 제1 강세를 나타내는 IPA 기호(작은따옴표)를 사용하는 경우 \'를 사용합니다. r 음색을 띤 중설 비원순 중저모음을 나타내는 X-SAMPA 기호(3')는 3\'로 표시합니다. 작은따옴표나 큰따옴표 문자열 기호는 변경되지 않습니다.

<prosody>

prosody 태그를 사용하여 전달하는 텍스트의 볼륨, 속도 및 음색을 제어할 수 있습니다.

다음 값은 volume 속성에 사용할 수 있으며 스피치의 볼륨을 수정할 수 있습니다.

- default: 볼륨을 현재 음성의 기본값으로 재설정합니다.
- silent, x-soft, soft, medium, loud, x-loud: 볼륨을 현재 음성에 사전 정의된 값으로 설정합니다.

- `+ndB`, `-ndB`: 볼륨을 현재 볼륨 레벨에 상대적으로 변경합니다. 값 "+0dB"은 볼륨 변경 없음, "+6dB"은 현재 진폭의 약 2배, "-6dB"은 현재 진폭의 약 절반을 의미합니다.

다음 값은 `rate` 속성에 사용할 수 있으며 스피치의 속도를 수정할 수 있습니다.

- `x-slow`, `slow`, `medium`, `fast`, `x-fast`: 현재 음성의 스피치 속도를 사전 정의된 값으로 설정합니다.

다음 값은 `pitch` 속성에 사용할 수 있으며 음성의 음색을 수정할 수 있습니다.

- `default`: 현재 음성의 음색을 기본 음색으로 재설정합니다.
- `x-low`, `low`, `medium`, `high`, `x-high`: 현재 음성의 음색을 사전 정의된 값으로 설정합니다.
- `+n%` 또는 `-n%`: 현재 음성의 현재 음색 레벨을 상대적 백분율로 변경하여 음색을 조절합니다. 예를 들어 `+4%` 또는 `-2%` 등으로 지정할 수 있습니다.

예를 들어, 구절의 운율을 다음과 같은 방식으로 설정할 수 있습니다.

```
<speak>
     Prosody can be used to change the way words sound. The following words are
     <prosody volume="x-loud"> quite a bit louder than the rest of this passage.
     </prosody> Each morning when I wake up, <prosody rate="x-slow">I speak
     quite slowly and deliberately until I have my coffee.</prosody> I can also
     change the pitch of my voice using prosody. Do you like <prosody pitch="+5%">
     speech with a pitch that is higher,</prosody> or <prosody pitch="-10%">
     is a lower pitch preferable?</prosody>
</speak>
```

<s>

이 태그는 텍스트에서 문장을 나타냅니다. 이는 다음과 동일합니다.

- 마침표(.)로 끝나는 문장
- `<break strength="strong"/>`으로 일시 중지 지정

`<break strength="strong"/>` 태그와 달리, 문장을 이 태그 안에 묶어야 합니다.

이 예에서 `<s>` 태그는 첫 번째 문장과 두 번째 문장 다음에 모두 일시 중지를 짧게 생성합니다. 최종 문장에는 `<s>` 태그가 없지만 마침표를 포함하기 때문에 문장 뒤에 일시 중지가 짧게 있습니다.

```
<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>
```

<say-as>

이 태그는 입력 텍스트가 해석되는 방식을 나타냅니다. 이 태그를 사용하면 추가 컨텍스트를 제공하여 Amazon Polly가 텍스트를 렌더링하는 방법에 대한 모든 불확실성을 제거할 수 있습니다.

`say-as` 태그를 사용하면 Amazon Polly가 `interpret-as` 속성과 함께 텍스트를 해석하는 방법을 지정해야 합니다.

다음 값은 `interpret-as` 속성과 함께 사용할 수 있습니다.

- `character` 또는 `spell-out`: a-b-c와 같이 각 문자를 발음합니다.
- `cardinal` 또는 `number`: 1,234와 같이 값을 기수로 발음합니다.
- `ordinal`: 1,234번째와 같이 값을 서수로 발음합니다.
- `digits`: 1-2-3-4와 같이 숫자의 각 자릿수를 개별적으로 발음합니다.
- `fraction`: 값을 분수로 발음합니다. 이 태그는 3/20과 같은 일반 분수와 2 1/2과 같은 혼합 분수 모두에 적용됩니다(아래 참조).
- `unit`: 값을 측정값으로 해석합니다. 값은 `1meter`와 같이 숫자 또는 분수 뒤에 단위(사이에 공백 없이)가 있어야 하고, 단위는 한 개만 사용해야 합니다.
- `date`: 값을 날짜로 해석합니다. 날짜의 형식은 형식 속성과 함께 지정되어야 합니다(아래 참조).
- `time`: `1'21"`과 같이 값을 기간(분과 초)으로 해석합니다.
- `address`: 값을 거리 주소의 부분으로 해석합니다.
- `expletive`: 태그 내에 포함된 콘텐츠에 "삐" 소리를 냅니다.
- `telephone`: 값을 `2025551212`와 같이 7자리 또는 10자리 전화 번호로 해석합니다. 또한 `2025551212x345`와 같이 내선 전화 번호를 처리할 수 있습니다.

  Note

  현재 `telephone` 옵션은 영어 음성에만 사용할 수 있습니다.

Amazon Polly 서비스는 <say-as> 태그가 없어도 텍스트의 형식에 따라 올바르게 제공된 텍스트를 해석하려고 시도합니다. 예를 들어, 텍스트에 "202-555-1212"가 포함된 경우 Amazon Polly는 10자리 전화 번호로 해석한 다음 각 대시는 짧게 일시 중지하면서 각 개별 숫자는 따로따로 말합니다. 이 경우 `<say-as interpret-as="telephone">`은 사용하지 않아도 됩니다. 그러나 텍스트 "2025551212"를 제공하고 Amazon Polly가 전화 번호로 말하기를 원하는 경우 `<say-as interpret-as="telephone">`을 사용해야 합니다.

각 요소의 해석에 내재된 논리는 언어별로 다릅니다. 예를 들어, 미국 및 영국 영어는 전화 번호를 발음하는 방법이 다릅니다. 영국 영어는 동일한 숫자가 이어서 나오면 "double five" 또는 "triple four"처럼 그룹으로 묶어서 말합니다. 미국 음성과 영국 음성으로 다음의 예를 테스트하여 차이점을 확인할 수 있습니다.

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

분수

Amazon Polly은 `interpret-as="fraction"` 속성을 갖는 `say-as` 내의 값을 보통 분수로 해석합니다. 다음은 분수에 대한 구문입니다.

- 분수

  구문: *`cardinal number`*/*`cardinal number`* 예: 2/9.

  예를 들어 `<say-as interpret-as="fraction">2/9</say-as>`는 "9분의 2"로 발음됩니다.
- 음수가 아닌 혼수

  Syntax: *`cardinal number`*+*`cardinal number`*/*`cardinal number`* 예: 3 1/2.

  예를 들어 `<say-as interpret-as="fraction">3+1/2</say-as>`은 "삼 더하기 이분의 일"로 발음됩니다.

날짜

다음 값은 `interpret-as`가 `date`로 설정된 경우 날짜의 형식을 나타내는 데 사용할 수 있습니다.

- mdy: 월-일-년
- dmy: 일-월-년
- ymd: 년-월-일
- md: 월-일
- dm: 일-월
- ym: 년-월
- my: 월-년
- d: day.
- m: 월
- y: 년.
- yyyymmdd: 년-월-일. 이 형식이 사용되는 경우 날짜의 일부분에 물음표를 포함하여 제거할 수 있습니다. 예를 들어, `<say-as interpret-as="date">????0922</say-as>`는 "September 22nd"입니다.

<sub>

이 태그는 태그로 묶인 텍스트의 발음을 `alias` 속성의 발음으로 대체합니다. 이 태그는 일반적으로 두문자어나 약어에 대해 확장된 발음을 제공하는데 사용됩니다.

```
<speak>
     My favorite chemical element is <sub alias="mercury">Hg</sub>, it looks cool.
</speak>
```

<w>

`<say-as>`와 유사한 이 태그는 스피치의 단어 부분을 지정하여 단어 발음을 사용자 정의합니다. 태그에 `role` 속성을 포함하여 표시되는 스피치의 일부분을 지정합니다.

다음 값은 `role` 속성에 사용할 수 있습니다.

- `amazon:VB`: 단어를 동사(단순 현재)로 해석합니다.
- `amazon:VBD`: 단어를 과거 시제 또는 과거 분사로 해석합니다.
- `amazon:SENSE_1`: 있는 경우 단어의 기본이 아닌 의미를 사용합니다. 예를 들어, "bass"의 기본 의미는 화음에서 낮은 음이나 음역의 가장 낮은 부분입니다. 이는 민물고기를 의미할 때와는 다르게 발음되며, "bass"로 나와도 다르게 발음됩니다. `<w role="amazon:SENSE_1">bass</w>`를 사용하면 해당 오디오 텍스트를 기본이 아닌 발음(민물고기)으로 렌더링합니다.

"read"의 미국식 영어 발음은 사용 의도에 따라, 사용되는 방식에 따라 다음과 같이 달라집니다.

```
<speak>
     The present simple form of the word is pronounced <w role="amazon:VB">read</w>,
     where the past tense or past participle is pronounced <w role="amazon:VBD">read</w>.
</speak>
```

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

자연스럽게 들리는 음성에는 올바르게 말하는 단어와 숨소리가 모두 포함됩니다. 음성 합성에 숨소리를 추가하면 더 자연스럽게 들리는 음성을 만들 수 있습니다. `<amazon:breath>` 및 `<amazon:auto-breaths>` 태그는 숨소리를 제공합니다. 다음과 같은 옵션이 있습니다.

- 수동 모드: 사용자가 텍스트 내에 있는 숨소리의 위치, 길이, 볼륨을 설정합니다.

- 자동화 모드: Amazon Polly에서 숨소리를 음성 출력에 자동으로 삽입합니다.
- 혼합 모드: 사용자와 Amazon Polly가 모두 숨소리를 추가합니다.

수동 모드

수동 모드에서는 숨소리를 배치할 입력 텍스트에 <amazon:breath/> 태그를 지정합니다. duration 및 volume 속성을 각각 사용하여 숨소리의 길이와 볼륨을 사용자 지정할 수 있습니다.

- duration: 숨소리의 길이를 제어합니다. 유효한 값은 default, x-short, short, medium, long, x-long입니다. 기본 값은 medium입니다.
- volume: 숨소리가 얼마나 크게 들리는지를 제어합니다. 유효한 값은 default, x-soft, soft, medium, loud, x-loud입니다. 기본 값은 medium입니다.

> **Note**
>
> 각 속성 값의 정확한 길이와 볼륨은 사용되는 특정 Amazon Polly 음성에 따라 다릅니다.

기본값을 사용하여 숨소리를 설정하려면 속성 없이 <amazon:breath/>를 사용합니다.

예를 들어, 속성을 사용하여 숨소리의 기간과 볼륨을 중간으로 설정하려면 속성을 다음과 같이 설정합니다.

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

기본값을 사용하려면 태그만 사용합니다.

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

다음과 같이 구절 내에 개별 숨소리를 추가할 수 있습니다.

```
<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>
```

자동화 모드

자동화 모드에서는 <amazon:auto-breaths> 태그를 사용하여 Amazon Polly에서 적절한 간격으로 숨소리 잡음을 자동으로 생성하도록 지정합니다. 간격의 빈도, 볼륨 및 기간을 설정할 수 있습니다. 자동화된 숨소리를 적용할 텍스트의 시작 부분에 </amazon:auto-breaths> 태그를 지정하고 끝 부분에 닫는 태그를 지정합니다.

> **Note**
>
> 수동 모드 태그인 <amazon:breath/>와는 달리, <amazon:auto-breaths> 태그에는 닫는 태그(</amazon:auto-breaths>)가 필요합니다.

다음과 같은 선택적 속성을 <amazon:auto-breaths> 태그와 함께 사용할 수 있습니다.

- volume: 숨소리가 얼마나 크게 들리는지를 제어합니다. 유효한 값은 default, x-soft, soft, medium, loud, x-loud입니다. 기본 값은 medium입니다.
- frequency: 텍스트에서 숨소리가 얼마나 자주 들리는지를 제어합니다. 유효한 값은 default, x-low, low, medium, high, x-high입니다. 기본 값은 medium입니다.
- duration: 숨소리의 길이를 제어합니다. 유효한 값은 default, x-short, short, medium, long, x-long입니다. 기본 값은 medium입니다.

기본적으로 숨소리의 빈도는 입력 텍스트에 따라 다릅니다. 하지만 주로 쉼표와 마침표 뒤에 숨소리가 들립니다.

다음 예제에서는 <amazon:auto-breaths> 태그를 사용하는 방법을 보여줍니다. 콘텐츠에 어떤 옵션을 사용할지 결정하려면 적용 가능한 예제를 Amazon Polly 콘솔에 복사하고 차이를 들어 봅니다.

- 선택적 파라미터 없이 자동화 모드를 사용합니다.

```
<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>
```

- 볼륨 제어와 함께 자동화 모드를 사용합니다. 비지정 파라미터(duration 및 frequency)는 기본값(medium)으로 설정됩니다.

```
<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>
```

- 빈도 제어와 함께 자동화 모드를 사용합니다. 비지정 파라미터(duration 및 volume)는 기본값(medium)으로 설정됩니다.

```
<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>
```

- 다중 파라미터와 함께 자동화 모드를 사용합니다. 비지정 `Duration` 파라미터의 경우 Amazon Polly에서 기본값(`medium`)을 사용합니다.

```
<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>
```

`amazon:auto-breaths` 태그의 효과를 향상하려면 태그를 중첩할 수 있습니다.

예:

```
<speak>
     <amazon:auto-breaths volume="soft" frequency="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:auto-breaths frequency="high">Amazon Polly is a
 text-to-speech
     service, that uses advanced deep learning technologies to synthesize speech that
 sounds like a human
     voice. </amazon:auto-breaths>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>
```

숨소리 태그를 중첩하면 새로운 설정을 지정하지 않는 한 Amazon Polly는 바깥쪽 태그에 대한 설정을 안쪽 태그에 사용합니다. 앞의 예제에서 바깥쪽 태그의 볼륨은 `soft`로 설정됩니다. 안쪽 태그는 볼륨을 지정하지 않기 때문에 바깥쪽 태그의 `soft` 설정을 사용합니다.

> **Note**
>
> 각 중첩 `<amazon:auto-breaths>` 태그는 `</amazon:auto-breaths>` 태그를 사용하여 닫아야 합니다.

혼합 모드

가장 유연하게 조정하려면 자동화 모드 태그를 수동 모드 태그와 조합할 수 있습니다. 혼합모드에서는 Amazon Polly가 전체 텍스트 구절에서 숨소리를 자동화할 수 있으며 사용자가 특정 위치에 숨소리를 추가할 수 있습니다.

예:

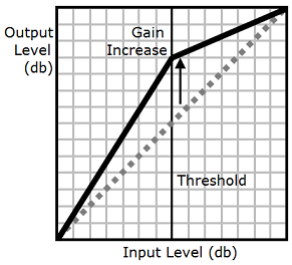
```
<speak>
     <amazon:auto-breaths> Breathing (or respiration) is the process of moving air into and
 out of the
```

```
    lungs to facilitate gas exchange between the internal and external environments,
 mostly by bringing in
    oxygen <amazon:breath/> and flushing out carbon dioxide. </amazon:auto-breaths>
</speak>
```

<amazon:effect name="drc">

오디오 파일에 사용된 텍스트, 언어, 음성에 따라 소리 크기는 작은 소리부터 큰 소리까지 다를 수 있습니다. 자동차 소리 같은 주변 소리로 인해 작은 소리가 가려져서 오디오 트랙을 정확히 듣기 어려울 수 있습니다. 오디오 파일의 특정 사운드 볼륨을 높이려면 동적 범위 압축(drc) 태그를 사용하십시오.

drc 태그는 오디오에 중간 "크기" 경계를 설정하고, 해당 경계 주변의 사운드 볼륨(게인)을 높입니다. 즉, 경계값에 가장 가까우면 최대 게인이 적용되고, 경계값에서 멀수록 게인이 감소합니다.



따라서 소음이 많은 환경에서 중간 크기 사운드를 쉽게 들을 수 있으며 오디오 파일을 전체적으로 듣기 쉽게 만듭니다.

drc 태그는 부울 파라미터입니다(참 또는 거짓). 이 태그는 <amazon:effect name="drc"> 구문을 사용하며 </amazon:effect>로 닫습니다.

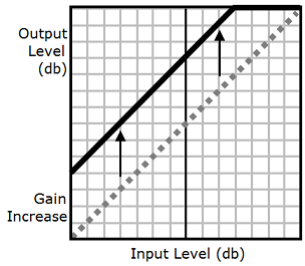
drc 태그는 Amazon Polly에서 지원하는 모든 음성이나 언어에 사용할 수 있습니다. 녹음 전체 섹션에 적용하거나 몇 단에만 적용할 수 있습니다. 예:

```
<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>
```

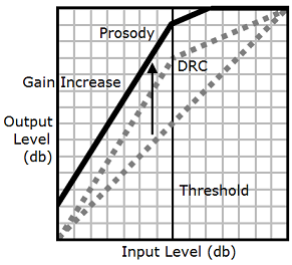
> **Note**
>
> amazon:effect 구문에 "drc"를 사용하면 대소문자를 구별합니다.

prosody volume 태그에 drc 사용

다음 그림에서 보는 것처럼, prosody volume 태그는 전체 오디오 파일의 볼륨을 원래 수준(점선)에서 조정된 수준(실선)으로 균일하게 올립니다. 파일의 특정 부분 볼륨을 더 높이려면 drc 태그와 prosody volume 태그를 함께 사용하십시오. 조합 태그는 prosody volume 태그 설정에는 영향을 주지 않습니다.



drc 태그와 prosody volume 태그를 함께 사용하면 Amazon Polly는 drc 태그를 먼저 적용하여 중간 크기 사운드(경계값에 가까운 사운드)를 강화합니다. 그런 다음 prosody volume 태그를 적용하여 전체 오디오 트랙의 볼륨을 균일하게 더 높입니다.



태그를 함께 사용하려면 한 태그를 다른 태그 내에 넣으십시오. 예:

```
<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>
```

이 텍스트에서 prosody volume 태그는 전체 구절의 볼륨을 "크게" 높입니다. drc 태그는 둘째 문장에서 중간 크기 값의 볼륨을 강화합니다.

> Note
>
> drc 태그와 prosody volume 태그를 함께 사용할 경우 중첩 태그에 대한 표준 XML 규칙을 사용하십시오.

<amazon:effect phonation="soft">

이 태그는 입력 텍스트를 정상 스피치보다 더 부드러운 음성으로 말해야 함을 의미합니다. 이 태그는 Amazon Polly Text-to-Speech 포트폴리오에 있는 모든 음성과 함께 사용할 수 있습니다. 이 태그는 <amazon:effect phonation="soft"> 구문을 사용하며 </amazon:effect>로 닫습니다.

예를 들어 Matthew 음성을 사용하는 경우 다음과 같습니다.

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

이 경우 합성된 음성의 첫 번째 부분은 정상적인 음성으로 말하지만 phonation 태그를 사용하는 부분은 더 부드럽게 말합니다.

<amazon:effect vocal-tract-length>

음색은 음의 고저가 같을 때도 음성 간의 차이를 알 수 있는 특징입니다. 음색을 내는 가장 중요한 생리적 특성 중 하나는 성대의 길이이며, 성대주름에서 입술 가장자리까지의 공기 통로입니다.

Amazon Polly에서 음성의 음색을 조절하려면 vocal-tract-length 태그를 사용합니다. 이 태그는 말하는 사람의 성도 길이를 변경하여 다르게 들리게 합니다. vocal-tract-length를 늘리면 말하는 사람의 음성이 더 커집니다. 반대로 줄이면 음성이 작아집니다. 이 태그는 Amazon Polly Text-to-Speech 포트폴리오에 있는 모든 음성에 사용할 수 있습니다.

음색을 변경하려면 다음 값을 사용합니다.

- `+n%` 또는 `-n%`: 현재 음성의 현재 음색 레벨을 상대적 백분율로 변경하여 성도 길이를 조정합니다. 예를 들어 +4% 또는 -2% 등으로 지정할 수 있습니다. 유효한 값은 +100%부터 -50%까지입니다. 이 범위를 벗어나는 값은 잘립니다. 예를 들어 +111%는 +100%와 동일하게 적용되고 -60%은 -50%와 동일하게 적용됩니다.
- `n%`: 현재 음성 성도 길이의 절대 백분율로 성도 길이를 변경합니다. 예를 들어 110% 또는 75% 등으로 지정합니다. 절대값 110%는 상대값 +10%와 동일합니다. 절대값 100%는 현재 음성의 기본값과 동일합니다.

다음 예는 성도 길이를 변경하여 음색을 변경하는 방법을 보여줍니다.

```
<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>
```

여러 태그 결합

`vocal-tract-length` 태그와 Amazon Polly에서 지원하는 다른 SSML 태그를 결합할 수 있습니다. 음색(성도 길이)와 음의 고저는 서로 밀접하기 연결되기 때문에 `vocal-tract-length` 태그와 `<prosody pitch>` 태그를 함께 사용하여 최적의 결과를 얻을 수 있습니다. 가장 사실적인 음성을 만들어 내기 위해 두 태그의 변경 백분율을 다르게 사용할 수 있습니다. 다양한 조합을 사용해 보면서 원하는 결과를 얻으십시오.

다음 예제는 태그를 결합하는 방법을 보여줍니다.

```
<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>
```

<amazon:effect name="whispered">

이 태그는 입력 텍스트를 일반 스피치보다 속삭이는 음성으로 말해야 함을 나타냅니다. 이 태그는 Amazon Polly Text-to-Speech 포트폴리오에 있는 모든 음성과 함께 사용할 수 있습니다. 이 태그는 `<amazon:effect name="whispered">` 구문을 사용하며 `</amazon:effect>`로 닫습니다.

예:

```
<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>
```

이 경우 캐릭터가 말하는 합성된 스피치는 속삭이는 음성이 되고, "she said" 구는 선택한 Amazon Polly 음성의 합성된 일반 스피치가 됩니다.

원하는 효과에 따라 운율 속도를 최대 10%까지 느리게 하여 "속삭임" 효과를 향상시킬 수 있습니다.

예:

```
<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>
```

속삭이는 음성에 대한 스피치 마크를 생성하는 경우 오디오 스트림은 속삭이는 음성도 포함하여 스피치 마크가 오디오 스트림과 일치하도록 해야 합니다.

Amazon Polly에서 사용하는 발음 표

다음 표에서는 미국식 영어의 Amazon Polly SSML 태그에 대한 음성 발음(음소)을 제공합니다.

IPA	X-SAMPA	설명	예
자음			
b	b	유성 양순 파열음	bed
d	d	유성 치경 파열음	dig
ʤ	dZ	유성 후치경 파찰음	jump
ð	D	유성 치 마찰음	then
f	f	무성 순치 마찰음	five
g	g	유성 연구개 파열음	game
h	h	무성 성문 마찰음	house
j	j	경구개 접근음	yes
k	k	무성 연구개 파열음	cat
l	l	치경 설측 접근음	lay
m	m	양순 비음	mouse
n	n	치경 비음	nap
ŋ	N	연구개 비음	thing
p	p	무성 양순 파열음	speak
ɹ	r\	치경 접근음	red
s	s	무성 치경 마찰음	seem
ʃ	S	무성 후치경 마찰음	ship
t	t	무성 치경 파열음	trap
ʧ	tS	무성 후치경 파찰음	chart
Θ	T	무성 치 마찰음	thin

IPA	X-SAMPA	설명	예
v	v	유성 순치 마찰음	vest
w	w	양순 연구개 접근음	west
z	z	유성 치경 마찰음	zero
ʒ	Z	유성 후치경 마찰음	vision
모음			
ə	@	중설 중모음	arena
ɚ	@'	중설 치경 접근 수반음	reader
æ	{	전설 비원순 근저모음	trap
aɪ	aI	이중 모음	price
aʊ	aU	이중 모음	mouth
ɑ	A	후설 비원순 저장모음	father
eɪ	eI	이중 모음	face
ɝ	3'	중설 비원순 중저 치경 접근 수반음	nurse
ɛ	E	전설 비원순 중저모음	dress
i:	i	전설 비원순 고장모음	fleece
ɪ	I	근전설 비원순 근고모음	kit
oʊ	oU	이중 모음	goat
ɔ	O	후설 원순 중저장모음	thought
ɔɪ	OI	이중 모음	choice
u	u	후설 원순 고장모음	goose
ʊ	U	근후설 원순 근고모음	foot
ʌ	V	후설 평순 중저모음	strut
기타 기호			
'	"	제1강세	Alabama
,	%	제2강세	Alabama
.	.	음절 경계	A.la.ba.ma

어휘 관리

발음 어휘를 사용하여 단어의 발음을 사용자 지정할 수 있습니다. Amazon Polly에서는 AWS 리전에 어휘를 저장하는 데 사용할 수 있는 API 작업을 제공합니다. 따라서 그러한 어휘는 특정 리전과 관련이 있습니다. `SynthesizeSpeech` 작업을 사용하여 텍스트를 합성할 때 해당 리전에서 어휘를 한 개 이상 사용할 수 있습니다. 그래서 합성이 시작되기 전에 지정된 어휘가 입력 텍스트에 적용됩니다. 자세한 내용은 SynthesizeSpeech (p. 148) 섹션을 참조하십시오.

> Note
>
> 이 어휘는 PLS(Pronunciation Lexicon Specification) W3C 권장 사항을 준수해야 합니다. 자세한 내용은 W3C 웹 사이트에서 Pronunciation Lexicon Specification (PLS) Version 1.0을 참조하십시오.

다음은 스피치 합성 엔진과 어휘를 사용하는 방법의 예입니다.

- 공통되는 단어는 "g3t sm4rt"(get smart)와 같이 문자를 대신하는 숫자 스타일이 사용되기도 합니다. 사람은 이 단어를 제대로 읽을 수 있지만 TTS(Text-to-Speech) 엔진은 철자에 따라 정확히 이름을 발음하여 문자 그대로 텍스트를 읽기 때문에 Amazon Polly를 사용하여 합성된 스피치를 사용자 지정하기 위해 어휘를 이용할 수 있습니다. 이 예에서는 어휘에서 "g3t sm4rt" 단어의 별칭(get smart)을 지정할 수 있습니다.
- W3C와 같이 두문자어를 텍스트에 포함할 수도 있습니다. 어휘를 사용하여 W3C라는 단어의 별칭을 정의할 수 있습니다. 그러면 확장된 전체 형태(World Wide Web Consortium)로 단어가 판독됩니다.

어휘를 사용하면 선택한 언어에 드물게 사용되는 단어를 Amazon Polly가 발음하는 방법을 더 자세히 제어할 수 있습니다. 예를 들어, 음성 기호를 사용하여 발음을 지정할 수 있습니다. 자세한 내용은 W3C 웹 사이트에서 Pronunciation Lexicon Specification (PLS) Version 1.0을 참조하십시오.

항목

여러 어휘 적용

텍스트에 어휘를 5개까지 적용할 수 있습니다. 텍스트에 적용하는 2개 이상의 어휘에 같은 문자소가 나타나면 적용 순서에 따라 결과 스피치가 달라질 수 있습니다. 예를 들어, "Hello, my name is Bob."이라는 텍스트와 어휘소 2개가 다른 어휘에 있으면 둘 다 문자소 `Bob`을 사용합니다.

LexA

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

LexB

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

LexA 다음에 LexB가 오는 순서로 어휘가 나열되면 합성된 스피치는 "Hello, my name is Robert."입니다. LexB 다음에 LexA가 오는 순서로 나열되면 합성된 스피치는 "Hello, my name is Bobby."입니다.

Example – LexB 앞에 LexA 적용

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

스피치 출력: "Hello, my name is Robert."

Example – LexA 앞에 LexB 적용

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

스피치 출력: "Hello, my name is Bobby."

Amazon Polly 콘솔을 사용하여 어휘를 적용하는 데 대한 자세한 내용은 콘솔을 사용하여 어휘 적용(스피치 합성) (p. 86) 항목을 참조하십시오.

Amazon Polly 콘솔을 사용하여 어휘 관리

Amazon Polly 콘솔을 사용하여 어휘를 업로드, 다운로드, 적용, 필터링 및 삭제할 수 있습니다. 다음 절차에서는 이 프로세스를 각각 보여줍니다.

콘솔을 사용하여 어휘 업로드

발음 어휘를 사용하려면 먼저 업로드해야 합니다. 콘솔에서 어휘를 업로드할 수 있는 위치는 [Text-to-Speech] 탭과 [Lexicons] 탭 두 곳입니다.

다음 프로세스에서는 선택한 언어에 드물게 사용되는 단어와 문구의 발음 방식을 사용자 지정하는 데 사용할 어휘를 추가하는 방법을 설명합니다.

[Lexicons] 탭에서 어휘를 추가하려면

1. AWS Management Console에 로그인한 후 https://console.aws.amazon.com/polly/에서 Amazon Polly 콘솔을 엽니다.
2. [Lexicons] 탭을 선택합니다.
3. Upload를 선택합니다.
4. 업로드할 어휘를 찾아봅니다. .pls 및 .xml 확장명을 사용하는 PLS 파일만 사용할 수 있습니다.
5. [Open]을 선택합니다. .pls 파일이건 .xml 파일이건 같은 이름의 어휘가 이미 있으면 업로드한 어휘가 기존 어휘를 덮어씁니다.

[Text-to-Speech] 탭에서 어휘를 추가하려면

1. AWS Management Console에 로그인한 후 https://console.aws.amazon.com/polly/에서 Amazon Polly 콘솔을 엽니다.

2. [Text-to-Speech] 탭을 선택합니다.
3. [Customize pronunciation of words or phrases using lexicons]를 선택하고 [Upload lexicon]을 선택합니다.
4. 업로드할 어휘를 찾아봅니다. .pls 및 .xml 확장명을 사용하는 PLS 파일만 사용할 수 있습니다.
5. [Open]을 선택합니다. .pls 파일이건 .xml 파일이건 같은 이름의 어휘가 이미 있으면 업로드한 어휘가 기존 어휘를 덮어씁니다.

콘솔을 사용하여 어휘 적용(스피치 합성)

다음 절차에서는 `W3c.pls` 어휘를 적용하여 "W3C" 대신 "World Wide Web Consortium"을 사용하도록 입력 텍스트에 어휘를 적용하는 방법을 보여줍니다. 텍스트에 여러 어휘를 적용하면 위에서 아래로 적용되며 첫 번째 일치 항목이 나중 일치 항목보다 우선합니다. 어휘에 지정된 언어가 선택한 언어와 같아야 어휘가 텍스트에 적용됩니다.

일반 텍스트 또는 SSML 입력에 어휘를 적용할 수 있습니다.

Example – W3C.pls 어휘 적용

이 연습에 필요한 어휘를 생성하려면 PutLexicon 작업 사용 (p. 88) 항목을 참조하십시오. 일반 텍스트 편집기를 사용하여 항목 맨 위에 표시된 W3C.pls 어휘를 생성하십시오. 이 파일을 저장한 위치를 기억해 두십시오.

입력에 W3C.pls 어휘를 적용하려면

이 예에서는 "W3C"를 "World Wide Web Consortium"으로 대체할 어휘를 소개합니다. 이 연습의 결과를 미국 영어 및 다른 언어의 Amazon Polly 콘솔과 함께 SSML 사용 (p. 62) 결과와 모두 비교하십시오.

1. AWS Management Console에 로그인한 후 https://console.aws.amazon.com/polly/에서 Amazon Polly 콘솔을 엽니다.
2. 다음 중 하나를 수행하십시오.
   - [Plain text] 탭을 선택한 후 이 텍스트를 입력하거나 텍스트 입력란에 붙여 넣습니다.

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

   - [SSML] 탭을 선택한 후 이 텍스트를 입력하거나 텍스트 입력란에 붙여 넣습니다.

     ```
     <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. [Choose a language and region] 목록에서 [English US]를 선택한 후 이 텍스트에 사용할 음성을 선택합니다.
4. [Customize pronunciation of words or phrases using lexicons]를 선택합니다.
5. 어휘 목록에서 [`W3C (English, US)`]를 선택합니다.

   [`W3C (English, US)`] 어휘가 목록에 없으면 [Upload lexicon]을 선택하여 업로드한 후 목록에서 선택합니다. 이 어휘를 생성하려면 PutLexicon 작업 사용 (p. 88) 단원을 참조하십시오.
6. 스피치를 즉시 들으려면 [Listen to speech]를 선택합니다.
7. 스피치를 파일에 저장하려면

   a. [Save speech to MP3]를 선택합니다.

b. 다른 파일 형식으로 변경하려면 [Change file format]을 선택하여 원하는 파일 형식을 선택한 다음 [Change]를 선택합니다.

이전 단계를 반복하되 다른 언어를 선택하고 출력의 차이를 확인하십시오.

콘솔을 사용하여 어휘 목록 필터링

다음 절차에서는 선택한 언어의 어휘만 표시되도록 어휘 목록을 필터링하는 방법을 설명합니다.

언어별로 나열되도록 어휘를 필터링하려면

1. AWS Management Console에 로그인한 후 https://console.aws.amazon.com/polly/에서 Amazon Polly 콘솔을 엽니다.
2. [Lexicons] 탭을 선택합니다.
3. [Filter]를 선택합니다.
4. 언어 목록에서 필터링할 언어를 선택합니다.

   선택한 언어의 어휘만 목록에 표시됩니다.

콘솔을 사용하여 어휘 다운로드

다음 프로세스에서는 어휘를 1개 이상 다운로드하는 방법을 설명합니다. 파일의 어휘 항목을 추가, 제거 또는 수정하고 다시 업로드하여 어휘를 최신 상태로 유지할 수 있습니다.

어휘를 1개 이상 다운로드하려면

1. AWS Management Console에 로그인한 후 https://console.aws.amazon.com/polly/에서 Amazon Polly 콘솔을 엽니다.
2. [Lexicons] 탭을 선택합니다.
3. 다운로드할 어휘를 선택합니다.

   a. 어휘 1개를 다운로드하려면 목록에서 어휘 이름을 선택합니다.
   b. 여러 어휘를 압축된 아카이브 파일 하나로 다운로드하려면 목록에서 다운로드할 각 항목 옆의 확인란을 선택합니다.
4. [Download]를 선택합니다.
5. 어휘를 다운로드할 폴더를 엽니다.
6. Save를 선택합니다.

콘솔을 사용하여 어휘 삭제

어휘를 삭제하려면

다음 프로세스에서는 어휘를 삭제하는 방법을 설명합니다. 어휘를 삭제한 후에는 다시 추가해야 다시 사용할 수 있습니다. 각 어휘 옆의 확인란을 선택하여 어휘 1개 이상을 동시에 삭제할 수 있습니다.

1. AWS Management Console에 로그인한 후 https://console.aws.amazon.com/polly/에서 Amazon Polly 콘솔을 엽니다.
2. [Lexicons] 탭을 선택합니다.
3. 목록에서 삭제할 어휘를 1개 이상 선택합니다.
4. [Delete]를 선택합니다.
5. [Delete]를 선택하여 리전에서 어휘를 제거하거나 [Cancel]을 선택하여 그대로 유지합니다.

AWS CLI를 사용하여 어휘 관리

다음 항목에서는 발음 어휘를 관리하는 데 필요한 AWS CLI 명령을 다룹니다.

항목

PutLexicon 작업 사용

Amazon Polly를 통해 PutLexicon (p. 146)을 사용하여 특정 AWS 리전의 계정에 발음 언어를 저장할 수 있습니다. 그런 다음 서비스에서 텍스트 합성을 시작하기 전에 적용할 SynthesizeSpeech (p. 148) 요청에 저장된 어휘를 1개 이상 지정할 수 있습니다. 자세한 내용은 어휘 관리 (p. 84) 섹션을 참조하십시오.

이 단원에서는 예제 어휘와 어휘 저장 및 테스트를 위한 단계별 지침을 제공합니다.

> **Note**
>
> 이 어휘는 PLS(Pronunciation Lexicon Specification) W3C 권장 사항을 준수해야 합니다. 자세한 내용은 W3C 웹 사이트에서 Pronunciation Lexicon Specification (PLS) Version 1.0을 참조하십시오.

예제 1: 어휘소가 1개인 어휘

다음의 W3C PLS 준수 어휘를 고려하십시오.

```
<?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>
```

다음을 참조하십시오.

- `<lexicon>` 요소에 지정된 속성 2개:
  - `xml:lang` 속성은 어휘가 적용되는 언어 코드 `en-US`를 지정합니다. `SynthesizeSpeech` 호출에 지정하는 음성의 언어 코드가 동일할 경우(en-US) Amazon Polly에서 이 예제 어휘를 사용할 수 있습니다.

    > **Note**
    >
    > `DescribeVoices` 작업을 사용하여 음성과 연결된 언어 코드를 찾을 수 있습니다.

  - `alphabet` 속성은 `IPA`를 지정하므로 IPA(International Phonetic Alphabet) 알파벳이 발음에 사용됩니다. IPA는 표음 문자 중 하나입니다. 또한 Amazon Polly는 X-SAMPA(Extended Speech Assessment Methods Phonetic Alphabet)를 지원합니다.

- <lexeme> 요소는 <grapheme>(단어의 텍스트 표현)과 <alias> 사이의 매핑을 설명합니다.

이 어휘를 테스트하려면 다음을 수행하십시오.

1. 어휘를 example.pls로 저장합니다.
2. put-lexicon AWS CLI 명령을 시행하여 us-east-2 리전에 어휘를 저장합니다(w3c로 이름 지정).

```
aws polly put-lexicon \
--name w3c \
--content file://example.pls
```

3. synthesize-speech 명령을 실행하여 샘플 텍스트를 오디오 스트림으로 합성하고(speech.mp3) 선택적 lexicon-name 파라미터를 지정합니다.

```
aws polly synthesize-speech \
--text 'W3C is a Consortium' \
--voice-id Joanna \
--output-format mp3 \
--lexicon-names="w3c" \
speech.mp3
```

4. 결과 speech.mp3를 재생하고 텍스트의 W3C라는 단어가 World Wide Web Consortium으로 바뀌는지 확인합니다.

앞에 나온 예제 어휘에 별칭이 사용되며 어휘에 사용된 IPA 알파벳은 사용되지 않습니다. 다음 어휘는 APA 알파벳과 <phoneme> 요소를 사용하여 음성 발음을 지정합니다.

```
<?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>
```

이 어휘를 테스트하려면 같은 단계를 따르십시오. "pecan"이라는 단어가 있는 입력 텍스트(예: "Pecan pie is delicious")를 지정합니다.

예제 2: 어휘소가 여러 개인 어휘

이 예제에서는 어휘에 지정된 어휘소가 합성할 입력 텍스트에만 적용됩니다. 다음 어휘를 고려하십시오.

```
<?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>
```

어휘가 어휘소 3개를 지정하고, 그 중 2개가 다음과 같이 문자소 W3C의 별칭을 정의합니다.

- 첫 번째 <lexeme> 요소가 별칭(World Wide Web Consortium)을 정의합니다.
- 두 번째 <lexeme>이 대체 별칭(WWW Consortium)을 정의합니다.

Amazon Polly가 첫 번째 대체 항목을 어휘의 지정된 문자소에 사용합니다.

세 번째 <lexeme>이 Consortium이라는 단어의 대체 항목(Community)을 정의합니다.

먼저 이 어휘를 테스트하겠습니다. 다음 샘플 텍스트를 오디오 파일(speech.mp3)로 합성하고 SynthesizeSpeech 호출에 어휘를 지정합니다.

```
The W3C is a Consortium
```

SynthesizeSpeech가 먼저 다음과 같이 어휘를 적용합니다.

- 첫 번째 어휘소에 따라 단어 W3C가 World Wide Web Consortium으로 바뀝니다. 바뀐 텍스트가 다음과 같이 표시됩니다.

```
The World Wide Web Consortium is a Consortium
```

- 세 번째 어휘소에 정의된 별칭은 원래 텍스트에 포함된 단어 Consortium에만 적용되므로 다음과 같은 텍스트가 생성됩니다.

```
The World Wide Web Consortium is a Community.
```

다음과 같이 AWS CLI를 사용하여 이 항목을 테스트할 수 있습니다.

1. 어휘를 example.pls로 저장합니다.
2. put-lexicon 명령을 실행하여 이름이 w3c인 어휘를 us-east-2 리전에 저장합니다.

```
aws polly put-lexicon \
--name w3c \
--content file://example.pls
```

3. list-lexicons 명령을 실행하여 w3c 어휘가 반환된 어휘 목록에 있는지 확인합니다.

```
aws polly list-lexicons
```

4. synthesize-speech 명령을 실행하여 샘플 텍스트를 오디오 파일로 합성하고(speech.mp3) 선택적 lexicon-name 파라미터를 지정합니다.

```
aws polly synthesize-speech \
```

```
--text 'W3C is a Consortium' \
--voice-id Joanna \
--output-format mp3 \
--lexicon-names="w3c" \
speech.mp3
```

5. 결과 `speech.mp3` 파일을 재생하여 합성된 스피치에 테스트 변경 사항이 반영되는지 확인합니다.

예제 3: 여러 어휘 지정

`SynthesizeSpeech` 호출에서 여러 어휘를 지정할 수 있습니다. 이 경우에는 지정된 첫 번째 어휘(왼쪽에서 오른쪽)가 이전의 어휘를 재정의합니다.

다음 어휘 2개를 고려하십시오. 각 어휘는 동일한 문자소 W3C의 여러 가지 별칭을 설명합니다.

- 어휘 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>
```

- 어휘 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>
```

이 어휘를 `w3c` 및 `w3cAlternate`로 각각 저장하겠습니다. `w3c` 호출에서 `w3cAlternate` 다음에 `SynthesizeSpeech`가 오는 순서로 어휘를 지정할 경우 첫 번째 어휘에 정의된 W3C의 별칭이 두 번째 항목보다 우선 적용됩니다. 어휘를 테스트하려면 다음을 수행하십시오.

1. 로컬에 있는 `w3c.pls` 및 `w3cAlternate.pls` 파일에 어휘를 저장합니다.
2. `put-lexicon` AWS CLI 명령을 사용하여 이 어휘를 업로드합니다.

   - `w3c.pls` 어휘를 업로드하고 `w3c`로 저장합니다.

   ```
   aws polly put-lexicon \
   --name w3c \
   --content file://w3c.pls
   ```

- `w3cAlternate.pls` 어휘를 서비스에 `w3cAlternate`로 업로드합니다.

```
aws polly put-lexicon \
--name w3cAlternate \
--content file://w3cAlternate.pls
```

3. `synthesize-speech` 명령을 실행하여 샘플 텍스트를 오디오 스트림으로 합성하고(`speech.mp3`) `lexicon-name` 파라미터를 사용하여 두 어휘를 모두 지정합니다.

```
aws polly synthesize-speech \
--text 'PLS is a W3C recommendation' \
--voice-id Joanna \
--output-format mp3 \
--lexicon-names '["w3c","w3cAlternative"]' \
speech.mp3
```

4. 결과 `speech.mp3`를 테스트합니다. 다음과 같이 읽힙니다.

```
PLS is a World Wide Web Consortium recommendation
```

PutLexicon API용 추가 코드 예제

- Java 샘플: PutLexicon (p. 98)
- Python(Boto3) 예제: PutLexicon (p. 100)

GetLexicon 작업 사용

Amazon Polly는 GetLexicon (p. 142) API 작업을 제공하여 특정 리전의 계정에 저장한 발음 어휘 내용을 검색합니다.

다음 `get-lexicon` AWS CLI 명령은 `example` 어휘 내용을 검색합니다.

```
aws polly get-lexicon \
--name example
```

계정에 저장된 어휘가 아직 없을 경우 `PutLexicon` 작업을 사용하여 어휘를 저장할 수 있습니다. 자세한 내용은 PutLexicon 작업 사용 (p. 88) 섹션을 참조하십시오.

다음은 응답 예입니다. 응답은 어휘 내용 외에도 어휘가 적용되는 언어, 어휘에 정의된 어휘소 개수, 리소스의 ARN(Amazon 리소스 이름) 및 어휘 크기(바이트 단위) 등의 메타데이터를 반환합니다. `LastModified` 값은 Unix 타임스탬프입니다.

```
{
    "Lexicon": {
        "Content": "lexicon content in plain text PLS format",
        "Name": "example"
    },
    "LexiconAttributes": {
        "LanguageCode": "en-US",
        "LastModified": 1474222543.989,
        "Alphabet": "ipa",
        "LexemesCount": 1,
        "LexiconArn": "arn:aws:polly:us-east-2:account-id:lexicon/example",
        "Size": 495
    }
```

```
}
```

GetLexicon API용 추가 코드 예제

- Java 샘플: GetLexicon (p. 97)
- Python(Boto3) 예제: GetLexicon (p. 101)

ListLexicons 작업 사용

Amazon Polly에서는 특정 AWS 리전의 계정에서 발음 어휘 목록을 가져오는 데 사용할 수 있는 ListLexicons (p. 144) API 작업을 제공합니다. 다음 AWS CLI 호출은 us-east-2 리전의 계정에 있는 어휘를 나열합니다.

```
aws polly list-lexicons
```

다음은 `w3c` 및 `tomato`라는 어휘 2개를 표시하는 예제 응답입니다. 각 어휘마다 응답은 어휘가 적용되는 언어, 어휘에 정의된 어휘소 개수, 바이트 단위의 크기 등의 메타데이터를 반환합니다. 언어 코드는 언어와 어휘에 정의된 어휘소가 적용되는 위치를 설명합니다.

```
{
    "Lexicons": [
        {
            "Attributes": {
                "LanguageCode": "en-US",
                "LastModified": 1474222543.989,
                "Alphabet": "ipa",
                "LexemesCount": 1,
                "LexiconArn": "arn:aws:polly:aws-region:account-id:lexicon/w3c",
                "Size": 495
            },
            "Name": "w3c"
        },
        {
            "Attributes": {
                "LanguageCode": "en-US",
                "LastModified": 1473099290.858,
                "Alphabet": "ipa",
                "LexemesCount": 1,
                "LexiconArn": "arn:aws:polly:aws-region:account-id:lexicon/tomato",
                "Size": 645
            },
            "Name": "tomato"
        }
    ]
}
```

ListLexicon API용 추가 코드 예제

- Java 샘플: ListLexicons (p. 97)
- Python(Boto3) 예제: ListLexicon (p. 102)

DeleteLexicon 작업 사용

Amazon Polly에서는 DeleteLexicon (p. 138) API 작업을 사용하여 계정의 특정 AWS 리전에서 발음 어휘를 삭제합니다. 다음 AWS CLI는 지정된 어휘를 삭제합니다.

다음은 Unix, Linux, macOS용 형식으로 지정된 AWS CLI 예제입니다. Windows의 경우 각 줄 끝에 있는 백슬래시(\) Unix 연속 문자를 캐럿(^)으로 바꿉니다. 입력 텍스트는 큰 따옴표(")로 감싸고 내부 태그에는 작은 따옴표(')를 사용합니다.

```
aws polly delete-lexicon \
--name example
```

DeleteLexicon API용 추가 코드 예제

- Java 샘플: DeleteLexicon (p. 95)
- Python(Boto3) 예제: DeleteLexicon (p. 103)

코드 및 애플리케이션 예제

이 단원에서는 Amazon Polly 탐색에 사용할 수 있는 코드 샘플과 예제 애플리케이션을 제공합니다.

항목

샘플 코드 항목에서는 프로그래밍 언어별로 분류되고 여러 Amazon Polly 기능에 대한 예제로 구분된 코드 조각을 제공합니다. 예제 애플리케이션 항목에는 Amazon Polly를 탐색하는 데 독립적으로 사용할 수 있는 프로그래밍 언어별로 분류된 애플리케이션이 나와 있습니다.

이 예제를 사용하려면 먼저 Amazon Polly: 사용 방법 (p. 3) 항목을 읽고 Amazon Polly로 시작하기 (p. 6)에 설명된 단계를 따르는 것이 좋습니다.

샘플 코드

이 항목에서는 Amazon Polly를 탐색하는 데 사용할 수 있는 다양한 기능을 위한 코드 샘플을 제공합니다.

프로그래밍 언어별 샘플 코드

Java 샘플

다음 코드 샘플은 Java 기반 애플리케이션을 사용하여 Amazon Polly의 다양한 작업을 수행하는 방법을 보여줍니다. 이러한 샘플은 전체 예제 중 일부이며 AWS SDK for Java를 사용하는 Java 애플리케이션에 포함될 수 있습니다.

코드 조각

DeleteLexicon

다음 Java 코드 샘플은 Java 기반 애플리케이션을 사용하여 AWS 리전에 저장된 특정 어휘를 삭제하는 방법을 보여줍니다. 삭제된 어휘는 음성 합성에서 사용할 수 없으며, `GetLexicon` `ListLexicon` API를 사용하여 검색할 수도 없습니다.

이 작업에 대한 자세한 내용은 `DeleteLexicon` API에 대한 참조 자료를 참조하십시오.

```
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

다음 Java 코드 샘플은 Java 기반 애플리케이션을 사용하여 음성 합성 요청 시 사용할 수 있는 음성 목록을 생성하는 방법을 보여줍니다. 필요한 경우, 언어 코드를 지정하여 사용 가능한 음성을 필터링할 수도 있습니다. 예를 들어 en-US를 지정하면 모든 미국 영어 음성 목록이 반환됩니다.

이 작업에 대한 자세한 내용은 `DescribeVoices` API에 대한 참조 자료를 참조하십시오.

```
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

다음 Java 코드 샘플은 Java 기반 애플리케이션을 사용하여 AWS 리전에 저장된 특정 발음 어휘의 내용을 생성하는 방법을 보여줍니다.

이 작업에 대한 자세한 내용은 `GetLexicon` API에 대한 참조 자료를 참조하십시오.

```
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

다음 Java 코드 샘플에서는 Java 기반 애플리케이션을 사용하여 AWS 리전에 저장된 발음 어휘 목록을 생성하는 방법을 보여줍니다.

이 작업에 대한 자세한 내용은 `ListLexicons` API에 대한 참조 자료를 참조하십시오.

```
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

다음 Java 코드 샘플은 Java 기반 애플리케이션을 사용하여 AWS 리전에 발음 어휘를 저장하는 방법을 보여 줍니다.

이 작업에 대한 자세한 내용은 `PutLexicon` API에 대한 참조 자료를 참조하십시오.

```
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);
        }
    }
}
```

스피치 마크

다음 코드 샘플은 Java 기반 애플리케이션을 사용하여 입력 텍스트의 스피치 마크를 합성하는 방법을 보여 줍니다. 이 기능은 SynthesizeSpeech API를 사용합니다.

이 기능에 대한 자세한 내용은 스피치 마크 (p. 14) 단원을 참조하십시오.

API에 대한 자세한 내용은 `SynthesizeSpeech` API에 대한 참조 자료를 참조하십시오.

```
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

다음 Java 코드 샘플은 Java 기반 애플리케이션을 사용하여 입력 텍스트의 스피치 마크를 합성하는 방법을 보여줍니다.

자세한 내용은 `SynthesizeSpeech` API에 대한 참조 자료를 참조하십시오.

```
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.");

        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 샘플

다음 코드 샘플은 Python(boto3) 기반 애플리케이션을 사용하여 Amazon Polly의 다양한 작업을 수행하는 방법을 보여줍니다. 이러한 샘플은 전체 예제 중 일부이며 AWS SDK for Python (Boto)을 사용하는 Python 애플리케이션에 포함될 수 있습니다.

코드 조각

- PutLexicon (p. 100)
- GetLexicon (p. 101)
- ListLexicon (p. 102)
- DeleteLexicon (p. 103)

PutLexicon

다음 코드 샘플은 Python(boto3) 기반 애플리케이션을 사용하여 AWS 리전에 발음 어휘를 저장하는 방법을 보여줍니다.

이 작업에 대한 자세한 내용은 `PutLexicon` API에 대한 참조 자료를 참조하십시오.

다음을 참조하십시오.

- 로컬 어휘 파일 이름 및 저장된 어휘 이름을 제공하여 코드를 업데이트해야 합니다.
- 예제에서는 하위 디렉터리 `pls`에 어휘 파일이 생성되어 있다고 가정합니다. 경로를 적절히 업데이트해야 합니다.

다음 코드 예제에서는 AWS SDK 구성 파일에 저장된 기본 자격 증명을 사용합니다. 구성 파일 생성에 대한 자세한 내용은 3.1단계: AWS Command Line Interface(AWS CLI) 설정 (p. 9) 항목을 참조하십시오.

```
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))
```

GetLexicon

다음 Python 코드 예제에서는 AWS SDK for Python (Boto)을 사용하여 AWS 리전에 저장된 모든 어휘를 검색합니다. 예제에서 어휘 이름을 명령줄 파라미터로 받아들이고 해당 어휘만 가져와 어휘가 로컬로 저장된 tmp 경로를 출력합니다.

다음 코드 예제에서는 AWS SDK 구성 파일에 저장된 기본 자격 증명을 사용합니다. 구성 파일 생성에 대한 자세한 내용은 3.1단계: AWS Command Line Interface(AWS CLI) 설정 (p. 9) 항목을 참조하십시오.

```
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

다음 Python 코드 예제는 AWS SDK for Python (Boto)을 사용하여 로컬 AWS 구성에 지정된 리전의 계정에 있는 어휘를 나열합니다. 구성 파일 생성에 대한 자세한 내용은 3.1단계: AWS Command Line Interface(AWS CLI) 설정 (p. 9) 항목을 참조하십시오.

```
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))
```

DeleteLexicon

다음 Python 코드 예제에서는 AWS SDK for Python (Boto)를 사용하여 로컬 AWS 구성에 지정된 리전에서 어휘를 삭제합니다. 예제에서는 지정된 어휘만 삭제합니다. 어휘를 삭제하기 전에 계속할지 확인하는 메시지가 나타납니다.

다음 코드 예제에서는 AWS SDK 구성 파일에 저장된 기본 자격 증명을 사용합니다. 구성 파일 생성에 대한 자세한 내용은 3.1단계: AWS Command Line Interface(AWS CLI) 설정 (p. 9) 항목을 참조하십시오.

```
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.")
```

예제 애플리케이션

이 단원에서는 Amazon Polly를 탐색하는 데 사용할 수 있는 예제 애플리케이션을 제공합니다.

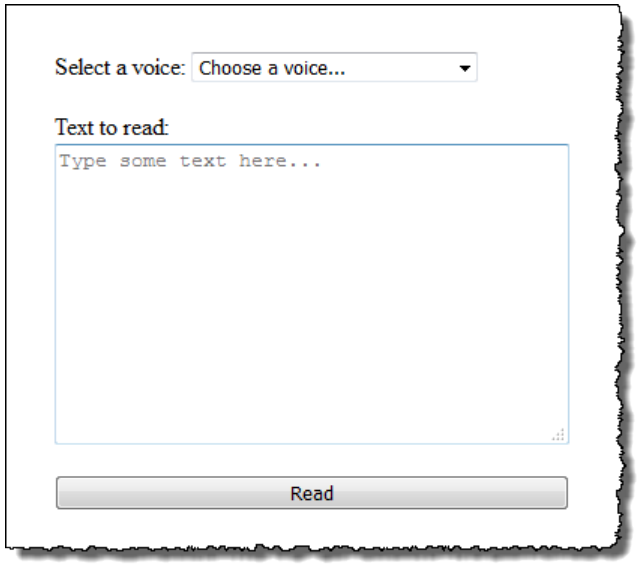
프로그래밍 언어별 예제 애플리케이션

- Python 예제(HTML5 클라이언트 및 Python 서버) (p. 104)

Python 예제(HTML5 클라이언트 및 Python 서버)

이 예제 애플리케이션은 다음과 같이 구성되어 있습니다.

- HTTP 조각난 전송 코딩을 사용하는 HTTP 1.1 서버(조각난 전송 코딩 참조)
- HTTP 1.1 서버와 상호 작용하는 간단한 HTML5 사용자 인터페이스(아래에 표시):



이 예제에서는 Amazon Polly를 사용하여 브라우저 기반 HTML5 애플리케이션에서 스피치를 스트리밍하는 방법을 보여줍니다. 대화 시스템, 화면 판독기 등 응답이 중요한 요인으로 간주되는 경우에는 텍스트가 합성될 때 Amazon Polly에서 생성된 오디오 스트림을 소비하는 것이 좋습니다.

이 예제 애플리케이션을 실행하려면 다음 사항이 필요합니다.

- HTML5 및 EcmaScript5 표준에 맞는 웹 브라우저(예: Chrome 23.0 이상, Firefox 21.0 이상, Internet Explorer 9.0 이상)
- Python 버전 3.0 이상

애플리케이션을 테스트하려면

1. 서버 코드를 `server.py`로 저장합니다. 코드는 Python 예제: Python 서버 코드(server.py) (p. 108) 항목을 참조하십시오.
2. HTML5 클라이언트 코드를 `index.html`로 저장합니다. 코드는 Python 예제: HTML5 사용자 인터페이스(index.html) (p. 105) 항목을 참조하십시오.
3. server.py를 저장한 경로에서 다음 명령을 실행하여 애플리케이션을 시작합니다(명령을 실행할 때 `python` 대신 `python3`을 사용해야 하는 일부 시스템에서).

```
$ python  server.py
```

애플리케이션이 시작되면 터미널에 URL이 나타납니다.

4. 터미널에 표시된 URL을 웹 브라우저에서 엽니다.

   파라미터로 사용할 애플리케이션 서버의 주소와 포트를 `server.py`에 전달할 수 있습니다. 자세한 내용을 보려면 `python server.py -h`를 실행하십시오.

5. 스피치를 들으려면 목록에서 음성을 선택하고 텍스트를 입력한 후 [Read]를 선택합니다. Amazon Polly에서 오디오 데이터의 첫 번째 사용 가능한 청크를 전송하자 마자 스피치가 재생되기 시작합니다.

6. 애플리케이션 테스트가 끝나 Python 서버를 중지하려면 서버가 실행되는 터미널에서 Ctrl+C를 누릅니다.

   Note

   서버에서 AWS SDK for Python (Boto)을 사용하여 Boto3 클라이언트를 생성합니다. 클라이언트는 컴퓨터의 AWS 구성 파일에 저장된 자격 증명을 사용하여 Amazon Polly에 대한 요청에 서명하고 인증합니다. AWS 구성 파일을 생성하고 자격 증명을 저장하는 방법에 대한 자세한 내용은 AWS Command Line Interface 사용 설명서에서 AWS 명령줄 인터페이스 구성 항목을 참조하십시오.

Python 예제: HTML5 사용자 인터페이스(index.html)

이 단원에서는 Python 예제(HTML5 클라이언트 및 Python 서버) (p. 104)에 설명된 HTML5 클라이언트용 코드를 제공합니다.

```
<html>

<head>
    <title>Text-to-Speech Example Application</title>
    <script>
        /*
         * This sample code requires a web browser with support for both the
         * HTML5 and ECMAScript 5 standards; the following is a non-comprehensive
         * list of compliant browsers and their minimum version:
         *
         * - Chrome 23.0+
         * - Firefox 21.0+
         * - Internet Explorer 9.0+
         * - Edge 12.0+
         * - Opera 15.0+
         * - Safari 6.1+
         * - Android (stock web browser) 4.4+
         * - Chrome for Android 51.0+
         * - Firefox for Android 48.0+
         * - Opera Mobile 37.0+
         * - iOS (Safari Mobile and Chrome) 3.2+
         * - Internet Explorer Mobile 10.0+
         * - Blackberry Browser 10.0+
         */

        // Mapping of the OutputFormat parameter of the SynthesizeSpeech API
        // and the audio format strings understood by the browser
        var AUDIO_FORMATS = {
            'ogg_vorbis': 'audio/ogg',
            'mp3': 'audio/mpeg',
            'pcm': 'audio/wave; codecs=1'
        };

        /**
         * Handles fetching JSON over HTTP
         */
        function fetchJSON(method, url, onSuccess, onError) {
```

```
    var request = new XMLHttpRequest();
    request.open(method, url, true);
    request.onload = function () {
        // If loading is complete
        if (request.readyState === 4) {
            // if the request was successful
            if (request.status === 200) {
                var data;

                // Parse the JSON in the response
                try {
                    data = JSON.parse(request.responseText);
                } catch (error) {
                    onError(request.status, error.toString());
                }

                onSuccess(data);
            } else {
                onError(request.status, request.responseText)
            }
        }
    };

    request.send();
}

/**
 * Returns a list of audio formats supported by the browser
 */
function getSupportedAudioFormats(player) {
    return Object.keys(AUDIO_FORMATS)
        .filter(function (format) {
            var supported = player.canPlayType(AUDIO_FORMATS[format]);
            return supported === 'probably' || supported === 'maybe';
        });
}

// Initialize the application when the DOM is loaded and ready to be
// manipulated
document.addEventListener("DOMContentLoaded", function () {
    var input = document.getElementById('input'),
        voiceMenu = document.getElementById('voice'),
        text = document.getElementById('text'),
        player = document.getElementById('player'),
        submit = document.getElementById('submit'),
        supportedFormats = getSupportedAudioFormats(player);

    // Display a message and don't allow submitting the form if the
    // browser doesn't support any of the available audio formats
    if (supportedFormats.length === 0) {
        submit.disabled = true;
        alert('The web browser in use does not support any of the' +
              ' available audio formats. Please try with a different' +
              ' one.');
    }

    // Play the audio stream when the form is submitted successfully
    input.addEventListener('submit', function (event) {
        // Validate the fields in the form, display a message if
        // unexpected values are encountered
        if (voiceMenu.selectedIndex <= 0 || text.value.length === 0) {
            alert('Please fill in all the fields.');
        } else {
            var selectedVoice = voiceMenu
                                    .options[voiceMenu.selectedIndex]
                                    .value;
```

```
                // Point the player to the streaming server
                player.src = '/read?voiceId=' +
                    encodeURIComponent(selectedVoice) +
                    '&text=' + encodeURIComponent(text.value) +
                    '&outputFormat=' + supportedFormats[0];
                player.play();
            }

            // Stop the form from submitting,
            // Submitting the form is allowed only if the browser doesn't
            // support Javascript to ensure functionality in such a case
            event.preventDefault();
        });

        // Load the list of available voices and display them in a menu
        fetchJSON('GET', '/voices',
            // If the request succeeds
            function (voices) {
                var container = document.createDocumentFragment();

                // Build the list of options for the menu
                voices.forEach(function (voice) {
                    var option = document.createElement('option');
                    option.value = voice['Id'];
                    option.innerHTML = voice['Name'] + ' (' +
                        voice['Gender'] + ', ' +
                        voice['LanguageName'] + ')';
                    container.appendChild(option);
                });

                // Add the options to the menu and enable the form field
                voiceMenu.appendChild(container);
                voiceMenu.disabled = false;
            },
            // If the request fails
            function (status, response) {
                // Display a message in case loading data from the server
                // fails
                alert(status + ' - ' + response);
            });
    });

</script>
<style>
    #input {
        min-width: 100px;
        max-width: 600px;
        margin: 0 auto;
        padding: 50px;
    }

    #input div {
        margin-bottom: 20px;
    }

    #text {
        width: 100%;
        height: 200px;
        display: block;
    }

    #submit {
        width: 100%;
    }
</style>
```

```
</head>

<body>
    <form id="input" method="GET" action="/read">
        <div>
            <label for="voice">Select a voice:</label>
            <select id="voice" name="voiceId" disabled>
                <option value="">Choose a voice...</option>
            </select>
        </div>
        <div>
            <label for="text">Text to read:</label>
            <textarea id="text" maxlength="1000" minlength="1" name="text"
                    placeholder="Type some text here..."></textarea>
        </div>
        <input type="submit" value="Read" id="submit" />
    </form>
    <audio id="player"></audio>
</body>

</html>
```

Python 예제: Python 서버 코드(server.py)

이 단원에서는 Python 예제(HTML5 클라이언트 및 Python 서버) (p. 104)에 설명된 Python 서버용 코드를 제공합니다.

```
"""
Example Python 2.7+/3.3+ Application

This application consists of a HTTP 1.1 server using the HTTP chunked transfer
coding (https://tools.ietf.org/html/rfc2616#section-3.6.1) and a minimal HTML5
user interface that interacts with it.

The goal of this example is to start streaming the speech to the client (the
HTML5 web UI) as soon as the first consumable chunk of speech is returned in
order to start playing the audio as soon as possible.
For use cases where low latency and responsiveness are strong requirements,
this is the recommended approach.

The service documentation contains examples for non-streaming use cases where
waiting for the speech synthesis to complete and fetching the whole audio stream
at once are an option.

To test the application, run 'python server.py' and then open the URL
displayed in the terminal in a web browser (see index.html for a list of
supported browsers). The address and port for the server can be passed as
parameters to server.py. For more information, run: 'python server.py -h'
"""
from argparse import ArgumentParser
from collections import namedtuple
from contextlib import closing
from io import BytesIO
from json import dumps as json_encode
import os
import sys

if sys.version_info >= (3, 0):
    from http.server import BaseHTTPRequestHandler, HTTPServer
    from socketserver import ThreadingMixIn
    from urllib.parse import parse_qs
else:
    from BaseHTTPServer import BaseHTTPRequestHandler, HTTPServer
    from SocketServer import ThreadingMixIn
```

```
    from urlparse import parse_qs

from boto3 import Session
from botocore.exceptions import BotoCoreError, ClientError

ResponseStatus = namedtuple("HTTPStatus",
                            ["code", "message"])

ResponseData = namedtuple("ResponseData",
                          ["status", "content_type", "data_stream"])

# Mapping the output format used in the client to the content type for the
# response
AUDIO_FORMATS = {"ogg_vorbis": "audio/ogg",
                 "mp3": "audio/mpeg",
                 "pcm": "audio/wave; codecs=1"}
CHUNK_SIZE = 1024
HTTP_STATUS = {"OK": ResponseStatus(code=200, message="OK"),
               "BAD_REQUEST": ResponseStatus(code=400, message="Bad request"),
               "NOT_FOUND": ResponseStatus(code=404, message="Not found"),
               "INTERNAL_SERVER_ERROR": ResponseStatus(code=500, message="Internal server
 error")}
PROTOCOL = "http"
ROUTE_INDEX = "/index.html"
ROUTE_VOICES = "/voices"
ROUTE_READ = "/read"


# Create a client using the credentials and region defined in the adminuser
# section of the AWS credentials and configuration files
session = Session(profile_name="adminuser")
polly = session.client("polly")


class HTTPStatusError(Exception):
    """Exception wrapping a value from http.server.HTTPStatus"""

    def __init__(self, status, description=None):
        """
        Constructs an error instance from a tuple of
        (code, message, description), see http.server.HTTPStatus
        """
        super(HTTPStatusError, self).__init__()
        self.code = status.code
        self.message = status.message
        self.explain = description


class ThreadedHTTPServer(ThreadingMixIn, HTTPServer):
    """An HTTP Server that handle each request in a new thread"""
    daemon_threads = True


class ChunkedHTTPRequestHandler(BaseHTTPRequestHandler):
    """"HTTP 1.1 Chunked encoding request handler"""
    # Use HTTP 1.1 as 1.0 doesn't support chunked encoding
    protocol_version = "HTTP/1.1"

    def query_get(self, queryData, key, default=""):
        """Helper for getting values from a pre-parsed query string"""
        return queryData.get(key, [default])[0]

    def do_GET(self):
        """Handles GET requests"""

        # Extract values from the query string
```

```
        path, _, query_string = self.path.partition('?')
        query = parse_qs(query_string)

        response = None

        print(u"[START]: Received GET for %s with query: %s" % (path, query))

        try:
            # Handle the possible request paths
            if path == ROUTE_INDEX:
                response = self.route_index(path, query)
            elif path == ROUTE_VOICES:
                response = self.route_voices(path, query)
            elif path == ROUTE_READ:
                response = self.route_read(path, query)
            else:
                response = self.route_not_found(path, query)

            self.send_headers(response.status, response.content_type)
            self.stream_data(response.data_stream)

        except HTTPStatusError as err:
            # Respond with an error and log debug
            # information
            if sys.version_info >= (3, 0):
                self.send_error(err.code, err.message, err.explain)
            else:
                self.send_error(err.code, err.message)

            self.log_error(u"%s %s %s - [%d] %s", self.client_address[0],
                           self.command, self.path, err.code, err.explain)

        print("[END]")

    def route_not_found(self, path, query):
        """Handles routing for unexpected paths"""
        raise HTTPStatusError(HTTP_STATUS["NOT_FOUND"], "Page not found")

    def route_index(self, path, query):
        """Handles routing for the application's entry point'"""
        try:
            return ResponseData(status=HTTP_STATUS["OK"], content_type="text_html",
                                # Open a binary stream for reading the index
                                # HTML file
                                data_stream=open(os.path.join(sys.path[0],
                                                              path[1:]), "rb"))
        except IOError as err:
            # Couldn't open the stream
            raise HTTPStatusError(HTTP_STATUS["INTERNAL_SERVER_ERROR"],
                                  str(err))

    def route_voices(self, path, query):
        """Handles routing for listing available voices"""
        params = {}
        voices = []

        while True:
            try:
                # Request list of available voices, if a continuation token
                # was returned by the previous call then use it to continue
                # listing
                response = polly.describe_voices(**params)
            except (BotoCoreError, ClientError) as err:
                # The service returned an error
                raise HTTPStatusError(HTTP_STATUS["INTERNAL_SERVER_ERROR"],
                                      str(err))
```

```
            # Collect all the voices
            voices.extend(response.get("Voices", []))

            # If a continuation token was returned continue, stop iterating
            # otherwise
            if "NextToken" in response:
                params = {"NextToken": response["NextToken"]}
            else:
                break

        json_data = json_encode(voices)
        bytes_data = bytes(json_data, "utf-8") if sys.version_info >= (3, 0) \
            else bytes(json_data)

        return ResponseData(status=HTTP_STATUS["OK"],
                            content_type="application/json",
                            # Create a binary stream for the JSON data
                            data_stream=BytesIO(bytes_data))

    def route_read(self, path, query):
        """Handles routing for reading text (speech synthesis)"""
        # Get the parameters from the query string
        text = self.query_get(query, "text")
        voiceId = self.query_get(query, "voiceId")
        outputFormat = self.query_get(query, "outputFormat")

        # Validate the parameters, set error flag in case of unexpected
        # values
        if len(text) == 0 or len(voiceId) == 0 or \
                outputFormat not in AUDIO_FORMATS:
            raise HTTPStatusError(HTTP_STATUS["BAD_REQUEST"],
                                  "Wrong parameters")
        else:
            try:
                # Request speech synthesis
                response = polly.synthesize_speech(Text=text,
                                                   VoiceId=voiceId,
                                                   OutputFormat=outputFormat)
            except (BotoCoreError, ClientError) as err:
                # The service returned an error
                raise HTTPStatusError(HTTP_STATUS["INTERNAL_SERVER_ERROR"],
                                      str(err))

            return ResponseData(status=HTTP_STATUS["OK"],
                                content_type=AUDIO_FORMATS[outputFormat],
                                # Access the audio stream in the response
                                data_stream=response.get("AudioStream"))

    def send_headers(self, status, content_type):
        """Send out the group of headers for a successful request"""
        # Send HTTP headers
        self.send_response(status.code, status.message)
        self.send_header('Content-type', content_type)
        self.send_header('Transfer-Encoding', 'chunked')
        self.send_header('Connection', 'close')
        self.end_headers()

    def stream_data(self, stream):
        """Consumes a stream in chunks to produce the response's output'"""
        print("Streaming started...")

        if stream:
            # Note: Closing the stream is important as the service throttles on
            # the number of parallel connections. Here we are using
            # contextlib.closing to ensure the close method of the stream object
```

```
            # will be called automatically at the end of the with statement's
            # scope.
            with closing(stream) as managed_stream:
                # Push out the stream's content in chunks
                while True:
                    data = managed_stream.read(CHUNK_SIZE)
                    self.wfile.write(b"%X\r\n%s\r\n" % (len(data), data))

                    # If there's no more data to read, stop streaming
                    if not data:
                        break

                # Ensure any buffered output has been transmitted and close the
                # stream
                self.wfile.flush()

            print("Streaming completed.")
        else:
            # The stream passed in is empty
            self.wfile.write(b"0\r\n\r\n")
            print("Nothing to stream.")

# Define and parse the command line arguments
cli = ArgumentParser(description='Example Python Application')
cli.add_argument(
    "-p", "--port", type=int, metavar="PORT", dest="port", default=8000)
cli.add_argument(
    "--host", type=str, metavar="HOST", dest="host", default="localhost")
arguments = cli.parse_args()

# If the module is invoked directly, initialize the application
if __name__ == '__main__':
    # Create and configure the HTTP server instance
    server = ThreadedHTTPServer((arguments.host, arguments.port),
                                ChunkedHTTPRequestHandler)
    print("Starting server, use <Ctrl-C> to stop...")
    print(u"Open {0}://{1}:{2}{3} in a web browser.".format(PROTOCOL,
                                                           arguments.host,
                                                           arguments.port,
                                                           ROUTE_INDEX))

    try:
        # Listen for requests indefinitely
        server.serve_forever()
    except KeyboardInterrupt:
        # A request to terminate has been received, stop the server
        print("\nShutting down...")
        server.socket.close()
```

Java 예제

이 예제에서는 Amazon Polly를 사용하여 Java 기반 애플리케이션에서 스피치를 스트리밍하는 방법을 보여 줍니다. 예제에서는 Java용 AWS SDK를 사용하여 목록에서 선택한 음성으로 지정된 텍스트를 읽습니다.

표시된 코드는 주요 작업에 해당하지만 사소한 오류 확인도 수행합니다. Amazon Polly에 오류가 생기면 애플리케이션이 종료됩니다.

이 예제 애플리케이션을 실행하려면 다음 사항이 필요합니다.

- Java 8 JDK(Java Development Kit)
- Java용 AWS SDK
- Apache Maven

애플리케이션을 테스트하려면

1. JAVA_HOME 환경 변수가 JDK에 맞게 설정되어 있어야 합니다.

   예를 들어 JDK 1.8.0_121을 Windows의 `C:\Program Files\Java\jdk1.8.0_121`에 설치한 경우 명령 프롬프트에 다음을 입력합니다.

   ```
   set JAVA_HOME=""C:\Program Files\Java\jdk1.8.0_121""
   ```

   JDK 1.8.0_121을 Linux의 `/usr/lib/jvm/java8-openjdk-amd64`에 설치한 경우 명령 프롬프트에 다음을 입력합니다.

   ```
   export JAVA_HOME=/usr/lib/jvm/java8-openjdk-amd64
   ```

2. 명령줄에서 Maven을 실행하도록 Maven 환경 변수를 설정합니다.

   예를 들어, Maven 3.3.9를 Windows의 `C:\Program Files\apache-maven-3.3.9`에 설치한 경우 다음을 입력합니다.

   ```
   set M2_HOME=""C:\Program Files\apache-maven-3.3.9""
   set M2=%M2_HOME%\bin
   set PATH=%M2%;%PATH%
   ```

   Maven 3.3.9를 Linux의 `/home/ec2-user/opt/apache-maven-3.3.9`에 설치한 경우 다음을 입력합니다.

   ```
   export M2_HOME=/home/ec2-user/opt/apache-maven-3.3.9
   export M2=$M2_HOME/bin
   export PATH=$M2:$PATH
   ```

3. 이름이 `polly-java-demo`인 디렉터리를 새로 생성합니다.
4. `polly-java-demo` 디렉터리에 이름이 `pom.xml`인 새 파일을 만들고 다음 코드를 파일에 붙여 넣습니다.

   ```
   <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. 이름이 `polly`인 새 디렉터리를 `src/main/java/com/amazonaws/demos`에 생성합니다.
6. `polly` 디렉터리에 이름이 `PollyDemo.java`인 새 Java 원본 파일을 만들고 다음 코드를 파일에 붙여 넣습니다.

```
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);
  SynthesizeSpeechResult synthRes = polly.synthesizeSpeech(synthReq);

  return synthRes.getAudioStream();
 }

 public static void main(String args[]) throws Exception {
  //create the test class
  PollyDemo helloWorld = new PollyDemo(Region.getRegion(Regions.US_EAST_1));
  //get the audio stream
  InputStream speechStream = helloWorld.synthesize(SAMPLE, OutputFormat.Mp3);

  //create an MP3 player
  AdvancedPlayer player = new AdvancedPlayer(speechStream,
    javazoom.jl.player.FactoryRegistry.systemRegistry().createAudioDevice());

  player.setPlayBackListener(new PlaybackListener() {
   @Override
   public void playbackStarted(PlaybackEvent evt) {
    System.out.println("Playback started");
    System.out.println(SAMPLE);
   }

   @Override
   public void playbackFinished(PlaybackEvent evt) {
    System.out.println("Playback finished");
   }
  });


  // play it!
  player.play();

 }
}
```

7. `polly-java-demo` 디렉터리로 돌아가 데모를 지우고, 컴파일하며, 실행합니다.

```
mvn clean compile exec:java
```

iOS 예제

다음 예제에서는 Amazon Polly용 iOS SDK를 사용하여 음성 목록에서 선택한 음성으로 지정된 텍스트를 읽습니다.

여기에 표시된 코드는 주요 작업에 해당하지만 오류는 처리하지 않습니다. 전체 코드는 AWS SDK for iOS Amazon Polly demo 항목을 참조하십시오.

초기화

```
// 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()
```

사용 가능한 음성 목록 가져오기

```
// 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
})
```

스피치 합성

```
// 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 예제

다음 예제에서는 Amazon Polly용 Android SDK를 사용하여 음성 목록에서 선택한 음성으로 지정된 텍스트를 읽습니다.

여기에 표시된 코드는 주요 작업에 해당하지만 오류는 처리하지 않습니다. 전체 코드는 AWS SDK for Android Amazon Polly demo 항목을 참조하십시오.

초기화

```
// 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);
```

사용 가능한 음성 목록 가져오기

```
// 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();
```

오디오 스트림 URL 가져오기

```
// 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);
```

합성된 스피치 재생

```
// Use MediaPlayer: https://developer.android.com/guide/topics/media/mediaplayer.html

// Create a media player to play the synthesized audio stream.
MediaPlayer mediaPlayer = new MediaPlayer();
mediaPlayer.setAudioStreamType(AudioManager.STREAM_MUSIC);
```

```
try {
    // Set media player's data source to previously obtained URL.
    mediaPlayer.setDataSource(presignedSynthesizeSpeechUrl.toString());
} catch (IOException e) {
    Log.e(TAG, "Unable to set data source for the media player! " + e.getMessage());
}

// Prepare the MediaPlayer asynchronously (since the data source is a network stream).
mediaPlayer.prepareAsync();

// Set the callback to start the MediaPlayer when it's prepared.
mediaPlayer.setOnPreparedListener(new MediaPlayer.OnPreparedListener() {
    @Override
    public void onPrepared(MediaPlayer mp) {
        mp.start();
    }
});

// Set the callback to release the MediaPlayer after playback is completed.
mediaPlayer.setOnCompletionListener(new MediaPlayer.OnCompletionListener() {
    @Override
    public void onCompletion(MediaPlayer mp) {
 mp.release();
    }
});
```

Amazon Polly용 WordPress 플러그인

WordPress용 Amazon Polly 플러그인을 사용하면 WordPress 웹 사이트 방문자가 선택한 언어와 음성으로 콘텐츠를 들을 수 있습니다. 플러그인을 사용하여 작성된 콘텐츠의 오디오 파일을 만들면 사용자가 편리하게 스트리밍할 수 있습니다. 웹 사이트 방문자는 인라인 오디오 플레이어 및 모바일 애플리케이션과 같은 새로운 채널을 사용하여 콘텐츠를 사용할 수 있습니다.

플러그인을 설치 및 구성하려면 기본 WordPress 플러그인 추가 페이지를 사용합니다 플러그인을 설치 및 활성화한 후에는 Amazon Polly 설정 페이지를 열고 플러그인을 AWS 계정에 연결합니다. 게시할 때 새 콘텐츠의 오디오 파일을 자동으로 만들거나 콘텐츠를 개별적으로 선택하도록 플러그인을 구성할 수 있습니다. 아카이브된 콘텐츠는 배치 처리되어 소비자에게 향상된 경험을 제공할 수 있습니다. Amazon Pollycast RSS 피드를 사용하여 새 오디오 콘텐츠를 팟캐스트할 수도 있습니다.

> **Note**
>
> 다음 절차에서는 WordPress에서 수행되는 작업에 대한 지침이 WordPress 사용자 인터페이스와 정확하게 일치하지 않을 수 있습니다.

항목

플러그인 설치

WordPress용 Amazon Polly 플러그인을 사용하려면 작업 중인 WordPress 설치 외에도 활성 AWS 계정이 있어야 합니다. 계정이 없는 경우 1.1단계: AWS에 가입 (p. 6) 단원을 참조하십시오. AWS 계정이 없는 경우 다음 단계에 따라 플러그인 설정을 완료합니다.

1. 권한 정책 생성 (p. 121)
2. 플러그인용 IAM 사용자 생성 (p. 122)
3. 플러그인 설치 및 구성 (p. 122)

권한 정책 생성

AWS Management 콘솔에서 다음 코드를 사용하여 PollyForWordPressPolicy라는 IAM 권한 정책을 생성합니다.

```
{
    "Version": "2012-10-17",
    "Statement": [use
        {
            "Sid": "Permissions1",
            "Effect": "Allow",
            "Action": [
                "polly:SynthesizeSpeech",
                "s3:HeadBucket",
                "polly:DescribeVoices"
```

```
            ],
           "Resource": "*"
        },
        {
            "Sid": "Permissions2",
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket",
                "s3:GetBucketAcl",
                "s3:GetBucketPolicy",
                "s3:PutObject”,
                                "s3:DeleteObject",
                "s3:CreateBucket",
                "s3:PutObjectAcl"
            ],
            "Resource": "arn:aws:s3:::audio_for_wordpress*"
        }
    ]
}
```

권한 정책을 만드는 방법에 대한 자세한 내용은 고객 관리형 정책 생성을 참조하십시오.

플러그인용 IAM 사용자 생성

플러그인을 AWS 계정에 연결하려면 AWS Identity and Access Management(IAM) 사용자를 만든 다음 권한 정책을 해당 사용자에게 연결합니다. Amazon EC2에 WordPress를 배포한 경우 이 단계를 건너뛰고 개별 IAM 사용자 대신 IAM 역할을 사용할 수 있습니다. 자세한 내용은 Amazon EC2 사용 설명서의 Amazon EC2용 IAM 역할을 참조하십시오.

IAM 사용자를 생성하려면

1. https://console.aws.amazon.com/iam/에서 IAM 콘솔을 열고 사용자를 선택합니다.
2. 사용자 추가를 선택합니다.
3. 사용자 이름에 **`WordPress`**를 입력합니다.
4. Access Type(액세스 유형)에서 Programmatic access(프로그래밍 방식 액세스)를 선택한 다음 다음: 권한을 선택합니다.
5. Attach existing policies direction(기존 정책 방향 연결)을 선택합니다. 목록에서 방금 생성한 정책(PollyForWordPressPolicy)을 선택한 후 다음: 검토를 선택합니다.
6. 사용자 생성을 선택합니다.
7. 액세스 키 ID(Access key ID) 및 Secret access key(보안 액세스 키)를 복사합니다. 이러한 키는 플러그인을 구성할 때 필요합니다.

   Important

   이번 한 번만 이러한 키에 액세스할 수 있으므로 기록해 두는 것이 좋습니다.

플러그인 설치 및 구성

플러그인을 설치 및 구성하려면

1. Amazon Polly 플러그인 GitHub 사이트에서 WordPress용 Amazon Polly 플러그인을 다운로드합니다.
2. WordPress Admin 페이지에서 Add New Plugin을 선택한 다음 플러그인을 설치 및 활성화합니다.
3. WordPress Admin 페이지에서 Settings을 선택한 다음 Amazon Polly Settings에서 플러그인을 구성합니다.

- AWS 액세스 키 및 AWS 보안 키 - 플러그인에서 Amazon Polly 및 Amazon Simple Storage Service(Amazon S3)를 사용할 수 있도록 해 주는 AWS 자격 증명. Amazon Elastic Compute Cloud(Amazon EC2)에서 WordPress 사이트를 호스팅하는 경우 자격 증명 대신 AWS Identity and Access Management(IAM) 역할을 사용할 수 있습니다. 이 경우 이러한 두 필드를 공백 상태로 남겨둡니다.
- 샘플 비율 - 생성할 오디오 파일의 샘플 비율(Hz). 샘플 비율이 높을수록 오디오 품질이 높아집니다.
- 음성 이름 - 오디오 파일을 생성하는 데 사용할 Amazon Polly 음성.
- 플레이어 위치 - 웹 사이트에서의 오디오 플레이어 위치. 게시물의 앞이나 뒤에 넣거나 전혀 사용하지 않을 수 있습니다. Amazon Pollycast를 사용하여 파일을 팟캐스트로 제공하려면 오디오 플레이어를 표시하지 않도록 선택합니다.
- 새 게시 기본값 - Amazon Polly가 모든 새 게시물에 대해 자동으로 활성화되는지 여부를 지정합니다. Amazon Polly에서 구성 설정을 사용하여 새 게시물마다 오디오 파일을 만들려면 이 옵션을 선택합니다.
- 자동 재생 - 사용자가 웹 사이트의 개별 게시물을 방문할 때 오디오 플레이어에서 자동으로 오디오 재생을 시작할지 여부를 지정합니다.
- Amazon S3에 오디오 저장 - 오디오 파일을 서버가 아닌 S3 버킷에 저장하려면 이 옵션을 선택합니다. Amazon Polly에서 자동으로 버킷을 만듭니다. 자세한 내용 및 요금은 Amazon S3를 참조하십시오.
- Amazon CloudFront(CDN) 도메인 이름 - Amazon CloudFront를 사용하여 오디오 파일을 브로드캐스트하려면 플러그인이 오디오 스트리밍에 사용할 CloudFront 도메인의 이름을 제공합니다. 먼저 Amazon CloudFront에 도메인을 만들어야 합니다.
- ITunes 범주 - 팟캐스트 범주. 범주를 선택하면 팟캐스트 사용자가 팟캐스트 범주에서 팟캐스트를 쉽게 찾을 수 있습니다.
- ITunes 명시적 허용 - Amazon Pollycast 팟캐스트를 사용할지 여부를 지정합니다.
- 모든 게시물 일괄 업데이트 - 새 플러그인 설정을 사용하도록 모든 게시물을 변환하려면 이 옵션을 선택합니다.

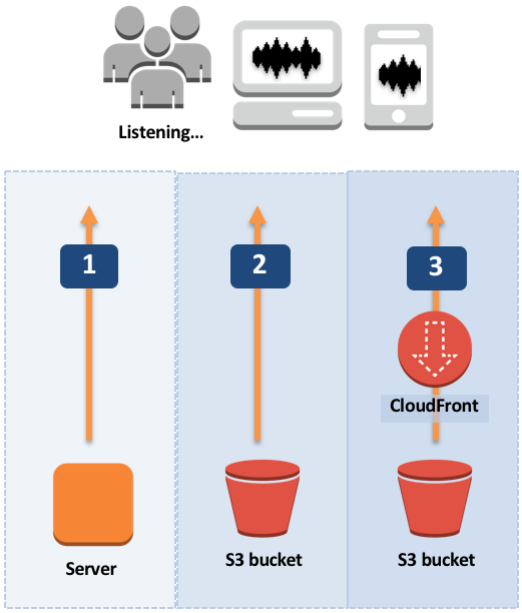
4. 변경 사항 저장을 선택합니다.

오디오 파일 저장

사이트에 콘텐츠를 게시하면 합성을 위해 Amazon Polly로 전송됩니다. 기본적으로 새 오디오 파일은 웹 서버에 저장됩니다. Amazon S3 또는 Amazon CloudFront를 사용하여 파일을 저장할 수도 있습니다.

WordPress 사용자가 웹 사이트에서 오디오 콘텐츠를 청취하는 방식은 콘텐츠가 저장된 위치에 따라 다릅니다.

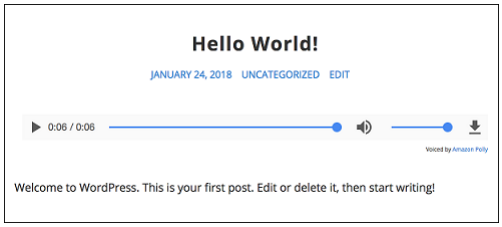
1. WordPress 서버에 저장된 오디오 파일의 경우 파일은 서버에서 직접 브로드캐스트됩니다.
2. Amazon S3 버킷에 저장된 파일의 경우 S3 버킷에서 브로드캐스트됩니다.
3. Amazon CloudFront(CDN)를 사용하는 경우 파일은 Amazon S3에 저장되고 Amazon CloudFront와 함께 브로드캐스트됩니다.



사용자는 오디오 파일을 저장하는 방법과 상관없이 동일한 청취 경험을 갖습니다.

HTML 플레이어 위치 지정

Amazon Polly 플러그인을 설치하면 WordPress 웹 사이트에 HTML 플레이어가 표시됩니다.



Amazon Polly 설정 페이지에서 플러그인을 구성할 때 플레이어 위치 옵션을 사용하여 텍스트 위 또는 아래에 플레이어를 표시하거나 아예 표시하지 않도록 설정할 수 있습니다. 설정 구성 옵션에 대한 자세한 내용은 플러그인 설치 및 구성 (p. 122) 단원을 참조하십시오.

Amazon Pollycast로 팟캐스트

사용자가 표준 팟캐스트 애플리케이션을 사용하여 오디오 콘텐츠를 들을 수 있게 하려면 Pollycast 피드를 설정합니다. RSS 2.0 호환 Pollycast 피드는 인기 있는 팟캐스트 모바일 애플리케이션 및 iTunes와 같은 팟캐스트 디렉터리를 통해 필요한 XML 데이터를 제공합니다. Amazon Polly 플러그인은 모든 WordPress 아카이브 URL에 Amazon Pollycast 엔드포인트를 자동으로 추가합니다. 이를 통해 사이트 전체 또는 대상 팟캐스트를 합성할 수 있습니다.

또한 팟캐스트 애플리케이션에서 페이지의 URL에 `/amazon-pollycast/`를 추가하여 Amazon Pollycast 엔드포인트를 수동으로 추가할 수 있습니다. 예:

```
example.com/amazon-pollycast/
example.com/category/news/amazon-pollycast/
```

```
example.com/author/john/amazon-pollcast/
```

Amazon Polly 참조

이 단원에서는 Amazon Polly 문서에 대한 추가 참조 자료를 제공합니다.

항목

사용 가능한 음성

Amazon Polly 사용 시 다음 음성을 사용할 수 있습니다.

이름/ID	성별
덴마크어(da-DK)	
Mads	남성
Naja	여성
네덜란드어(nl-NL)	
Lotte	여성
Ruben	남성
영어(호주)(en-AU)	
Nicole	여성
Russell	남성
영어(영국)(en-GB)	
Amy	여성
Brian	남성
Emma	여성
영어(인도)(en-IN)	
Aditi	여성
Raveena	여성
영어(미국)(en-US)	
Ivy	여성
Joanna	여성
Joey	남성
Justin	남성
Kendra	여성

이름/ID	성별
Kimberly	여성
Matthew	남성
Salli	여성
영어(웨일스)(en-GB-WLS)	
Geraint	남성
프랑스어(fr-FR)	
Céline/Celine	여성
Mathieu	남성
프랑스어(캐나다)(fr-CA)	
Chantal	여성
독일어(de-DE)	
Hans	남성
Marlene	여성
Vicki	여성
아이슬란드어(is-IS)	
Dóra/Dora	여성
Karl	남성
이탈리아어(it-IT)	
Carla	여성
Giorgio	남성
일본어(ja-JP)	
Mizuki	여성
Takumi	남성
한국어(ko-KR)	
Seoyeon	여성
노르웨이어(nb-NO)	
Liv	여성
폴란드어(pl-PL)	
Jacek	남성
1월	남성
Ewa	여성

이름/ID	성별
Maja	여성
포르투갈어(브라질)(pt-BR)	
Ricardo	남성
Vitória/Vitoria	여성
포르투갈어(유럽)(pt-PT)	
Cristiano	남성
Inês/Ines	여성
루마니아어(ro-RO)	
Carmen	여성
러시아어(ru-RU)	
Maxim	남성
Tatyana	여성
스페인어(카스티야)(es-ES)	
Conchita	여성
Enrique	남성
스페인어(남미)(es-US)	
Miguel	남성
Penélope/Penelope	여성
스웨덴어(sv-SE)	
Astrid	여성
터키어(tr-TR)	
Filiz	여성
웨일스어(cy-GB)	
Gwyneth	여성

Amazon Polly에서 지원하는 언어

언어	언어 코드
덴마크어	da-DK
네덜란드어	nl-NL
영어(호주)	en-AU

언어	언어 코드
영어(영국)	en-GB
영어(인도)	en-IN
영어(미국)	en-US
영어(웨일스)	en-GB-WLS
프랑스어	fr-FR
프랑스어(캐나다)	fr-CA
독일어	de-DE
아이슬란드어	is-IS
이탈리아어	it-IT
일본어	ja-JP
한국어	ko-KR
노르웨이어	nb-NO
폴란드어	pl-PL
포르투갈어(브라질)	pt-BR
포르투갈어(유럽)	pt-PT
루마니아어	ro-RO
러시아어	ru-RU
스페인어	es-ES
스페인어(남미)	es-US
스웨덴어	sv-SE
터키어	tr-TR
웨일스어	cy-GB

Amazon Polly 제한

다음은 Amazon Polly를 사용할 때 알아야 할 제한 사항입니다.

지원하는 리전

Amazon Polly를 사용할 수 있는 AWS 리전 목록은 Amazon Web Services 일반 참조의 AWS 리전 및 엔드포인트를 참조하십시오.

Throttling

- 계정당 조절 속도: 100건의 초당 트랜잭션(요청)(tps) 및 버스트 한도 120tps입니다.

  계정당 동시 연결 수: 90
- 작업당 조절 속도:

작업	한도
어휘	

`DeleteLexicon`
`PutLexicon`
`GetLexicon`
`ListLexicons`
	이 작업이 완료된 후 2건의 초당 트랜잭션(tps)입니다.
	허용되는 최대 버스트는 4tps입니다.

음성	
`DescribeVoices`	80tps 및 버스트 한도 100tps
`SynthesizeSpeech`	80tps 및 버스트 한도 100tps

발음 어휘

- 계정당 어휘를 최대 100개까지 저장할 수 있습니다.
- 어휘 이름은 최대 20자의 영숫자 문자열로 지정할 수 있습니다.
- 각 어휘의 최대 크기는 4,000자입니다. 어휘의 크기는 SynthesizeSpeech 작업의 지연 시간에 영향을 줍니다.
- 어휘의 각 <phoneme> 또는 <alias> 대체에 대해 최대 100자까지 지정할 수 있습니다.

어휘를 사용하는 방법에 대한 자세한 내용은 어휘 관리 (p. 84) 단원을 참조하십시오.

SynthesizeSpeech API 작업

SynthesizeSpeech API 작업 사용과 관련하여 다음 제한 사항에 유의하십시오.

- 청구되는 입력 텍스트의 최대 크기는 1,500자(총 3,000자)입니다. SSML 태그는 청구 문자 수로 계산되지 않습니다.
- 입력 텍스트에 적용할 최대 5개의 어휘를 지정할 수 있습니다.
- 출력 오디오 스트림(합성)은 5분으로 제한되며, 이 시간을 초과할 경우 나머지 음성은 잘립니다.

자세한 내용은 SynthesizeSpeech (p. 148) 섹션을 참조하십시오.

> **Note**
>
> `SynthesizeSpeech` API 작업의 몇 가지 제한을 AWS Batch 또는 다른 서비스를 사용하여 우회할 수 있습니다. AWS Batch에 대한 자세한 내용은 AWS Batch란 무엇인가?를 참조하십시오

SSML(Speech Synthesis Markup Language)

SSML 사용과 관련하여 다음 제한 사항에 유의하십시오.

- `<audio>`, `<lexicon>`, `<lookup>` 및 `<voice>` 태그는 지원되지 않습니다.
- `<break>` 요소는 각각 10초의 최대 기간을 지정할 수 있습니다.
- `<prosody>` 태그는 -80% 미만의 비율 속성 값을 지원하지 않습니다.

자세한 내용은 SSML 사용 (p. 62) 섹션을 참조하십시오.

AWS CloudTrail을 사용하여 Amazon Polly API 호출 로깅

Amazon Polly에서는 모든 Amazon Polly API 호출을 캡처하고 로그 파일을 사용자가 지정한 Amazon S3 버킷에 전달하는 서비스인 CloudTrail에 포함되어 있습니다. CloudTrail은 Amazon Polly 콘솔 또는 Amazon Polly API에 대한 코드에서 API 호출을 캡처합니다. CloudTrail에서 수집하는 정보를 사용하여 Amazon Polly에 어떤 요청이 이루어졌는지, 어떤 소스 IP 주소에서 요청했는지, 누가 언제 요청했는지 등을 확인할 수 있습니다.

그 구성 및 활성화 방법을 포함하여 CloudTrail에 대한 자세한 내용은 AWS CloudTrail User Guide 단원을 참조하십시오.

CloudTrail의 Amazon Polly 정보

AWS 계정에서 CloudTrail 로깅을 활성화하면, Amazon Polly 작업에 적용된 API 호출이 CloudTrail 로그 파일로 추적되어 다른 AWS 서비스 기록과 함께 작성됩니다. CloudTrail는 기간 및 파일 크기를 기준으로 새 파일을 만들고 기록하는 시점을 결정합니다.

모든 Amazon Polly 작업이 CloudTrail에서 로깅되고 Amazon Polly API 참조 (p. 137)에 문서화됩니다. 다음 작업이 지원됩니다.

- DeleteLexicon (p. 138)
- DescribeVoices (p. 140)
- GetLexicon (p. 142)
- ListLexicons (p. 144)
- PutLexicon (p. 146)
- SynthesizeSpeech (p. 148)

모든 로그 항목은 누가 요청을 생성했는가에 대한 정보가 들어 있습니다. 로그 항목의 사용자 신원 정보를 이용하면 다음을 쉽게 판단할 수 있습니다.

- 요청을 루트로 했는지, IAM 사용자 자격 증명으로 했는지
- 역할 또는 연합된 사용자에 대한 임시 보안 자격 증명을 사용하여 요청이 생성되었는지 여부
- 요청을 다른 AWS 서비스로 했는지

자세한 내용은 CloudTrail userIdentity 요소를 참조하십시오.

원하는 기간만큼 Amazon S3 버킷에 로그 파일을 저장할 수 있습니다. 그러나 Amazon S3 수명 주기 규칙을 정의하여 자동으로 로그 파일을 보관하거나 삭제할 수도 있습니다. 기본적으로 로그 파일은 Amazon S3 서버 측 암호화(SSE)를 통해 암호화됩니다.

로그 파일 전송 시 알림을 받으려면, 새로운 로그 파일이 전달될 때 Amazon SNS 알림이 게시되도록 CloudTrail를 구성하십시오. 자세한 내용은 CloudTrail용 Amazon SNS 알림 설정을 참조하십시오.

또한 여러 AWS 리전 및 여러 AWS 계정의 Amazon Polly 로그 파일을 하나의 Amazon S3 버킷으로 통합할 수도 있습니다.

자세한 내용은 여러 리전에서 CloudTrail 로그 파일 받기와 여러 계정에서 CloudTrail 로그 파일 받기를 참조하십시오.

Amazon Polly 로그 파일 항목 이해

CloudTrail 로그 파일에는 하나 이상의 로그 항목이 포함될 수 있습니다. 각 항목은 여러 개의 JSON 형식 이벤트를 표시합니다. 로그 항목은 어떤 소스로부터의 단일 요청을 나타내며 요청된 작업, 작업 날짜와 시간, 요청 파라미터 등에 대한 정보가 들어 있습니다. 로그 항목은 퍼블릭 API 호출의 주문 스택 트레이스가 아니기 때문에 특정 순서로 표시되지 않습니다.

잠재적인 기밀성 문제로 인해 로그 항목은 합성 텍스트를 포함하지 않습니다. 대신에 이 텍스트는 로그 항목에서 수정됩니다.

다음 예시는 `SynthesizeSpeech`를 보여주는 CloudTrail 로그 항목입니다.

```
{
    "Records": [
        {
            "awsRegion": "us-east-2",
            "eventID": "19bd70f7-5e60-4cdc-9825-936c552278ae",
            "eventName": "SynthesizeSpeech",
            "eventSource": "tts.amazonaws.com",
            "eventTime": "2016-11-02T03:49:39Z",
            "eventType": "AwsApiCall",
            "eventVersion": "1.05",
            "recipientAccountId": "123456789012",
            "requestID": "414288c2-a1af-11e6-b17f-d7cfc06cb461",
            "requestParameters": {
                "lexiconNames": [
                    "SampleLexicon"
                ],
                "outputFormat": "mp3",
                "sampleRate": "22050",
                "text": "**********",
                "textType": "text",
                "voiceId": "Kendra"
            },
            "responseElements": {
                "contentType": "audio/mpeg",
                "requestCharacters": 25
            },
            "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"
            }
        }

    ]
}
```

`eventName` 요소는 발생한 작업을 구별하며 `"SynthesizeSpeech20161128"`와 같은 날짜 및 버전 정보를 포함할 수 있지만 여전히 동일한 퍼블릭 API를 참조합니다.

CloudWatch를 Amazon Polly과 통합

Amazon Polly는 사용자와 상호 작용할 때 매분 다음 지표와 차원을 CloudWatch로 전송합니다. 다음 절차에 따라 Amazon Polly에 대한 지표를 볼 수 있습니다.

Amazon Polly에서 원시 데이터를 수집하여 읽기 가능하며 실시간에 가까운 지표로 처리하는 CloudWatch를 통해 Amazon Polly를 모니터링할 수 있습니다. 이러한 통계는 2주간 기록되므로 `historical information`을 보고 웹 애플리케이션이나 서비스가 어떻게 실행되고 있는지 전체적으로 더 잘 파악할 수 있습니다. Amazon Polly 지표 데이터는 기본적으로 1분 간격으로 CloudWatch에 전송됩니다. 자세한 내용은 Amazon CloudWatch 사용 설명서의 What Is Amazon CloudWatch를 참조하십시오.

CloudWatch 측정치 가져오기(콘솔)

1. https://console.aws.amazon.com/cloudwatch/에서 CloudWatch 콘솔을 엽니다.
2. 탐색 창에서 [Metrics]를 선택합니다.
3. [CloudWatch Metrics by Category] 창의 Amazon Polly에 대한 지표 범주에서 지표 범주를 선택하고 위쪽 창에서 아래로 스크롤하여 모든 지표 목록을 봅니다.

CloudWatch 측정치 가져오기(CLI)

다음 코드는 Amazon Polly에 사용 가능한 측정치를 표시합니다.

```
aws cloudwatch list-metrics --namespace "AWS/Polly"
```

이전 명령은 다음과 비슷한 Amazon Polly 측정치 목록을 반환합니다. `MetricName` 요소는 측정치를 식별합니다.

```
{
    "Metrics": [
        {
            "Namespace": "AWS/Polly",
            "Dimensions": [
                {
                    "Name": "Operation",
                    "Value": "SynthesizeSpeech"
                }
            ],
            "MetricName": "ResponseLatency"
        },
        {
            "Namespace": "AWS/Polly",
            "Dimensions": [
                {
                    "Name": "Operation",
                    "Value": "SynthesizeSpeech"
                }
            ],
            "MetricName": "RequestCharacters"
        }
```

자세한 내용은 Amazon CloudWatch API 참조의 GetMetricStatistics를 참조하십시오.

Amazon Polly 지표

Amazon Polly은(는) 각 요청에 대한 다음 지표를 만듭니다. 이러한 지표는 집계된 다음 1분 간격으로 CloudWatch(으)로 전송되며 집계된 지표는 CloudWatch에서 사용할 수 있습니다.

측정치	설명
`RequestCharacters`	요청의 문자 수. 청구 가능한 문자이며 SSML 태그가 포함되지 않습니다. 유효한 차원: Operation 유효한 통계: Minimum, Maximum, Average, SampleCount, Sum 단위: 수
`ResponseLatency`	요청 시간과 스트리밍 응답이 시작되는 시간 사이의 지연 시간 유효한 차원: Operation 유효한 통계: Minimum, Maximum, Average, SampleCount 단위: 밀리초
`2XXCount`	성공적 응답 시 반환되는 HTTP 200 레벨 코드 유효한 차원: Operation 유효한 통계: Average, SampleCount, Sum 단위: 수
`4XXCount`	오류 발생 시 반환되는 HTTP 400 레벨 오류 코드. 성공적 응답이 발생할 때마다 0을 내보냅니다. 유효한 차원: Operation 유효한 통계: Average, SampleCount, Sum 단위: 수
`5XXCount`	오류 발생 시 반환되는 HTTP 500 레벨 오류 코드. 성공적 응답이 발생할 때마다 0을 내보냅니다. 유효한 차원: Operation 유효한 통계: Average, SampleCount, Sum 단위: 수

Amazon Polly 지표 차원

Amazon Polly 지표는 AWS/Polly 네임스페이스를 사용하며 다음 차원의 지표를 제공합니다.

차원	설명
`Operation`	지표는 참조하는 API 메서드별로 그룹화됩니다. 가능한 값은 `SynthesizeSpeech`, `PutLexicon`, `DescribeVoices` 등입니다.

Amazon Polly API 참조

이 섹션에는 Amazon Polly API 참조가 포함되어 있습니다.

> **Note**
>
> 인증된 API 호출은 서명 버전 4 서명 프로세스를 사용하여 서명해야 합니다. 자세한 내용은 Amazon Web Services 일반 참조의 AWS API 요청 서명을 참조하십시오.

항목

Actions

The following actions are supported:

DeleteLexicon

Deletes the specified pronunciation lexicon stored in an AWS Region. A lexicon which has been deleted is not available for speech synthesis, nor is it possible to retrieve it using either the `GetLexicon` or `ListLexicon` APIs.

For more information, see Managing Lexicons.

Request Syntax

```
DELETE /v1/lexicons/LexiconName HTTP/1.1
```

URI Request Parameters

The request requires the following URI parameters.

Name (p. 138)

The name of the lexicon to delete. Must be an existing lexicon in the region.

Pattern: `[0-9A-Za-z]{1,20}`

Request Body

The request does not have a request body.

Response Syntax

```
HTTP/1.1 200
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

LexiconNotFoundException

Amazon Polly can't find the specified lexicon. This could be caused by a lexicon that is missing, its name is misspelled or specifying a lexicon that is in a different region.

Verify that the lexicon exists, is in the region (see ListLexicons (p. 144)) and that you spelled its name is spelled correctly. Then try again.

HTTP Status Code: 404

ServiceFailureException

An unknown condition has caused a service failure.

HTTP Status Code: 500

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

DescribeVoices

Returns the list of voices that are available for use when requesting speech synthesis. Each voice speaks a specified language, is either male or female, and is identified by an ID, which is the ASCII version of the voice name.

When synthesizing speech (`SynthesizeSpeech`), you provide the voice ID for the voice you want from the list of voices returned by `DescribeVoices`.

For example, you want your news reader application to read news in a specific language, but giving a user the option to choose the voice. Using the `DescribeVoices` operation you can provide the user with a list of available voices to select from.

You can optionally specify a language code to filter the available voices. For example, if you specify `en-US`, the operation returns a list of all available US English voices.

This operation requires permissions to perform the `polly:DescribeVoices` action.

Request Syntax

```
GET /v1/voices?LanguageCode=LanguageCode&NextToken=NextToken HTTP/1.1
```

URI Request Parameters

The request requires the following URI parameters.

LanguageCode (p. 140)

The language identification tag (ISO 639 code for the language name-ISO 3166 country code) for filtering the list of voices returned. If you don't specify this optional parameter, all available voices are returned.

Valid Values: `cy-GB | da-DK | de-DE | en-AU | en-GB | en-GB-WLS | en-IN | en-US | es-ES | es-US | fr-CA | fr-FR | is-IS | it-IT | ja-JP | nb-NO | nl-NL | pl-PL | pt-BR | pt-PT | ro-RO | ru-RU | sv-SE | tr-TR`

NextToken (p. 140)

An opaque pagination token returned from the previous `DescribeVoices` operation. If present, this indicates where to continue the listing.

Request Body

The request does not have a request body.

Response Syntax

```
HTTP/1.1 200
Content-type: application/json

{
   "NextToken": "string",
   "Voices": [
      {
         "Gender": "string",
         "Id": "string",
```

```
            "LanguageCode": "string",
            "LanguageName": "string",
            "Name": "string"
        }
    ]
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

NextToken (p. 140)

The pagination token to use in the next request to continue the listing of voices. `NextToken` is returned only if the response is truncated.

Type: String

Voices (p. 140)

A list of voices with their properties.

Type: Array of Voice (p. 156) objects

Errors

InvalidNextTokenException

The NextToken is invalid. Verify that it's spelled correctly, and then try again.

HTTP Status Code: 400

ServiceFailureException

An unknown condition has caused a service failure.

HTTP Status Code: 500

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

GetLexicon

Returns the content of the specified pronunciation lexicon stored in an AWS Region. For more information, see Managing Lexicons.

Request Syntax

```
GET /v1/lexicons/LexiconName HTTP/1.1
```

URI Request Parameters

The request requires the following URI parameters.

Name (p. 142)

Name of the lexicon.

Pattern: `[0-9A-Za-z]{1,20}`

Request Body

The request does not have a request body.

Response Syntax

```
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
   }
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

Lexicon (p. 142)

Lexicon object that provides name and the string content of the lexicon.

Type: Lexicon (p. 152) object

LexiconAttributes (p. 142)

Metadata of the lexicon, including phonetic alphabetic used, language code, lexicon ARN, number of lexemes defined in the lexicon, and size of lexicon in bytes.

Type: LexiconAttributes (p. 153) object

Errors

LexiconNotFoundException

Amazon Polly can't find the specified lexicon. This could be caused by a lexicon that is missing, its name is misspelled or specifying a lexicon that is in a different region.

Verify that the lexicon exists, is in the region (see ListLexicons (p. 144)) and that you spelled its name is spelled correctly. Then try again.

HTTP Status Code: 404

ServiceFailureException

An unknown condition has caused a service failure.

HTTP Status Code: 500

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

ListLexicons

Returns a list of pronunciation lexicons stored in an AWS Region. For more information, see Managing Lexicons.

Request Syntax

```
GET /v1/lexicons?NextToken=NextToken HTTP/1.1
```

URI Request Parameters

The request requires the following URI parameters.

NextToken (p. 144)

An opaque pagination token returned from previous `ListLexicons` operation. If present, indicates where to continue the list of lexicons.

Request Body

The request does not have a request body.

Response Syntax

```
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"
}
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The following data is returned in JSON format by the service.

Lexicons (p. 144)

A list of lexicon names and attributes.

Type: Array of LexiconDescription (p. 155) objects

NextToken (p. 144)

The pagination token to use in the next request to continue the listing of lexicons. `NextToken` is returned only if the response is truncated.

Type: String

Errors

InvalidNextTokenException

The NextToken is invalid. Verify that it's spelled correctly, and then try again.

HTTP Status Code: 400

ServiceFailureException

An unknown condition has caused a service failure.

HTTP Status Code: 500

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

PutLexicon

Stores a pronunciation lexicon in an AWS Region. If a lexicon with the same name already exists in the region, it is overwritten by the new lexicon. Lexicon operations have eventual consistency, therefore, it might take some time before the lexicon is available to the SynthesizeSpeech operation.

For more information, see Managing Lexicons.

Request Syntax

```
PUT /v1/lexicons/LexiconName HTTP/1.1
Content-type: application/json

{
   "Content": "string"
}
```

URI Request Parameters

The request requires the following URI parameters.

Name (p. 146)

Name of the lexicon. The name must follow the regular express format [0-9A-Za-z]{1,20}. That is, the name is a case-sensitive alphanumeric string up to 20 characters long.

Pattern: `[0-9A-Za-z]{1,20}`

Request Body

The request accepts the following data in JSON format.

Content (p. 146)

Content of the PLS lexicon as string data.

Type: String

Required: Yes

Response Syntax

```
HTTP/1.1 200
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response with an empty HTTP body.

Errors

InvalidLexiconException

Amazon Polly can't find the specified lexicon. Verify that the lexicon's name is spelled correctly, and then try again.

HTTP Status Code: 400

LexiconSizeExceededException

The maximum size of the specified lexicon would be exceeded by this operation.

HTTP Status Code: 400

MaxLexemeLengthExceededException

The maximum size of the lexeme would be exceeded by this operation.

HTTP Status Code: 400

MaxLexiconsNumberExceededException

The maximum number of lexicons would be exceeded by this operation.

HTTP Status Code: 400

ServiceFailureException

An unknown condition has caused a service failure.

HTTP Status Code: 500

UnsupportedPlsAlphabetException

The alphabet specified by the lexicon is not a supported alphabet. Valid values are `x-sampa` and `ipa`.

HTTP Status Code: 400

UnsupportedPlsLanguageException

The language specified in the lexicon is unsupported. For a list of supported languages, see Lexicon Attributes.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

SynthesizeSpeech

Synthesizes UTF-8 input, plain text or SSML, to a stream of bytes. SSML input must be valid, well-formed SSML. Some alphabets might not be available with all the voices (for example, Cyrillic might not be read at all by English voices) unless phoneme mapping is used. For more information, see How it Works.

Request Syntax

```
POST /v1/speech HTTP/1.1
Content-type: application/json

{
   "LexiconNames": [ "string" ],
   "OutputFormat": "string",
   "SampleRate": "string",
   "SpeechMarkTypes": [ "string" ],
   "Text": "string",
   "TextType": "string",
   "VoiceId": "string"
}
```

URI Request Parameters

The request does not use any URI parameters.

Request Body

The request accepts the following data in JSON format.

LexiconNames (p. 148)

List of one or more pronunciation lexicon names you want the service to apply during synthesis. Lexicons are applied only if the language of the lexicon is the same as the language of the voice. For information about storing lexicons, see PutLexicon.

Type: Array of strings

Array Members: Maximum number of 5 items.

Pattern: `[0-9A-Za-z]{1,20}`

Required: No

OutputFormat (p. 148)

The format in which the returned output will be encoded. For audio stream, this will be mp3, ogg_vorbis, or pcm. For speech marks, this will be json.

Type: String

Valid Values: `json | mp3 | ogg_vorbis | pcm`

Required: Yes

SampleRate (p. 148)

The audio frequency specified in Hz.

The valid values for `mp3` and `ogg_vorbis` are "8000", "16000", and "22050". The default value is "22050".

Valid values for `pcm` are "8000" and "16000" The default value is "16000".

Type: String

Required: No

SpeechMarkTypes (p. 148)

The type of speech marks returned for the input text.

Type: Array of strings

Array Members: Maximum number of 4 items.

Valid Values: `sentence | ssml | viseme | word`

Required: No

Text (p. 148)

Input text to synthesize. If you specify `ssml` as the `TextType`, follow the SSML format for the input text.

Type: String

Required: Yes

TextType (p. 148)

Specifies whether the input text is plain text or SSML. The default value is plain text. For more information, see Using SSML.

Type: String

Valid Values: `ssml | text`

Required: No

VoiceId (p. 148)

Voice ID to use for the synthesis. You can get a list of available voice IDs by calling the DescribeVoices operation.

Type: String

Valid Values: `Geraint | Gwyneth | Mads | Naja | Hans | Marlene | Nicole | Russell | Amy | Brian | Emma | Raveena | Ivy | Joanna | Joey | Justin | Kendra | Kimberly | Salli | Conchita | Enrique | Miguel | Penelope | Chantal | Celine | Mathieu | Dora | Karl | Carla | Giorgio | Mizuki | Liv | Lotte | Ruben | Ewa | Jacek | Jan | Maja | Ricardo | Vitoria | Cristiano | Ines | Carmen | Maxim | Tatyana | Astrid | Filiz`

Required: Yes

Response Syntax

```
HTTP/1.1 200
Content-Type: ContentType
x-amzn-RequestCharacters: RequestCharacters

AudioStream
```

Response Elements

If the action is successful, the service sends back an HTTP 200 response.

The response returns the following HTTP headers.

ContentType (p. 149)

Specifies the type audio stream. This should reflect the `OutputFormat` parameter in your request.

- If you request `mp3` as the `OutputFormat`, the `ContentType` returned is audio/mpeg.
- If you request `ogg_vorbis` as the `OutputFormat`, the `ContentType` returned is audio/ogg.
- If you request `pcm` as the `OutputFormat`, the `ContentType` returned is audio/pcm in a signed 16-bit, 1 channel (mono), little-endian format.
- If you request `json` as the `OutputFormat`, the `ContentType` returned is audio/json.

RequestCharacters (p. 149)

Number of characters synthesized.

The response returns the following as the HTTP body.

AudioStream (p. 149)

Stream containing the synthesized speech.

Errors

InvalidSampleRateException

The specified sample rate is not valid.

HTTP Status Code: 400

InvalidSsmlException

The SSML you provided is invalid. Verify the SSML syntax, spelling of tags and values, and then try again.

HTTP Status Code: 400

LexiconNotFoundException

Amazon Polly can't find the specified lexicon. This could be caused by a lexicon that is missing, its name is misspelled or specifying a lexicon that is in a different region.

Verify that the lexicon exists, is in the region (see ListLexicons (p. 144)) and that you spelled its name is spelled correctly. Then try again.

HTTP Status Code: 404

MarksNotSupportedForFormatException

Speech marks are not supported for the `OutputFormat` selected. Speech marks are only available for content in `json` format.

HTTP Status Code: 400

ServiceFailureException

An unknown condition has caused a service failure.

HTTP Status Code: 500

SsmlMarksNotSupportedForTextTypeException

SSML speech marks are not supported for plain text-type input.

HTTP Status Code: 400

TextLengthExceededException

The value of the "Text" parameter is longer than the accepted limits. The limit for input text is a maximum of 6000 characters total, of which no more than 3000 can be billed characters. SSML tags are not counted as billed characters.

HTTP Status Code: 400

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS Command Line Interface
- AWS SDK for .NET
- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for JavaScript
- AWS SDK for PHP V3
- AWS SDK for Python
- AWS SDK for Ruby V2

Data Types

The following data types are supported:

- Lexicon (p. 152)
- LexiconAttributes (p. 153)
- LexiconDescription (p. 155)
- Voice (p. 156)

Lexicon

Provides lexicon name and lexicon content in string format. For more information, see Pronunciation Lexicon Specification (PLS) Version 1.0.

Contents

Content

Lexicon content in string format. The content of a lexicon must be in PLS format.

Type: String

Required: No

Name

Name of the lexicon.

Type: String

Pattern: `[0-9A-Za-z]{1,20}`

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2

LexiconAttributes

Contains metadata describing the lexicon such as the number of lexemes, language code, and so on. For more information, see Managing Lexicons.

Contents

Alphabet

Phonetic alphabet used in the lexicon. Valid values are `ipa` and `x-sampa`.

Type: String

Required: No

LanguageCode

Language code that the lexicon applies to. A lexicon with a language code such as "en" would be applied to all English languages (en-GB, en-US, en-AUS, en-WLS, and so on.

Type: String

Valid Values: `cy-GB | da-DK | de-DE | en-AU | en-GB | en-GB-WLS | en-IN | en-US | es-ES | es-US | fr-CA | fr-FR | is-IS | it-IT | ja-JP | nb-NO | nl-NL | pl-PL | pt-BR | pt-PT | ro-RO | ru-RU | sv-SE | tr-TR`

Required: No

LastModified

Date lexicon was last modified (a timestamp value).

Type: Timestamp

Required: No

LexemesCount

Number of lexemes in the lexicon.

Type: Integer

Required: No

LexiconArn

Amazon Resource Name (ARN) of the lexicon.

Type: String

Required: No

Size

Total size of the lexicon, in characters.

Type: Integer

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2

LexiconDescription

Describes the content of the lexicon.

Contents

Attributes

Provides lexicon metadata.

Type: LexiconAttributes (p. 153) object

Required: No

Name

Name of the lexicon.

Type: String

Pattern: `[0-9A-Za-z]{1,20}`

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2

Voice

Description of the voice.

Contents

Gender

Gender of the voice.

Type: String

Valid Values: `Female | Male`

Required: No

Id

Amazon Polly assigned voice ID. This is the ID that you specify when calling the `SynthesizeSpeech` operation.

Type: String

Valid Values: `Geraint | Gwyneth | Mads | Naja | Hans | Marlene | Nicole | Russell | Amy | Brian | Emma | Raveena | Ivy | Joanna | Joey | Justin | Kendra | Kimberly | Salli | Conchita | Enrique | Miguel | Penelope | Chantal | Celine | Mathieu | Dora | Karl | Carla | Giorgio | Mizuki | Liv | Lotte | Ruben | Ewa | Jacek | Jan | Maja | Ricardo | Vitoria | Cristiano | Ines | Carmen | Maxim | Tatyana | Astrid | Filiz`

Required: No

LanguageCode

Language code of the voice.

Type: String

Valid Values: `cy-GB | da-DK | de-DE | en-AU | en-GB | en-GB-WLS | en-IN | en-US | es-ES | es-US | fr-CA | fr-FR | is-IS | it-IT | ja-JP | nb-NO | nl-NL | pl-PL | pt-BR | pt-PT | ro-RO | ru-RU | sv-SE | tr-TR`

Required: No

LanguageName

Human readable name of the language in English.

Type: String

Required: No

Name

Name of the voice (for example, Salli, Kendra, etc.). This provides a human readable voice name that you might display in your application.

Type: String

Required: No

See Also

For more information about using this API in one of the language-specific AWS SDKs, see the following:

- AWS SDK for C++
- AWS SDK for Go
- AWS SDK for Java
- AWS SDK for Ruby V2

Amazon Polly에 대한 인증 및 액세스 제어

Amazon Polly에 액세스하려면 자격 증명이 필요합니다. 이 자격 증명에는 Amazon Polly lexicon 또는 Amazon Elastic Compute Cloud(Amazon EC2) 인스턴스와 같은 AWS 리소스에 액세스할 수 있는 권한이 있습니다. 다음 단원에서는 리소스 액세스를 보호하기 위해 AWS Identity and Access Management(IAM) 및 Amazon Polly 사용법에 대한 세부 정보를 제공합니다.

인증

다음과 같은 자격 증명 유형으로 AWS에 액세스할 수 있습니다.

- AWS 계정 루트 사용자 – AWS 계정을 처음 생성하는 경우에는 전체 AWS 서비스 및 계정 리소스에 대해 완전한 액세스 권한을 지닌 단일 로그인 자격 증명으로 시작합니다. 이 자격 증명은 AWS 계정 루트 사용자라고 하며, 계정을 생성할 때 사용한 이메일 주소와 암호로 로그인하여 액세스합니다. 일상적인 작업, 심지어 관리 작업의 경우에도 루트 사용자를 사용하지 마실 것을 강력히 권장합니다. 대신, IAM 사용자를 처음 생성할 때만 루트 사용자를 사용하는 모범 사례를 준수합니다. 그런 다음 루트 사용자를 안전하게 보관해 두고 몇 가지 계정 및 서비스 관리 작업을 수행할 때만 자격 증명을 사용합니다.
- IAM 사용자 – IAM 사용자는 특정 사용자 지정 권한(예: Amazon Polly에서 a lexicon을 만들 권한)이 있는 AWS 계정 내의 자격 증명입니다. IAM 사용자 이름과 암호를 사용하여 AWS Management 콘솔, AWS 토론 포럼 또는 AWS Support Center와 같은 보안 AWS 웹 페이지에 로그인할 수 있습니다.

  사용자 이름과 암호 외에도 각 사용자에 대해 액세스 키를 생성할 수 있습니다. 여러 SDK 중 하나를 통해 또는 AWS Command Line Interface(CLI)를 사용하여 AWS 서비스에 프로그래밍 방식으로 액세스할 때 이러한 키를 사용할 수 있습니다. SDK 및 CLI 도구는 액세스 키를 사용하여 암호화 방식으로 요청에 서명합니다. AWS 도구를 사용하지 않는 경우 요청에 직접 서명해야 합니다. Amazon Polly supports 는 인바운드 API 요청을 인증하기 위한 프로토콜인 서명 버전 4를 지원합니다. 요청 인증에 대한 자세한 내용은 AWS General Reference의 서명 버전 4 서명 프로세스를 참조하십시오.

- IAM 역할 - IAM 역할은 특정 권한을 가진 계정에 생성할 수 있는 IAM 자격 증명입니다. IAM 사용자와 유사하지만, 특정 개인과 연결되지 않습니다. IAM 역할을 사용하면 AWS 서비스 및 리소스에 액세스하는 데 사용할 수 있는 임시 액세스 키를 얻을 수 있습니다. 임시 자격 증명이 있는 IAM 역할은 다음과 같은 상황에서 유용합니다.

  - 연합된 사용자 액세스 – IAM 사용자를 만드는 대신 AWS Directory Service의 기존 사용자 자격 증명, 엔터프라이즈 사용자 디렉터리 또는 웹 자격 증명 공급자를 사용할 수 있습니다. 이러한 사용자를 연합된 사용자라고 합니다. 자격 증명 공급자를 통해 액세스를 요청하면 AWS가 연합된 사용자에게 역할을 할당합니다. 연합된 사용자에 대한 자세한 내용은 IAM 사용 설명서의 연합된 사용자 및 역할을 참조하십시오.

  - AWS 서비스 액세스 – 계정의 IAM 역할을 사용하여 AWS 서비스에 계정의 리소스에 액세스할 수 있는 권한을 부여할 수 있습니다. 예를 들어 Amazon Redshift에서 사용자 대신 Amazon S3 버킷에 액세스하

도록 허용하는 역할을 만든 후 그 버킷으로부터 데이터를 Amazon Redshift 클러스터로 로드할 수 있습니다. 자세한 내용은 IAM 사용 설명서의 Creating a Role to Delegate Permissions to an AWS Service 단원을 참조하십시오.

- Amazon EC2에서 실행하는 애플리케이션 – EC2 인스턴스에서 실행되고 AWS API 요청을 하는 애플리케이션의 임시 자격 증명을 IAM 역할을 사용하여 관리할 수 있습니다. 이것은 EC2 인스턴스 내에 액세스 키를 저장하는 경우에 바람직한 방법입니다. EC2 인스턴스에 AWS 역할을 할당하고 그 역할을 모든 애플리케이션에서 사용할 수 있도록 하려면 인스턴스에 연결된 인스턴스 프로파일을 만들어야 합니다. 인스턴스 프로파일에는 역할이 포함되어 있으며 EC2 인스턴스에서 실행되는 프로그램이 임시 자격 증명을 얻을 수 있습니다. 자세한 내용은 IAM 사용 설명서의 IAM 역할을 사용하여 Amazon EC2 인스턴스에서 실행되는 애플리케이션에 권한 부여하기 섹션을 참조하십시오.

액세스 제어

요청을 인증하는 데 유효한 자격 증명이 있더라도 권한이 없다면 Amazon Polly 리소스를 생성하거나 액세스할 수 없습니다. 예를 들어 Amazon Polly lexicon를 생성할 권한이 있어야 합니다.

다음 단원에서는 Amazon Polly에 대한 권한을 관리하는 방법에 대해 설명합니다. 먼저 개요를 읽어 보면 도움이 됩니다.

Amazon Polly 리소스에 대한 액세스 권한 관리 개요

모든 AWS 리소스는 AWS 계정의 소유이고, 리소스 생성 또는 리소스 액세스 권한은 권한 정책에 따라 결정됩니다. 계정 관리자는 IAM 자격 증명(즉, 사용자, 그룹, 역할)에 권한 정책을 연결할 수 있고, 일부 서비스(예: AWS Lambda)에서는 리소스에 대한 권한 정책 연결도 지원합니다.

> Note
>
> 계정 관리자 또는 관리자 사용자는 관리자 권한이 있는 사용자입니다. 자세한 정보는 IAM 사용 설명서에서 IAM 모범 사례를 참조하십시오.

권한을 부여하려면 권한을 부여 받을 사용자, 권한 대상이 되는 리소스, 해당 리소스에 허용되는 특정 작업을 결정합니다.

항목

Amazon Polly 리소스 및 작업

Amazon Polly에서 기본 리소스는 a lexicon입니다. 정책에서 Amazon 리소스 이름(ARN)을 사용하여 정책이 적용되는 리소스를 식별합니다.

다음 표에 나와 있는 것처럼 이러한 리소스와 하위 리소스에는 고유한 Amazon 리소스 이름(ARN)이 연결되어 있습니다.

리소스 유형	ARN 형식
어휘	`arn:aws:polly:region:account-id:lexicon/LexiconName`

Amazon Polly은(는) Amazon Polly 리소스를 처리하기 위한 작업을 제공합니다. 사용 가능한 작업 목록은 Amazon Polly Amazon Polly API 참조 (p. 137)를 참조하십시오.

리소스 소유권 이해

AWS 계정은 리소스를 누가 생성했든 상관없이 계정에서 생성된 리소스를 소유합니다. 특히, 리소스 소유자는 리소스 생성 요청을 인증하는 보안 주체 엔터티(즉, 루트 계정, IAM 사용자 또는 IAM 역할)의 AWS 계정입니다. 다음 예에서는 이 계정의 작동 방식을 설명합니다.

- AWS 계정의 루트 계정 자격 증명을 사용하여 a lexicon를 만들면 AWS 계정은 해당 리소스의 소유자가 됩니다(Amazon Polly에서 리소스는 a lexicon).
- AWS 계정에서 IAM 사용자를 만들고 a lexicon를 생성할 수 있는 권한을 사용자에게 부여하면 해당 사용자가 a lexicon를 생성할 수 있습니다. 하지만 lexicon 리소스는 해당 사용자가 속한 AWS 계정이 소유합니다.
- AWS 계정에서 a lexicon를 생성할 수 있는 권한이 있는 IAM 역할을 만들면 해당 역할을 담당할 수 있는 사용자는 누구나 a lexicon를 생성할 수 있습니다. lexicon 리소스는 해당 사용자가 속한 AWS 계정이 소유합니다.

리소스 액세스 관리

권한 정책은 누가 무엇에 액세스 할 수 있는지를 나타냅니다. 다음 단원에서는 권한 정책을 만드는 데 사용 가능한 옵션에 대해 설명합니다.

> **Note**
>
> 이 단원에서는 Amazon Polly의 맥락에서 IAM을 사용하는 방법에 대해 설명하며, IAM 서비스에 대한 자세한 정보는 다루지 않습니다. 전체 IAM 설명서는 IAM이란 무엇인가?(출처: IAM 사용 설명서) 단원을 참조하십시오. IAM 정책 구문과 설명에 대한 자세한 정보는 IAM 사용 설명서의 AWS IAM Policy Reference를 참조하십시오.

IAM 자격 증명에 연결된 정책을 자격 증명 기반 정책(IAM 정책)이라고 하고, 리소스에 연결된 정책을 리소스 기반 정책이라고 합니다. Amazon Polly은 자격 증명 기반 정책을 지원합니다.

항목

- 자격 증명 기반 정책(IAM 정책) (p. 160)
- 리소스 기반 정책 (p. 161)

자격 증명 기반 정책(IAM 정책)

정책을 IAM 자격 증명에 연결할 수 있습니다. 예를 들면,

- 계정 내 사용자 또는 그룹에 권한 정책 연결 – a lexicon과 같은 Amazon Polly 리소스를 생성할 사용자 권한을 부여하기 위해 권한 정책을 특정 사용자 또는 해당 사용자가 속한 그룹에 연결할 수 있습니다.
- 역할에 권한 정책 연결(교차 계정 권한 부여) – 자격 증명 기반 권한 정책을 IAM 역할에 연결하여 교차 계정 권한을 부여할 수 있습니다. 예를 들어, 계정 A의 관리자는 다음과 같이 다른 AWS 계정(예: 계정 B) 또는 AWS 서비스에 교차 계정 권한을 부여할 역할을 생성할 수 있습니다.

1. 계정 A 관리자는 IAM 역할을 생성하고 계정 A의 리소스에 대한 권한을 부여하는 역할에 권한 정책을 연결합니다.
2. 계정 A 관리자는 계정 B를 역할을 수임할 보안 주체로 식별하는 역할에 신뢰 정책을 연결합니다.
3. 계정 B 관리자는 계정 B의 사용자에게 역할을 수임할 권한을 위임할 수 있습니다. 그러면 계정 B의 사용자가 계정 A에서 리소스를 생성하거나 액세스할 수 있습니다. AWS 서비스에 역할 수임 권한을 부여할 경우 신뢰 정책의 보안 주체가 AWS 서비스 보안 주체이기도 합니다.

IAM을 사용하여 권한을 위임하는 방법에 대한 자세한 내용은 IAM 사용 설명서의 액세스 관리를 참조하십시오.

다음은 어휘를 입력하거나 가져오고 현재 사용 가능한 어휘를 나열할 수 있는 권한을 부여하는 정책 예제입니다.

Amazon Polly에서는 리소스 수준에서 작업에 대한 ID 기반 정책을 지원합니다. 따라서 `Resource` 값은 ARN으로 표시됩니다. 예: `arn:aws:polly:us-east-2:account-id:lexicon/*` as the `Resource` 값은 `us-east-2` 리전 내에서 소유한 모든 어휘에 대한 권한을 지정합니다.

```
{
   "Version": "2012-10-17",
   "Statement": [{
      "Sid": "AllowPut-Get-ListActions",
      "Effect": "Allow",
      "Action": [
         "polly:PutLexicon",
         "polly:GetLexicon",
         "polly:ListLexicons"],
      "Resource": "arn:aws:polly:us-east-2:account-id:lexicon/*"
      }
   ]
}
```

Amazon Polly에서 자격 증명 기반 정책을 사용하는 방법에 대한 자세한 내용은 Amazon Polly에 대한 자격 증명 기반 정책(IAM 정책) 사용 (p. 162) 단원을 참조하십시오. 사용자, 그룹, 역할 및 권한에 대한 자세한 내용은 IAM 사용 설명서의 자격 증명(사용자, 그룹 및 역할) 단원을 참조하십시오.

리소스 기반 정책

Amazon S3와 같은 다른 서비스도 리소스 기반 권한 정책을 지원합니다. 예를 들어, 정책을 S3 버킷에 연결하여 해당 버킷에 대한 액세스 권한을 관리할 수 있습니다. Amazon Polly은(는) 리소스 기반 정책을 지원하지 않습니다.

정책 요소 지정: 작업, 효과, 보안 주체

각 Amazon Polly 리소스에 대해 서비스는 일련의 API 작업을 정의합니다. 이러한 API 작업에 대한 권한을 부여하기 위해 Amazon Polly에서는 정책에서 지정할 수 있는 작업을 정의합니다. 일부 API 작업에서는 API 작업을 수행하기 위해 복수의 작업에 대한 권한이 필요할 수 있습니다. 리소스 및 API 작업에 대한 자세한 내용은 Amazon Polly 리소스 및 작업 (p. 159) 및 Amazon Polly API 참조 (p. 137) 단원을 참조하십시오.

다음은 가장 기본적인 정책 요소입니다.

- Resource – Amazon 리소스 이름(ARN)을 사용하여 ID 기반 정책을 적용할 리소스를 식별합니다. 자세한 내용은 Amazon Polly 리소스 및 작업 (p. 159) 단원을 참조하세요.
- 작업 – 작업 키워드를 사용하여 허용 또는 거부할 리소스 작업을 식별합니다. 예를 들어 `polly:PutLexicon`을 사용하여 어휘를 리전에 추가할 수 있습니다.
- Effect – 사용자가 특정 작업을 요청하는 경우의 결과를 지정합니다. 명시적으로 리소스에 대한 액세스 권한을 부여(허용)하지 않는 경우, 액세스는 묵시적으로 거부됩니다. 다른 정책에서는 액세스 권한을 부여하

더라도 리소스에 대한 액세스를 명시적으로 거부하여 사용자가 해당 리소스에 액세스하지 못하게 할 수도 있습니다.

- 보안 주체 – 자격 증명 기반 정책(IAM 정책)에서, 해당 정책이 연결된 사용자를 묵시적인 보안 주체라고 합니다. 리소스 기반 정책의 경우 사용자, 계정, 서비스 또는 권한의 수신자인 기타 엔터티를 지정합니다(리소스 기반 정책에만 해당). Amazon Polly은(는) 리소스 기반 정책을 지원하지 않습니다.

IAM 정책 구문과 설명에 대한 자세한 내용은 IAM 사용 설명서의 AWS IAM 정책 참조 단원을 참조하십시오.

모든 Amazon Polly API 작업과 해당 작업이 적용되는 리소스를 보여주는 목록은 Amazon Polly API 권한: 작업, 권한 및 리소스 참조 (p. 166) 단원을 참조하십시오.

정책에서 조건 지정

권한을 부여할 때 액세스 정책 언어를 사용하여 조건이 적용되는 조건을 지정할 수 있습니다. 예를 들어, 특정 날짜 이후에만 정책을 적용할 수 있습니다. 정책 언어에서의 조건 지정에 관한 자세한 정보는 IAM 사용 설명서의 조건을 참조하세요.

조건을 표시하려면 미리 정의된 조건 키를 사용합니다. Amazon Polly에 특정한 조건 키는 없습니다. 하지만 적절하게 사용할 수 있는 AWS 차원의 조건 키가 있습니다. AWS 차원 키의 전체 목록은 IAM 사용 설명서의 사용 가능한 조건 키 단원을 참조하세요.

Amazon Polly에 대한 자격 증명 기반 정책(IAM 정책) 사용

이 주제에서는 계정 관리자가 권한 정책을 IAM 자격 증명(즉, 사용자, 그룹 및 역할)에 연결하고 이 과정을 통해 Amazon Polly 리소스에서 작업을 수행할 권한을 부여할 수 있는 방법을 보여 주는 자격 증명 기반 정책의 예를 제공합니다.

> **Important**
>
> Amazon Polly 리소스에 대한 액세스 관리를 위해 제공되는 기본 개념과 옵션 설명에 대한 소개 주제 부분을 우선 읽어 보는 것이 좋습니다. 자세한 내용은 Amazon Polly 리소스에 대한 액세스 권한 관리 개요 (p. 159) 단원을 참조하십시오.

항목

다음은 권한 정책의 예입니다.

```
{
   "Version": "2012-10-17",
   "Statement": [{
      "Sid": "AllowGet-Delete-ListActions",
      "Effect": "Allow",
      "Action": [
         "polly:GetLexicon",
         "polly:DeleteLexicon",
         "polly:ListLexicons"],
      "Resource": "*"
      }
   ],
```

```
    "Statement": [{
      "Sid": "NoOverrideMyLexicons",
      "Effect": "Deny",
      "Action": [
         "polly:PutLexicon"],
      "Resource": "arn:aws:polly:us-east-2:123456789012:lexicon/my*"
      }
   ]
}
```

이 정책에는 두 설명문이 있습니다:

- 첫 번째 문은 어휘에 대한 세 Polly 작업(`polly:GetLexicon`, `polly:DeleteLexicon`, `polly:ListLexicons`)에 대한 권한을 부여합니다. 와일드카드 문자(*)를 리소스로 사용하여 모든 리전과 이 계정이 소유한 어휘와 관련한 작업에 대한 보편적인 권한을 부여합니다.
- 두 번째 문은 단일 Polly 작업(`polly:PutLexicon`)에 대한 권한을 명시적으로 거부합니다. 리소스로 표시된 ARN은 `us-east-2` 리전에서 문자 "my"로 시작하는 모든 어휘에 이 권한을 세부적으로 적용합니다.

모든 Amazon Polly API 작업과 해당 작업이 적용되는 리소스를 보여주는 표는 Amazon Polly API 권한: 작업, 권한 및 리소스 참조 (p. 166)를 참조하십시오.

Amazon Polly 콘솔 사용에 필요한 권한

사용자가 Amazon Polly 콘솔로 작업하려면 AWS 계정에서 Amazon Polly 리소스를 설명하도록 허용하는 최소 권한이 있어야 합니다.

최소 필수 권한보다 더 제한적인 IAM 정책을 만들면 콘솔은 해당 IAM 정책에 연결된 사용자에 대해 의도대로 작동하지 않습니다.

AWS CLI 또는 Amazon Polly API만 호출하는 사용자에게 최소 콘솔 권한을 허용할 필요가 없습니다.

Amazon Polly 콘솔을 사용하려면 모든 Amazon Polly API에 권한을 부여해야 합니다. 추가 권한은 필요하지 않습니다. Amazon Polly 콘솔을 사용하는 데 필요한 모든 권한 정책은 다음과 같습니다.

```
}
"Version": "2012-10-17",
   "Statement": [{
      "Sid": "Console-AllowAllPollyActions",
      "Effect": "Allow",
      "Action": [
         "polly:*"],
      "Resource": "*"
      }
   ]
}
```

Amazon Polly에 대한 AWS 관리형(미리 정의된) 정책

AWS는 AWS에서 생성하고 관리하는 독립형 IAM 정책을 제공하여 많은 일반 사용 사례를 처리합니다. 이러한 AWS 관리형 정책은 사용자가 필요한 권한을 조사할 필요가 없도록 일반 사용 사례에 필요한 권한을 부여합니다. 자세한 내용은 IAM 사용 설명서의 AWS 관리형 정책을 참조하십시오.

계정의 사용자에 연결할 수 있는 다음 AWS 관리형 정책은 Amazon Polly에 고유합니다.

- AmazonPollyReadOnlyAccess – 리소스에 대한 읽기 전용 액세스 권한을 부여하고, 어휘 나열, 어휘 불러오기, 사용 가능한 음성 나열, 스피치 합성(합성된 스피치에 어휘 적용 포함) 등을 허용합니다.
- AmazonPollyFullAccess – 리소스와 모든 지원되는 작업에 대한 전체 액세스 권한을 부여합니다.

Note

IAM 콘솔에 로그인하고 이 콘솔에서 특정 정책을 검색하여 이러한 권한 정책을 검토할 수 있습니다.

Amazon Polly 작업 및 리소스에 대한 권한을 허용하는 고유의 사용자 지정 IAM 정책을 생성할 수도 있습니다. 해당 권한이 필요한 IAM 사용자 또는 그룹에 이러한 사용자 지정 정책을 연결할 수 있습니다.

고객 관리형 정책 예

이 단원에서는 다양한 Amazon Polly 작업에 대한 권한을 부여하는 사용자 정책의 예를 제공합니다. 이러한 정책은 AWS SDK 또는 AWS AWS CLI를 사용하는 경우에 유효합니다. 콘솔을 사용할 경우 모든 Amazon Polly API에 권한을 부여해야 합니다. 자세한 내용은 Amazon Polly 콘솔 사용에 필요한 권한 (p. 163) 단원을 참조하십시오.

Note

모든 예는 us-east-2 리전을 사용하며 가상의 계정 ID를 포함합니다.

예제

예제 1: 모든 Amazon Polly 작업 허용

가입한 후(1.1단계: AWS에 가입 (p. 6) 참조) 사용자 생성, 권한 관리를 비롯한 계정 관리 작업을 수행할 수 있는 관리자 사용자를 생성합니다.

Amazon Polly를 사용하는 동안 모든 Amazon Polly 작업에 대한 권한을 가진 사용자(이 사용자를 서비스 관련 관리자로 간주)를 생성할 수 있습니다. 이 사용자에게 다음 권한 정책을 연결할 수 있습니다.

```
{
   "Version": "2012-10-17",
   "Statement": [{
      "Sid": "AllowAllPollyActions",
      "Effect": "Allow",
      "Action": [
         "polly:*"],
      "Resource": "*"
      }
   ]
}
```

예제 2: DeleteLexicon을 제외한 모든 Polly 작업 허용

다음 권한 정책은 모든 리전에서 명시적으로 거부되는 삭제 권한을 포함하여 모든 작업(`DeleteLexicon` 제외)을 수행할 수 있는 권한을 사용자에게 부여합니다.

```
{
   "Version": "2012-10-17",
   "Statement": [{
      "Sid": "AllowAllActions-DenyDelete",
      "Effect": "Allow",
      "Action": [
```

```
            "polly:DescribeVoices",
            "polly:GetLexicon",
            "polly:PutLexicon",
            "polly:SynthesizeSpeech",
            "polly:ListLexicons"],
        "Resource": "*"
        }
        {
        "Sid": "DenyDeleteLexicon",
        "Effect": "Deny",
        "Action": [
            "polly:DeleteLexicon"],
        "Resource": "*"
        }
    ]
}
```

예제 3: DeleteLexicon 허용

다음 권한 정책은 위치한 리전 또는 프로젝트에 상관없이 소유한 모든 어휘를 삭제할 수 있는 권한을 부여합니다.

```
{
  "Version": "2012-10-17",
  "Statement": [{
      "Sid": "AllowDeleteLexicon",
      "Effect": "Allow",
      "Action": [
         "polly:DeleteLexicon"],
      "Resource": "*"
      }
    ]
}
```

예제 4: 지정된 리전에서 어휘 삭제 허용

다음 권한 정책은 단일 리전(이 경우 us-east-2)에서 소유한 모든 프로젝트에서 어휘를 삭제할 수 있는 권한을 부여합니다.

```
{
  "Version": "2012-10-17",
  "Statement": [{
      "Sid": "AllowDeleteSpecifiedRegion",
      "Effect": "Allow",
      "Action": [
         "polly:DeleteLexicon"],
      "Resource": "arn:aws:polly:us-east-2:123456789012:lexicon/*"
      }
    ]
}
```

예제 5: 지정된 어휘에 대한 DeleteLexicon 허용

다음 권한 정책은 특정 리전(이 경우 us-east-2)에서 소유한 특정 어휘(이 경우 myLexicon)를 삭제할 수 있는 권한을 부여합니다.

```
{
  "Version": "2012-10-17",
  "Statement": [{
```

```
      "Sid": "AllowDeleteForSpecifiedLexicon",
      "Effect": "Allow",
      "Action": [
         "polly:DeleteLexicon"],
      "Resource": "arn:aws:polly:us-east-2:123456789012:lexicon/myLexicon"
      }
   ]
}
```

Amazon Polly API 권한: 작업, 권한 및 리소스 참조

액세스 제어 (p. 159)을 설정하고 IAM 자격 증명에 연결할 수 있는 권한 정책(자격 증명 기반 정책)을 작성할 때 다음 목록을 참조로 사용할 수 있습니다. 에는 Amazon Polly API 작업, 부여된 권한으로 수행할 수 있는 작업, 권한 부여된 AWS 리소스가 각각 나열되어 있습니다. 정책의 `Action` 필드에서 작업을 지정하고, 정책의 `Resource` 필드에서 리소스 값을 지정합니다.

Amazon Polly 정책의 AWS 차원 조건 키를 사용하여 조건을 표시할 수 있습니다. AWS 차원 키의 전체 목록은 IAM 사용 설명서의 사용할 수 있는 키를 참조하십시오.

> **Note**
>
> 작업을 지정하려면 `polly` 접두사 다음에 API 작업 이름을 사용합니다(예: `polly:GetLexicon`).

Amazon Polly에서는 리소스 수준에서 작업에 대한 ID 기반 정책을 지원합니다. 따라서 `Resource` 값은 ARN으로 표시됩니다. 예: `arn:aws:polly:`*`us-east-2`*`:`*`account-id`*`:lexicon/*` as the `Resource` 값은 `us-east-2` 리전 내에서 소유한 모든 어휘에 대한 권한을 지정합니다.

Amazon Polly에서는 리소스 수준의 작업에 대한 권한을 지원하지 않으므로, 대부분의 정책에서는 와일드카드 문자(*)를 `Resource` 값으로 지정합니다. 하지만 권한을 특정 리전으로 제한해야 하는 경우 이 와일드카드 문자를 적절한 ARN으로 대체합니다. `arn:aws:polly:`*`region`*`:`*`account-id`*`:lexicon/*`.

Amazon Polly API 및 작업에 필요한 권한

API 작업: DeleteLexicon (p. 138)

필요한 권한(API 작업): `polly:DeleteLexicon`

리소스: `arn:aws:polly:`*`region`*`:`*`account-id`*`:lexicon/`*`LexiconName`*

API 작업: DescribeVoices (p. 140)

필요한 권한(API 작업): `polly:DescribeVoices`

리소스: `arn:aws:polly:`*`region`*`:`*`account-id`*`:lexicon/`*`voice-name`*

API 작업: GetLexicon (p. 142)

필요한 권한(API 작업): `polly:GetLexicon`

리소스: `arn:aws:polly:`*`region`*`:`*`account-id`*`:lexicon/`*`voice-name`*

API 작업: ListLexicons (p. 144)

필요한 권한(API 작업): `polly:ListLexicons`

리소스: `arn:aws:polly:`*`region`*`:`*`account-id`*`:lexicon/*`

API 작업: PutLexicon (p. 146)

필요한 권한(API 작업): `polly:ListLexicons`

리소스: *

API 작업: SynthesizeSpeech (p. 148)

필요한 권한(API 작업): `polly:SynthesizeSpeech`

리소스: `*`

Amazon Polly의 문서 이력

다음 표에서는 본 Amazon Polly 릴리스 문서에 대해 설명합니다.

- 최종 문서 업데이트:2017년 4월 19일

변경 사항	설명	날짜
새로운 서비스 및 가이드	이 문서는 AWS 텍스트 투 스피치 서비스, Amazon Polly 및 Amazon Polly 개발자 안내서의 최초 릴리스입니다.	2016년 11월 30일
새로운 기능 및 확장 설명서	이 문서는 Amazon Polly에 대한 업데이트이며, 새로운 스피치 마크 기능과 SSML 기능 확장을 포함합니다.	2017년 4월 19일
2가지 새로운 음성, 새로운 언어, 리전 확장	이 릴리스는 Amazon Polly 업데이트이며 다음을 포함합니다. • 새로운 음성 2개: Aditi(여성, 인도 영어) 및 Seoyeon(여성, 한국어) • 새로운 언어, 한국어(ko-KR) • Amazon Polly 서비스를 모든 상용 리전으로 확대.	2017년 11월 15일

AWS Glossary

For the latest AWS terminology, see the AWS Glossary in the AWS General Reference.