



開發人員指南

Amazon Polly



Amazon Polly: 開發人員指南

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon 的商標和商業外觀不得用於任何非 Amazon 的產品或服務，也不能以任何可能造成客戶混淆、任何貶低或使 Amazon 名譽受損的方式使用 Amazon 的商標和商業外觀。所有其他非 Amazon 擁有的商標均為其各自擁有者的財產，這些擁有者可能附屬於 Amazon，或與 Amazon 有合作關係，亦或受到 Amazon 贊助。

Table of Contents

什麼是 Amazon Polly ?	1
運作方式	1
優勢	2
您是第一次使用 的新手嗎 ?	2
使用 AWS SDKs	3
開始使用	4
註冊 AWS	4
註冊 AWS 帳戶	4
設定 AWS CLI	5
重新設定 AWS CLI	6
合成語音範例	8
雙向串流	12
比較 SynthesizeSpeech 和 StartSpeechSynthesisStream	12
傳送文字和接收音訊	13
開啟串流	14
傳送文字	14
接收音訊	15
關閉串流	15
程式碼範例	15
Amazon Polly 中的語音	25
可用語音	25
品牌聲音	33
雙語語音	33
有腔調的雙語語音	33
全雙語語音	34
套用新聞播報員語音	35
聆聽語音	37
計時語音速度	37
變更語音速度	38
Amazon Polly 中的語言	40
阿拉伯文 (arb)	44
阿拉伯文 (海灣) (ar-AE)	47
加泰隆尼亞文 (ca-ES)	52
中文 (中文) (yue-CN)	55

中文 (國語) (cmn-CN)	59
捷克文 (cs-CZ)	62
丹麥文 (da-DK)	65
荷蘭文 (比利時) (nl-BE)	68
荷蘭文 (nl-NL)	71
英文 (美國) (en-US)	73
英文 (澳洲) (en-AU)	76
英文 (英國) (en-GB)	79
英文 (印度) (en-IN)	82
英文 (愛爾蘭) (en-IE)	84
英文 (紐西蘭) (en-NZ)	87
英文 (新加坡) (en-SG)	92
英文 (南非) (en-ZA)	94
英文 (威爾斯) (en-GB-WLS)	98
芬蘭文 (fi-FI)	101
法文 (fr-FR)	104
法文 (比利時) (fr-BE)	107
法文 (加拿大) (fr-CA)	109
德文 (de-DE)	111
德文 (奧地利) (de-AT)	114
德文 (瑞士標準) (de-CH)	117
印地語 (hi-IN)	120
冰島文 (is-IS)	123
義大利文 (it-IT)	126
日文 (ja-JP)	128
韓文 (ko-KR)	131
挪威文 (nb-NO)	133
波蘭文 (pl-PL)	135
葡萄牙文 (pt-PT)	138
葡萄牙文 (巴西) (pt-BR)	140
羅馬尼亞文 (ro-RO)	142
俄文 (ru-RU)	144
西班牙文 (es-ES)	147
西班牙文 (墨西哥) (es-MX)	149
西班牙文 (美國) (es-US)	151
瑞典文 (sv-SE)	153

土耳其文 (tr-TR)	156
威爾斯文 (cy-GB)	159
語音引擎	163
生成式引擎	163
可用的生成語音	164
功能和區域相容性	166
長型引擎	167
可用的長格式語音	168
功能和區域相容性	168
神經引擎	169
可用的神經語音	169
功能和區域相容性	173
標準引擎	174
可用的標準語音	174
功能和區域相容性	177
選擇語音引擎	179
語音標記	180
語音標記類型	180
Visemes 和 Amazon Polly	181
語音標記輸出	181
請求語音標記	182
不含 SSML 的語音標記範例	184
使用 SSML 的語音標記範例	185
使用 SSML	187
保留字元	188
在主控台上使用 SSML	190
搭配 Synthesize-Speech 命令使用 SSML	191
合成 SSML 增強文件	192
支援的 SSML 標籤	193
識別 SSML 增強的文字	195
新增暫停	195
強調單字	196
為特定單字指定另一種語言	197
在文字中放置自訂標籤	198
在段落之間新增暫停	199
使用音標發音	199

控制音量、說話速率和音調	201
設定合成語音的最長持續時間	203
在句子之間新增暫停	207
控制如何說出特殊類型的單字	207
發音縮寫和縮寫	210
透過指定語音部分來改善發音	211
新增呼吸聲	212
新聞播報員說話風格	216
新增動態範圍壓縮	217
柔和地說話	219
控制波紋	219
輕聲低語	221
管理語彙	222
使用多個語彙	223
上傳語彙	224
套用語彙（合成語音）	229
在主控台上篩選語彙清單	232
在主控台下載語彙	234
刪除語彙	235
長音訊檔案	237
設定非同步合成的 IAM 政策	238
建立長時間音訊檔案	238
配額	242
支援的 區域	242
配額和調節速率	243
並行請求	244
緩解限流的最佳實務	244
發音語彙	244
SynthesizeSpeech API 操作	245
SpeechSynthesisTask API 操作	245
語音合成標記語言 (SSML)	246
範例程式碼和應用程式	247
Java 範例	247
DeleteLexicon	248
DescribeVoices	250
GetLexicon	251

ListLexicons	253
PutLexicon	254
StartSpeechSynthesisTask	256
語音標記	260
SynthesizeSpeech	263
Python 範例	264
DeleteLexicon	265
GetLexicon	266
ListLexicon	267
PutLexicon	268
StartSpeechSynthesisTask	269
SynthesizeSpeech	270
Java 範例	270
Python 範例	275
Python 範例：index.html	277
Python 範例：https://server.py	281
iOS 範例	288
Android 範例	290
程式碼範例	293
基本概念	293
動作	294
案例	344
將文字轉換為語音然後返回文字	345
建立 lip-sync 應用程式	345
建立應用程式以分析客戶意見回饋	346
text-to-speech 合成入門	352
安全	360
資料保護	360
靜態加密	361
傳輸中加密	361
網際網路流量隱私權	361
身分和存取權管理	361
目標對象	362
使用身分驗證	362
使用政策管理存取權	363
Amazon Polly 如何與 IAM 搭配使用	364

身分型政策範例	370
Amazon Polly API 許可參考	376
疑難排解	377
記錄和監控	379
合規驗證	379
恢復能力	380
基礎設施安全性	380
安全最佳實務	380
使用介面 VPC 端點	380
可用性	381
為 Amazon Polly 建立 VPC 端點	381
測試 VPC 與 Amazon Polly 之間的連線	381
控制對 Amazon Polly 端點的存取	382
VPC 內容金鑰支援	383
使用 記錄 Amazon Polly API 呼叫 AWS CloudTrail	384
CloudTrail 中的 Amazon Polly 資訊	384
範例：Amazon Polly 日誌檔案項目	385
CloudWatch 整合	387
取得 CloudWatch 指標（主控台）	387
在上取得 CloudWatch 指標 AWS CLI	387
Amazon Polly 指標	388
Amazon Polly 指標的維度	389
API 參考	390
動作	390
DeleteLexicon	391
DescribeVoices	393
GetLexicon	397
GetSpeechSynthesisTask	400
ListLexicons	403
ListSpeechSynthesisTasks	406
PutLexicon	409
StartSpeechSynthesisStream	412
StartSpeechSynthesisTask	418
SynthesizeSpeech	426
資料類型	432
AudioEvent	433

CloseStreamEvent	434
FlushStreamConfiguration	435
Lexicon	436
LexiconAttributes	437
LexiconDescription	439
StartSpeechSynthesisStreamActionStream	440
StartSpeechSynthesisStreamEventStream	441
StreamClosedEvent	443
SynthesisTask	444
TextEvent	449
ThrottlingReason	450
ValidationExceptionField	451
Voice	452
常見錯誤類型	454
常見參數	456
文件歷史記錄	459
.....	cdlxxii

什麼是 Amazon Polly？

Amazon Polly 是一種雲端服務，可將文字轉換為逼真的語音。您可以使用 Amazon Polly 開發可提高參與度和可存取性的應用程式。Amazon Polly 支援多種語言，並包含各種栩栩如生的語音。使用 Amazon Polly，您可以建置語音應用程式，在多個位置運作，並為客戶使用理想的語音。此外，您只需為合成的文字付費。您也可以快取和重播 Amazon Polly 產生的語音，無需額外費用。

Amazon Polly 提供許多語音選項，包括生成式、長格式、神經和標準text-to-speech(TTS) 選項。這些語音使用新的機器學習技術，在語音品質方面提供突破性的改善，盡可能提供最自然且人性化text-to-speech語音。神經 TTS 技術也支援新聞播報員說話風格，專為新聞敘述使用案例量身打造。

Amazon Polly 的常見使用案例包括但不限於：行動應用程式，例如新聞閱讀器、遊戲、eLearning 平台、視覺障礙者的可存取性應用程式，以及快速成長的物聯網 (IoT) 區段。

Amazon Polly 已通過認證，可與 HIPAA (1996 年健康保險流通與責任法案) 和支付卡產業資料安全標準 (PCI DSS) 的管制工作負載搭配使用。

Amazon Polly 的運作方式

Amazon Polly 會將輸入文字轉換為逼真的語音。若要使用 Amazon Polly 語音，請選擇[語音引擎](#)、呼叫語音合成方法、提供您要合成的文字，然後指定音訊輸出格式。然後，Amazon Polly 會將提供的文字合成為高品質的語音音訊串流。

- 輸入文字 – 提供您要合成的文字，Amazon Polly 會傳回音訊串流。您可以純文字或以語音合成標記語言 (SSML) 格式提供輸入。您可以使用 SSML 控制語音的各方面，例如發音、音量、音調和語音速率。如需詳細資訊，請參閱[從 SSML 文件產生語音](#)。
- 可用語音 – Amazon Polly 提供語言和各種語音的產品組合，包括雙語語音（適用於英文和印地文）。對於大多數語言而言，您可以從多種語音中選擇，包括男性與女性。啟動語音合成任務時，您可以指定語音 ID，然後 Amazon Polly 使用此語音將文字轉換為語音。Amazon Polly 不是翻譯服務，合成的語音與文字使用相同的語言。以數字表示的數字（例如 53，而不是 50-3）是以語音的語言合成，而不是文字。如需詳細資訊，請參閱[Amazon Polly 中的語音](#)。
- 輸出格式 – Amazon Polly 可以多種格式傳遞合成語音。您可以選擇符合您需求的音訊格式。例如，您可以請求 MP3 或 Ogg Vorbis 格式的語音，在 Web 和行動應用程式中使用。或者，您可以請求 PCM 輸出格式，以供 AWS IoT 裝置和電話通訊解決方案使用。對於電話語音應用程式，您可以使用 Mu-law 或 A-law 格式。

 Note

若要在瀏覽器中聆聽 Amazon Polly 語音範例，請參閱 [Amazon Polly 產品概觀](#)。

優勢

使用 Amazon Polly 的一些優點包括：

- 高品質 – Amazon Polly 提供高效能生成式、長格式、神經和高品質的text-to-speech(TTS) 語音。這些技術以高發音準確性合成自然語音（包括縮寫、縮寫擴展、日期/時間解釋和同義詞歧義）。
- 低延遲 – Amazon Polly 實現快速回應，這使得它成為對話系統等低延遲使用案例的可行選項。
- 支援大量語言和語音組合 – Amazon Polly 支援數十種語音和語言，為大多數語言提供男性和女性語音選項。隨著我們將更多神經語音上線，這個數字也會持續增加。美式英文語音 Matthew 和 Joanna 也可以使用神經新聞播報員說話風格，與您聽到的專業新聞主播的聲音類似。
- 經濟實惠 – Amazon Polly pay-per-use模型表示沒有設定成本。從小型開始，隨著您的應用程式成長而擴展。
- 雲端型解決方案 – 裝置內 TTS 解決方案需要大量的運算資源，特別是 CPU 電源、RAM 和磁碟空間。這些可能會導致平板電腦、智慧型手機等裝置的開發成本和耗電量更高。相反地，在中完成的 TTS 轉換可 AWS 雲端大幅降低本機資源需求。這可支援所有可用語言和具有卓越品質的語音。此外，語音改善功能可立即供所有最終使用者使用，且裝置不需要額外的更新。

 Note

若要在瀏覽器中聆聽 Amazon Polly 語音範例，請參閱 [Amazon Polly 產品概觀](#)。

您是第一次使用的新手嗎？

如果您是第一次使用 Amazon Polly，建議您依照列出的順序閱讀下列各節：

1. [Amazon Polly 的運作方式](#) – 本節介紹您可以使用的各種 Amazon Polly 輸入和選項，以建立簡單的體驗。
2. [Amazon Polly 入門](#) – 在本節中，您會設定帳戶並測試 Amazon Polly 語音合成。
3. [Amazon Polly 的範例程式碼和應用程式](#) – 本節提供可用於探索 Amazon Polly 的其他範例。

搭配 SDK 使用 Amazon Polly AWS

AWS 軟體開發套件 (SDKs) 適用於許多熱門的程式設計語言。每個 SDK 都提供 API、程式碼範例和說明文件，讓開發人員能夠更輕鬆地以偏好的語言建置應用程式。

SDK 文件	代碼範例
適用於 C++ 的 AWS SDK	適用於 C++ 的 AWS SDK 程式碼範例
AWS CLI	AWS CLI 程式碼範例
適用於 Go 的 AWS SDK	適用於 Go 的 AWS SDK 程式碼範例
適用於 Java 的 AWS SDK	適用於 Java 的 AWS SDK 程式碼範例
適用於 JavaScript 的 AWS SDK	適用於 JavaScript 的 AWS SDK 程式碼範例
適用於 Kotlin 的 AWS SDK	適用於 Kotlin 的 AWS SDK 程式碼範例
適用於 .NET 的 AWS SDK	適用於 .NET 的 AWS SDK 程式碼範例
適用於 PHP 的 AWS SDK	適用於 PHP 的 AWS SDK 程式碼範例
AWS Tools for PowerShell	AWS Tools for PowerShell 程式碼範例
適用於 Python (Boto3) 的 AWS SDK	適用於 Python (Boto3) 的 AWS SDK 程式碼範例
適用於 Ruby 的 AWS SDK	適用於 Ruby 的 AWS SDK 程式碼範例
適用於 Rust 的 AWS SDK	適用於 Rust 的 AWS SDK 程式碼範例
適用於 SAP ABAP 的 AWS SDK	適用於 SAP ABAP 的 AWS SDK 程式碼範例
適用於 Swift 的 AWS SDK	適用於 Swift 的 AWS SDK 程式碼範例

可用性範例

找不到所需的內容嗎？請使用本頁面底部的提供意見回饋連結申請程式碼範例。

Amazon Polly 入門

Amazon Polly 提供數個 API 操作，您可以輕鬆地與現有應用程式整合。如需支援的操作清單，請參閱 [動作](#)。

您可以在 Amazon Polly 主控台和 [上](#) 執行幾乎所有相同的操作 AWS CLI。不過，您無法在 [上](#) 接聽合成語音 AWS CLI。若要在 [上](#) 使用音訊 AWS CLI，請將文字儲存到 檔案。然後在您選擇的音訊應用程式中開啟 檔案。

您可以使用下列任一選項：

- AWS SDKs – 使用 SDKs，您向 Amazon Polly 提出的請求會使用您提供的登入資料自動簽署和驗證。這是建置您的應用程式的建議選擇。
- AWS CLI – 您可以使用 AWS CLI 來使用 Amazon Polly，而無需撰寫任何程式碼。

首次使用 Amazon Polly 之前，您必須先註冊 AWS。當您註冊 Amazon Web Services (AWS) 時，AWS 您的帳戶會自動註冊所有 服務 AWS，包括 Amazon Polly。您只需為所使用的服務和資源付費。如果您是新 AWS 客戶，可免費開始使用 Amazon Polly。如需更多詳細資訊，請參閱 [AWS 免費用量方案](#)。

下列各節說明如何開始使用 Amazon Polly。

主題

- [註冊 AWS](#)
- [設定 AWS CLI](#)
- [重新設定 AWS CLI](#)

註冊 AWS

您必須註冊，才能使用任何 AWS 服務，包括 Amazon Polly AWS。

註冊 AWS 帳戶

若要開始使用 AWS，您需要 AWS 帳戶。如需建立 的相關資訊 AWS 帳戶，請參閱《AWS 帳戶管理參考指南》中的 [入門 AWS 帳戶](#)。

如需 IAM 的詳細資訊，請參閱下列各項：

- [AWS Identity and Access Management \(IAM\)](#)
- [IAM 入門](#)
- [IAM 使用者指南](#)

 Note

請記下 AWS 您的帳戶 ID。在接下來的步驟中，您將需要它。

設定 AWS CLI

請依照下列步驟下載並設定 AWS CLI 以使用 Amazon Polly。

若要設定 AWS Command Line Interface

1. 下載和設定 AWS CLI。如需說明，請參閱《AWS Command Line Interface 使用者指南》中的下列主題：
 - [使用 進行設定 AWS Command Line Interface](#)
 - [設定 AWS Command Line Interface](#)
2. 在 AWS CLI AWS Config 檔案中為管理員使用者新增具名設定檔。您可以在執行 AWS CLI 命令時使用此設定檔。如需具名描述檔的詳細資訊，請參閱《AWS Command Line Interface 使用者指南》中的[具名描述檔](#)。

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

如需可用 AWS 區域和 Amazon Polly 支援區域的清單，請參閱 中的 [區域和端點](#) Amazon Web Services 一般參考。

 Note

如果您使用的是您在設定時指定的 Amazon Polly 支援的區域 AWS CLI，請從 AWS CLI 程式碼範例中省略以下幾行。

```
--region aws-region
```

3. 在命令提示字元中輸入以下說明命令以驗證設定。

```
aws help
```

有效 AWS 命令的清單應該會出現在 AWS CLI 視窗中。

重新設定 AWS CLI

如果您先前已下載並設定 AWS CLI，除非您重新設定，否則 Amazon Polly 可能無法使用 AWS CLI。下列程序會檢查這是否為必要。

從 重新啟用 Amazon Polly AWS CLI

1. 在命令 AWS CLI 提示中輸入下列說明命令，以驗證 Amazon Polly 的可用性。

```
aws polly help
```

如果您看到 Amazon Polly 的描述，且有效命令的清單出現在 AWS CLI 視窗中，則可以立即從使用 AWS CLI Amazon Polly。在此情況下，您可以略過此程序的剩餘部分。如果未顯示此項，請繼續步驟 2。

2. 使用下列兩個選項之一來啟用 Amazon Polly：

- a. 解除安裝並重新安裝 AWS CLI。

如需說明，請參閱AWS Command Line Interface 《使用者指南》[中的安裝 AWS Command Line Interface](#)。

或

- b. 下載檔案 [service-2.json](#)。

在命令提示中，執行下列命令。

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

3. 重新驗證 Amazon Polly 的可用性。

```
aws polly help
```

Amazon Polly 的描述應該可見。

使用 Amazon Polly 合成語音範例

此頁面顯示使用 Python 在主控台、AWS CLI 和 中執行的簡短語音合成範例。此範例會從純文字而非 SSML 執行語音合成。

Console

在主控台上合成語音

1. 登入 AWS 管理主控台，並在 <https://console.aws.amazon.com/polly/>：// 開啟 Amazon Polly 主控台。
2. 選擇 Text-to-Speech (文字轉換語音) 索引標籤。文字欄位將載入範例文字，讓您可以快速試用 Amazon Polly。
3. 關閉 SSML。
4. 輸入此文字或將其貼到輸入方塊。

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

5. 在引擎下，選擇生成式、長形、神經或標準。
6. 選擇語言和 AWS 區域，然後選擇語音。（如果您為引擎選取神經，則只有支援 NTTS 的語言和語音可用。所有標準和長形語音都會停用。）
7. 若要立即接聽語音，請選擇接聽。
8. 若要將語音儲存至檔案，請執行以下其中一項：
 - a. 選擇 Download (下載)。
 - b. 若要變更為不同的檔案格式，請展開其他設定、開啟語音檔案格式設定、選擇您想要的檔案格式（例如 MP3、OGG、PCM、Mu-law 或 A-law），然後選擇下載。

AWS CLI

在本練習中，您可以透過傳遞輸入文字來呼叫 SynthesizeSpeech 操作。您可以將產生的音訊儲存為檔案並驗證其內容。

1. 執行 synthesize-speech AWS CLI 命令，將範例文字合成到音訊檔案 (hello.mp3)。

下列 AWS CLI 範例已針對 Unix、Linux 和 macOS 格式化。對於 Windows，將每一行尾端的反斜線 (\) Unix 接續字元取代為插入符號 (^)，並在輸入文字周圍使用完整引號 (") 取代內部標籤的單引號 (')。

```
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
```

在呼叫 `synthesize-speech`，您可以提供範例文字，以您選擇的語音合成。您必須提供語音 ID（在下列步驟中說明）和輸出格式。命令會將產生的音訊儲存至 `hello.mp3` 檔案。除了 MP3 檔案，此操作會將以下輸出傳送到主控台。

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

2. 播放產生的 `hello.mp3` 檔案，以驗證合成的語音。

Python

若要測試 Python 範例程式碼，您需要適用於 Python (Boto) 的 AWS SDK。如需相關指示，請參閱[適用於 Python \(Boto3\) 的 AWS SDK](#)。

此範例中的 Python 程式碼會執行下列動作：

- 叫用適用於 Python (Boto) 的 AWS SDK 將 `SynthesizeSpeech` 請求傳送至 Amazon Polly（透過提供一些文字做為輸入）。
- 存取回應中產生的音訊串流，並將音訊儲存至本機磁碟上的檔案 (`speech.mp3`)。
- 使用本機系統的預設音訊播放器播放音訊檔案。

將程式碼儲存至檔案 (`example.py`) 並執行。

```
"""Getting Started Example for Python 2.7+/3.3+"""  
from boto3 import Session
```

```
from botocore.exceptions import BotoCoreError, ClientError
from contextlib import closing
import os
import sys
import subprocess
from tempfile import gettempdir

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

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

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

        try:
            # Open a file for writing the output as a binary stream
            with open(output, "wb") as file:
                file.write(stream.read())
        except IOError as error:
            # Could not write to file, exit gracefully
            print(error)
            sys.exit(-1)
else:
    # The response didn't contain audio data, exit gracefully
    print("Could not stream audio")
    sys.exit(-1)
```

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

如需深入範例的詳細資訊，請參閱下列主題：

- [在主控台上使用 SSML](#)
- [套用語彙（合成語音）](#)
- [Amazon Polly 的範例程式碼和應用程式](#)

使用雙向串流合成語音

Amazon Polly 提供的 `StartSpeechSynthesisStream` 操作會建立 HTTP/2 連線，並在您的應用程式和服務之間進行雙向通訊。合成音訊轉返時，文字會從應用程式流向 Amazon Polly。您可以在可用時傳送文字，Amazon Polly 會在合成時傳回音訊，而不會等待另一個邊完成。

當文字逐漸產生而非一次全部時，這很有用。例如，由 Amazon Bedrock 上基礎模型提供支援的客戶服務聊天機器人會透過權杖產生其回應權杖。透過雙向串流，您的應用程式可以在模型產生每個文字區塊時轉送至 Amazon Polly，並在模型仍在產生其餘回應時開始播放音訊回來電者。

此操作需要生成引擎和支援 HTTP/2 事件串流的 AWS SDK。音訊會以一系列區塊的形式到達，您的應用程式會累積到完整的音訊輸出。此操作不支援語音標記。

Note

不支援 AWS CLI (v1 和 v2)、AWS Tools for PowerShell (v4 和 v5)、Python 和 .NET v3。您可以使用雙向串流 API 搭配下列 SDKs：AWS SDK for Java 2.x、JavaScript v3、.NET v4、C++、Go v2、Kotlin、PHP v3、Ruy v3、Rust 和 Swift。

主題

- [比較 `SynthesizeSpeech` 和 `StartSpeechSynthesisStream`](#)
- [傳送文字和接收音訊](#)
- [程式碼範例](#)

比較 `SynthesizeSpeech` 和 `StartSpeechSynthesisStream`

[SynthesizeSpeech](#) 是一種請求回應操作。您可以在單一請求中提供完整文字，並在單一回應中接收完整合成的音訊。它支援所有引擎（標準、神經、長形、生成式）、包括語音標記在內的所有輸出格式，每個請求的文字限制為 6,000 個字元（其中不超過 3,000 個字元）。回應會在第一個位元組可用時立即串流音訊。當您有所有可用的文字時，請使用此操作。

[StartSpeechSynthesisStream](#) 是一種雙向串流操作。它會開啟 HTTP/2 連線，您在其中遞增傳送文字，並在合成時接收音訊。由於文字會持續串流，因此沒有每個請求的文字限制。它需要生成式引擎，且不支援語音標記。當文字增量到達，且您希望音訊輸出在所有輸入可用之前開始，請使用此操作。常見案例包括：

- 對話式 AI 和語音助理。大型語言模型會以小區塊（權杖）產生回應文字。將每個文字區塊轉送到 Amazon Polly，讓使用者在模型仍在產生時聽到語音。
- 即時轉譯。轉譯系統會依區段產生翻譯的文字區段。串流每個區段以進行合成，無需等待完整轉譯完成。
- 超過 SynthesizeSpeech 限制的長格式內容。超過 6,000 個字元的文字可以持續串流，而無需分割為多個請求或管理區塊邊界。

SynthesizeSpeech 和 StartSpeechSynthesisStream 的比較

面向	SynthesizeSpeech	StartSpeechSynthesisStream
通訊協定	請求-回應	雙向事件串流 (HTTP/2)
文字交付	請求內文中的全文	透過 TextEvent 訊息串流輸入文字
音訊交付	透過 HTTP 回應內文串流音訊回應	透過 AudioEvent 訊息串流音訊回應
引擎支援	標準、神經、長形、生成式	僅限生成式
SSML 支援	是（所有引擎； 支援的標籤因引擎而異 ）	是（ 僅限生成引擎標籤 ）
語彙	是	是
語音標記	是	否
文字限制	每個請求總計 6,000 個字元 (3,000 個計費)	每個 TextEvent 總計 6,000 個字元 (3,000 個計費)
AWS CLI 支援	是	否（雙向串流需要 SDK）

傳送文字和接收音訊

雙向串流工作階段涉及開啟連線、同時傳送文字和接收音訊，然後在輸入完成時關閉串流。下列各節詳細說明每個階段。

開啟串流

您的應用程式會透過 SDK 呼叫 [StartSpeechSynthesisStream](#) 操作，指定合成參數 (Engine、OutputFormat、VoiceId 和選用的 LanguageCode、LexiconNames、)SampleRate。開發套件會建立 HTTP/2 連線，且雙向串流已準備好接受輸入事件。

傳送文字

用戶端會在輸入串流上傳送一或多個 [TextEvent](#) 訊息。每個事件都可以在文字可用時立即傳送，而無需等待完整輸入就緒。文字事件不需要符合句子或標點符號邊界。Amazon Polly 會在內部重新組合文字，並產生自然聲音語音，無論輸入在事件中分割的方式為何。

Note

使用 [SSML](#) 時，每個 SSML 文件都必須在單一 `TextEvent` 中獨立。`TextEvent` 您無法跨多個事件分割 SSML 標籤。不過，您可以在相同的串流中混合純文字事件和 SSML 事件。

串流受到下列時間限制：

- 串流持續時間上限：10 分鐘。無論活動為何，Amazon Polly 會在 10 分鐘後關閉串流。如果您的內容需要更多時間，請為剩餘的文字開啟新的串流。
- 連續事件之間的閒置逾時：5 秒。如果 5 秒內沒有傳送輸入事件，Amazon Polly 會關閉串流。如果您的文字來源暫停超過 5 秒，請傳送 `TextEvent` 具有空白或空格字串的保持連線，以防止逾時。

強制合成緩衝文字與排清

根據預設，Amazon Polly 會根據自然語言界限決定何時合成緩衝文字。這會產生最佳的音訊品質，但表示傳送後可能不會立即傳回音訊 `TextEvent`。

排清可讓您控制合成發生的時間。當您排清時，Amazon Polly 會立即合成到目前為止緩衝的所有文字，無論文字是否在自然界限結束。當您的文字來源在邏輯區段之間暫停，而且您想要為到目前為止傳送的内容交付音訊時，這會很有用。

若要排清，請在 `true` 上將 [FlushStreamConfiguration](#).Force 參數設定為 `TextEvent`。您也可以傳送具有排清旗標的空白來觸發合成 `TextEvent`，而不新增新內容。

排清是權衡。Force 設定為true中音可能會影響發音和語調，因為合成器缺乏以下內容的內容。為了獲得最佳結果，允許 Amazon Polly 盡可能緩衝至自然界限，並只在延遲需求需要時強制合成。

接收音訊

當 Amazon Polly 合成文字時，它會傳回輸出串流上的 [AudioEvent](#) 訊息。每個事件都包含音訊資料區塊。您的應用程式必須累積這些區塊（例如，將它們依序寫入檔案或音訊緩衝區），才能產生完整的音訊輸出。當您仍在傳送文字事件時，音訊事件可能會抵達。

關閉串流

傳送所有輸入文字後，用戶端會傳送 [CloseStreamEvent](#)。Amazon Polly 完成處理任何剩餘的緩衝文字，傳送最終音訊事件，並傳回包含合成字元總數的 [StreamClosedEvent](#)。一律傳送 [CloseStreamEvent](#) 而不是依賴排清來結束串流。關閉可確保合成並傳回所有緩衝文字。

如需請求參數、事件類型和錯誤的完整詳細資訊，請參閱 [StartSpeechSynthesisStream API 參考](#)。

程式碼範例

下列範例示範完整的雙向串流工作階段。程式會建立非同步 Amazon Polly 用戶端、開啟以生成引擎和 MP3 輸出設定的串流、以一系列 [TextEvent](#) 訊息的形式傳送文字、將傳回的 [AudioEvent](#) 區塊累積到輸出檔案中，並使用 [關閉串流CloseStreamEvent](#)。由於輸入和輸出同時發生，音訊資料會在傳送所有文字之前開始抵達。

Java

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

import java.io.FileOutputStream;
import java.io.OutputStream;
import java.util.concurrent.CompletableFuture;
import java.util.concurrent.Semaphore;
import org.reactivestreams.Publisher;
import org.reactivestreams.Subscription;

public class BidirectionalStreamExample {

    public static void main(String[] args) throws Exception {
        try (PollyAsyncClient pollyClient = PollyAsyncClient.builder()
```

```

        .region(Region.US_EAST_1)
        .build()) {

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

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

        StartSpeechSynthesisStreamResponseHandler responseHandler =
            StartSpeechSynthesisStreamResponseHandler.builder()

                .subscriber(StartSpeechSynthesisStreamResponseHandler.Visitor.builder()
                    .onAudioEvent(audioEvent -> {
                        try {
                            byte[] audio =
audioEvent.audioChunk().asByteArray();
                            System.out.println("Received AudioEvent: " +
audio.length + " bytes");

                            audioOutput.write(audio);
                        } catch (Exception e) {
                            throw new RuntimeException(e);
                        }
                    })
                    .onStreamClosedEvent(closedEvent -> {
                        System.out.println("Stream closed. Characters
synthesized: "
                                + closedEvent.requestCharacters());
                    })
                    .onDefault(event -> {})
                    .build())
                .onError(error -> {
                    System.err.println("Stream error: " +
error.getMessage());
                })
                .build();

        String[] textChunks = {
            "The weather forecast for today shows clear skies ",
            "with temperatures reaching twenty five degrees. ",

```

```

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

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

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

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

        {
            new Thread(() -> {
                for (String chunk : textChunks) {
                    try { permits.acquire(); } catch (InterruptedException
e) { return; }

                    if (cancelled) return;
                    System.out.println("Sending TextEvent: " +
chunk.trim());

                    subscriber.onNext(StartSpeechSynthesisStreamActionStream.textEventBuilder()
                        .text(chunk).textType(TextType.TEXT).build());
                    // Simulate delay between chunks (e.g. waiting for LLM
tokens)

                    try { Thread.sleep(300); } catch (InterruptedException
e) { return; }

                }
                if (!cancelled) {
                    subscriber.onNext(StartSpeechSynthesisStreamActionStream
                        .closeStreamEventBuilder().build());
                    subscriber.onComplete();
                }
            });
        }
    });
}

```

```

        }
    }).start();
}
});
};

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

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

```

JavaScript

```

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

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

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

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

async function* createInputEvents() {
    for (const chunk of textChunks) {
        console.log(`Sending TextEvent: ${chunk.trim()}`);
        yield {
            TextEvent: {
                Text: chunk,
                TextType: "text",
            },
        },
    }
}

```

```

    };
    // Simulate delay between chunks (e.g. waiting for LLM tokens)
    await sleep(300);
  }

  yield { CloseStreamEvent: {} };
}

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

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

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

  outputStream.end();
}

synthesizeStream().catch(console.error);

```

Go

```

package main

import (

```

```
"context"
"fmt"
"os"
"strings"
"time"

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

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

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

    client := polly.NewFromConfig(cfg)

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

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

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

```

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

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

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

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

```

```

    fmt.Printf("Stream closed. Characters synthesized: %d\n",
v.Value.RequestCharacters)
}
}

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

```

Rust

```

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

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

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

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

    let input_stream = async_stream::stream! {

```

```

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

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

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

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

```

```
}  
  }  
    }  
      _ => {}
```

Amazon Polly 中的語音

Amazon Polly 為各種語言提供數十種逼真的語音和支援。每一個語音都是透過說母語的使用者建立的，所以每個語音之間都會有變化，即使是相同語言內的語音也一樣。您也可以使用 [AWS 管理主控台](#) 來測試每個語音。對於大多數語言，至少會有一個男性和一個女性聲音，通常每個聲音都不只一個。少數語言只有單一語音。

這份清單包含的多種語言的語音庫存，會持續更新以包含其他的選項。若要建議新語言或語音，請在此頁面提供意見回饋。遺憾的是，我們無法在特定新語言的計劃發佈前對它們進行評論。

Note

若要在瀏覽器中聆聽 Amazon Polly 語音範例，請參閱 [Amazon Polly 產品概觀](#)。

主題

- [可用語音](#)
- [雙語語音](#)
- [套用新聞播報員語音](#)
- [聆聽語音](#)
- [計時語音速度](#)
- [變更語音速度](#)

可用語音

Amazon Polly 以多種語言提供各種逼真的語音，以合成文字語音。下表顯示 Amazon Polly 提供的所有語音。

	語言和語言變體	語言代碼	名稱/ID	Gender	生成式語音	長形語音	神經語音	標準語音
1	阿拉伯文	arb	Zeina	女性	否	否	否	是

	語言和語言變體	語言代碼	名稱/ID	Gender	生成式語音	長形語音	神經語音	標準語音
2	阿拉伯文 (海灣)	ar-AE	Hala*	女性	否	否	是	否
			Zayd*	男性	否	否	是	否
3	荷蘭文 (比利時)	nl-BE	利薩	女性	是	否	是	否
4	加泰隆尼亞文	ca-ES	Arlet	女性	否	否	是	否
5	捷克文	cs-CZ	Jitka	女性	否	否	是	否
6	中文 (中文)	yue-CN	海津	女性	否	否	是	否
7	中文 (國語)	cmn-CN	Zhiyu	女性	否	否	是	是
8	丹麥文	da-DK	Naja	女性	否	否	否	是
			Mads	男性	否	否	否	是
			Sofie	女性	否	否	是	否
9	荷蘭文	nl-NL	月亮	女性	是	否	是	否
			Lotte	女性	否	否	否	是
			Ruben	男性	否	否	否	是

	語言和語言變體	語言代碼	名稱/ID	Gender	生成式語音	長形語音	神經語音	標準語音
10	英文 (澳洲)	en-AU	Nicole	女性	否	否	否	是
			奧利維亞	女性	是	否	是	否
				男性	否	否	否	是
			Russell					
11	英文 (英國)	en-GB	Amy**	女性	是	否	是	是
			Emma	女性	否	否	是	是
			Brian	男性	是	否	是	是
			Arthur	男性	否	否	是	否
12	英文 (印度)	en-IN	Aditi*	女性	否	否	否	是
			Raveena	女性	否	否	否	是
			Kajal*	女性	是	否	是	否
13	英文 (愛爾蘭)	en-IE	Niamh	女性	是	否	是	否
14	英文 (紐西蘭)	en-NZ	Aria	女性	是	否	是	否
15	英文 (新加坡)	en-SG	Jasmine	女性	是	否	是	否
16	英文 (南非)	en-ZA	Ayanda	女性	是	否	是	否

	語言和 語言變 體	語言代 碼	名稱/ID	Gender	生成式 語音	長形語 音	神經語 音	標準語 音
17	英文 (美 國)	zh-TW	丹尼爾 文	女性	是	是	是	否
				男性	否	是	是	否
			Gregory	女性 (兒 童)	否	否	是	是
			Ivy		是	否	是	是
			Joanna**	女性	否	否	是	是
			Kendra	女性	否	否	是	是
			Kimberly	女性	是	否	是	是
			Salli	女性	否	否	是	是
			Joey	男性	否	否	是	否
			Justin	男性 (兒 童)	否	否	是	是
			Kevin	男性 (兒 童)	是	否	是	否
			Matthew**		是	是	是	否
			Ruth	男性	是	否	是	否
			Stephen	女性	是	否	否	否
			Tiffany	男性	否	是	否	否
			派翠克	女性				
				男性				
18	英文 (威 爾)	en-GB- WLS	Geraint	男性	否	否	否	是
19	芬蘭文	fi-FI	Suvi	女性	否	否	是	否

	語言和 語言變 體	語言代 碼	名稱/ID	Gender	生成式 語音	長形語 音	神經語 音	標準語 音
20	法文	fr-FR	Ambre	女性	是	否	否	否
			Céline/ Celine	女性	是	否	否	是
				男性	是	否	否	否
			佛羅里 安	女性	是	否	是	是
			Léa	男性	否	否	否	是
			Mathieu	男性	是	否	是	否
			Rémi					
21	法文 (比利 時)	fr-BE	伊薩貝 爾	女性	是	否	是	否
22	法文 (加 拿大)	fr-CA	Chantal	女性	否	否	否	是
			加布里 爾	女性	是	否	是	否
				男性	是	否	是	否
			Liam					
23	德文	de-DE	Marlene	女性	否	否	否	是
			Vicki	女性	是	否	是	是
			Hans	男性	否	否	否	是
			丹尼爾	男性	是	否	是	否
			利納特	男性	是	否	否	否

	語言和語言變體	語言代碼	名稱/ID	Gender	生成式語音	長形語音	神經語音	標準語音
24	德文 (奧地利)	de-AT	漢納	女性	是	否	是	否
25	德文 (瑞士)	de-CH	薩布林納	女性	是	否	是	否
26	印地文	hi-IN	Aditi*	女性	否	否	否	是
			Kajal*	女性	否	否	是	否
27	冰島文	is-IS	Dóra/ Dora	女性	否	否	否	是
			Karl	男性	否	否	否	是
28	義大利文	it-IT	披頭四	女性	是	否	否	否
			Carla	女性	否	否	否	是
			Bianca	女性	是	否	是	是
			Lorenzo	男性	是	否	否	否
			Giorgio	男性	否	否	否	是
			Adriano	男性	否	否	是	否
29	日文	ja-JP	Mizuki	女性	否	否	否	是
			Takumi	男性	否	否	是	是
			卡茲哈	女性	否	否	是	否
			Tomoko	女性	否	否	是	否

	語言和語言變體	語言代碼	名稱/ID	Gender	生成式語音	長形語音	神經語音	標準語音
30	韓文	ko-KR	Seoyeon	女性	是	否	是	是
			Jihye	女性	否	否	是	否
31	挪威文	nb-NO	Liv	女性	否	否	否	是
			Ida	女性	否	否	是	否
32	波蘭文	pl-PL	Ewa	女性	是	否	否	是
			Maja	女性	否	否	否	是
			Jacek	男性	否	否	否	是
			Jan	男性	否	否	否	是
			奧拉	女性	是	否	是	否
33	葡萄牙文 (巴西)	pt-BR	Camila	女性	是	否	是	是
			Vitória/	女性	否	否	是	是
			Vitoria	男性	否	否	否	是
			Ricardo	男性	否	否	是	否
			Thiago					
34	葡萄牙文 (歐洲)	pt-PT	Inês/	女性	否	否	是	是
			Ines	男性	否	否	否	是
			Cristiano					
35	羅馬尼亞文	ro-RO	Carmen	女性	否	否	否	是
36	俄文	ru-RU	Tatyana	女性	否	否	否	是
			Maxim	男性	否	否	否	是

	語言和語言變體	語言代碼	名稱/ID	Gender	生成式語音	長形語音	神經語音	標準語音
37	西班牙文（西班牙）	es-ES	Conchita	女性	否	否	否	是
			Lucia	女性	是	否	是	是
			Alba	女性	否	是	否	否
			Enrique	男性	否	否	否	是
			塞爾吉奧	男性	是	否	是	否
			Raúl	男性	否	是	否	否
38	西班牙文（墨西哥）	es-MX	Mia	女性	是	否	是	是
			安德列	男性	是	否	是	否
39	西班牙文（美國）	es-US	Lupe**	女性	是	否	是	是
			Penélope/	女性	否	否	否	是
			Penelope	男性	否	否	否	是
			Miguel	男性	是	否	是	否
40	瑞典文	sv-SE	Astrid	女性	否	否	否	是
			Elin	女性	否	否	是	否
4.1	土耳其文	tr-TR	Filiz	女性	否	否	否	是
			爆量	女性	否	否	是	否
42	威爾斯文	cy-GB	Gwyneth	女性	否	否	否	是

* 此語音為雙語。如需詳細資訊，請參閱[雙語語音](#)。

** 與神經格式搭配使用時，這些語音可以與新聞播報員說話風格搭配使用。如需詳細資訊，請參閱[套用新聞播報員語音](#)。

每個 Amazon Polly 語音引擎都有獨特的功能。進一步了解 Amazon Polly 提供之語音引擎的功能和區域可用性：

- [生成式語音](#)
- [長形語音](#)
- [神經語音](#)
- [標準語音](#)

品牌聲音

除了上表中列出的可用語音之外，您還可以使用 Amazon Polly 為您的品牌角色建立自訂語音。透過品牌語音，您可以為客戶提供獨特且獨佔的語音。若要進一步了解 Amazon Polly 品牌語音，請參閱[品牌語音](#)。

雙語語音

Amazon Polly 有兩種產生雙語語音的方式：

- [有腔調的雙語語音](#)
- [全雙語語音](#)

有腔調的雙語語音

重音雙語語音可以使用任何 Amazon Polly 語音建立，但僅限於使用 SSML 標籤時。

一般而言，輸入文字中的所有字詞都會以您正在使用之語音的預設語言來朗讀。

例如，如果您使用 Joanna 的語音（會說美式英文），Amazon Polly 會以不含法文重音的 Joanna 語音說出以下內容：

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

```
</speak>
```

在這種情況下，Je ne parle pas français 這句話會用英語的方式說出。

不過，如果您使用 Joanna 語音搭配 `<lang>` 標籤，Amazon Polly 會以美式法文的 Joanna 語音說出句子：

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

由於 Joanna 不是以法文為母語，發音會以其母語語言 (美式英文) 為主。例如，完美的法語發音在發出文字 français 的 /R/ 時會帶有小舌顫音，但 Joanna 的美式英文語音則會發音為對應的聲音 /r/。

如果您使用說義大利文的 Giorgio 語音搭配下列文字，Amazon Polly 會使用義大利文發音，以 Giorgio 的聲音說出句子：

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

全雙語語音

Aditi 或 Kajal (印度英文和印地文) 等全雙語語音可以流暢地說兩種語言。這可讓您使用相同語音，在單一文字中使用來自兩種語言的文字和句子。

目前，Aditi、Kajal、Hala 和 Zayd 是唯一可用的全雙語語音。

使用雙語語音 (範例：Aditi)

Aditi 可同時流暢地說印度英文 (en-IN) 和印地語 (hi-IN)。您可以同時合成使用英文和印地語的語音，而語音可在這兩個語言間切換，即使是在相同句子中亦然。

您可以透過兩個不同的形式使用印地語：

- Devanagari : "उसेन कहाँ, खेल तोह अब शुरू होगा"
- Romanagari (使用拉丁字母) : "Usne kahan, khel toh ab shuru hoga"

此外，您可以在單一句子中混合使用英文和印地語其中之一或兩個形式：

- Devanagari + 英文 : "This is the song कभी कभी अदिति"
- Romanagari + 英文 : "This is the song from the movie Jaane Tu Ya Jaane Na."
- Devanagari + Romanagari + 英文 : "This is the song कभी कभी अदिति from the movie Jaane Tu Ya Jaane Na."

由於 Aditi 是雙語語音，因此所有這些案例中的文字都會正確讀取，因為 Amazon Polly 可以區分語言和指令碼。

Amazon Polly 也支援英文（阿拉伯數字）和印地文 (Devanagari 數字) 的數字、日期、時間和貨幣擴展。在預設情況下，阿拉伯數字會使用印度英文讀取。若要讓 Amazon Polly 在印地文中讀取它們，您必須使用 `hi-IN` 語言程式碼參數。

套用新聞播報員語音

人們會根據情境使用不同的說話風格。例如，輕鬆的交談聽起來與電視或廣播新聞非常不同。由於標準語音的製作方式，無法產生不同的說話風格。不過，神經語音可以。它們可以針對特定說話風格進行訓練，其變化和強調該風格固有的某些語音部分。

除了預設的神經語音之外，Amazon Polly 還提供新聞播報員說話風格，使用神經系統以電視或廣播新聞播報員風格產生語音。新聞播報員風格適用於美式英文 (en-US) 的 Matthew 和 Joanna 語音、美式西班牙文 (es-US) 的 Lupe 語音，以及英式英文 (en-GB) 的 Amy 語音。

若要使用新聞播報員風格，您必須先選擇神經引擎，然後在您的輸入文字中使用以下步驟所述的語法。

Note

- 若要使用任何神經說話風格，您必須使用支援神經語音的 AWS 其中一個區域。並非所有區域皆可提供此選項。如需詳細資訊，請參閱[功能和區域相容性](#)。

Console

套用新聞播報員風格

1. 開啟位於 <https://console.aws.amazon.com/polly/> 的 Amazon Polly 主控台。
2. 請確定您使用支援 AWS 神經語音的區域。
3. 在 Text-to-Speech (文字轉語音) 頁面上，針對 Engine (引擎) 選擇 Neural (神經)。

4. 選擇您要使用的語言和語音。只有 Matthew 和 Joanna for US English (en-US)、Lupe for US Spanish (es-US) 和 Amy for British English (en-GB) 可在新聞播報員語音中使用。
5. 開啟 SSML。
6. 使用新聞播報員風格 SSML 語法，將輸入文字新增到您的文字轉語音請求。

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

例如，您可以使用新聞播報員標籤，如下所示：

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

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

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

7. 選擇接聽。

AWS CLI

套用新聞播報員風格

1. 在您的 API 請求中，包含引擎參數，其 `neural` 值為：

```
--engine neural
```

2. 使用新聞播報員風格 SSML 語法，將輸入文字新增到您的 API 請求。

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

例如，您可以使用新聞播報員標籤，如下所示：

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

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

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

如需 SSML 的詳細資訊，請參閱[支援的 SSML 標籤](#)。

聆聽語音

[設定](#) Amazon Polly 後，您可以在 主控台上使用自訂文字測試語音。

在主控台上接聽 Amazon Polly 語音

1. 登入 AWS 管理主控台，並在 <https://console.aws.amazon.com/polly/>：// 開啟 Amazon Polly 主控台。
2. 選擇 Text-to-Speech (文字轉換語音) 索引標籤。
3. 針對引擎，選擇生成式、長形、神經或標準。
4. 選取語言和區域。然後選擇語音。
5. 輸入語音的文字以說話或使用預設片語，然後選擇接聽。

計時語音速度

由於語音之間的自然變化，每個可用的語音會以稍微不同的速度說話。例如，使用美式英文語音時，Ivy 和 Joanna 比 Matthew 略快，而且比 Joey 快得多。由於語音之間差異很大，因此 Amazon Polly 語音沒有標準速度（每分鐘字數）。不過，您可以找到語音使用[語音標記](#)說出所選文字所需的時間。

計時口語文字段落的長度

1. 開啟 AWS CLI。
2. 執行下列程式碼，視需要填入。

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

3. 打開 LengthOfText.txt.

如果文字是「Mary has a little lamb」，Amazon Polly 傳回的最後幾行會是：

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

基本上，最後視素即「lamb」中最後字母發音，是在開始說話後 1082 毫秒處開始。雖然這並非音訊確實的長度，但是已經很接近，可做為語音比較的基礎。

變更語音速度

對於特定的應用程式，您可能會發現您希望語音放慢或加快。如果語音的速度有問題，Amazon Polly 會提供使用 SSML 標籤修改的功能。例如，如果您的組織製作的應用程式會將書籍讀給外來的觀眾，您可能會想要改變語音速度。您的受眾可能會說英文，但流暢程度有限。Amazon Polly 使用 SSML `<prosody>` 標籤協助您降低語音速率。

您可以使用百分比：

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

或預設速度：

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

搭配 Amazon Polly 使用 SSML 時，您可以使用兩種速度選項：

- 預設速度：x-slow、slow、fast、medium 和 x-fast。在這些情況下，每個選項是大約的速度，其取決於您慣用的語音。medium 選項是正常的語音速度。
- n% 的語音率：可使用介於 20% 到 200% 之間的任何百分比語音率。在這些情況下，您可以選擇完全符合所需的速度。不過，視您選擇的語音而定，語音的實際速度為近似值。100% 視為語音的正常速度。

Note

以各種速度測試您選取的語音。每個選項的速度都是近似值，取決於您選擇的語音。

如需使用 prosody 標籤的詳細資訊，請參閱 [控制音量、說話速率和音調](#)。

Amazon Polly 中的語言

Amazon Polly 支援下列語言，可用於合成語音。每種語言都有唯一的語言代碼。這些語言代碼是 [W3C 語言識別標籤](#)（語言名稱為 *ISO 639-3*，國家代碼為 *ISO 3166*）。

從下表選取語言，以取得 Amazon Polly 提供之音素和視素的詳細資訊。

語言	語言代碼
阿拉伯文	arb
阿拉伯文 (海灣)	ar-AE
加泰隆尼亞文	ca-ES
中文 (中文)	yue-CN
中文 (國語)	cmn-CN
捷克文	cs-CZ
丹麥文	da-DK
荷蘭文 (比)	nl-BE

語言	語言代碼	
利時)		
荷蘭文	nl-NL	
英文 (澳洲)	en-AU	
英文 (英國)	en-GB	
英文 (印度)	en-IN	
英文 (紐西蘭)	en-NZ	
英文 (新加坡)	en-SG	
英文 (南非)	en-ZA	
英文 (美國)	zh-TW	

語言	語言代碼	
英文 (威爾)	en-GB-WLS	
芬蘭文	FI	
法文	fr-FR	
法文 (比利時)	fr-BE	
法文 (加拿大)	fr-CA	
印地文	hi-IN	
德文	de-DE	
德文 (奧地利)	de-AT	
德文 (瑞士標準)	de-CH	
冰島文	is-IS	

語言	語言代碼	
義大利文	it-IT	
日文	ja-JP	
韓文	ko-KR	
挪威文	nb-NO	
波蘭文	pl-PL	
葡萄牙文 (巴西)	pt-BR	
葡萄牙文 (歐洲)	pt-PT	
羅馬尼亞文	ro-RO	
俄文	ru-RU	
西班牙文 (西班牙)	es-ES	

語言	語言代碼	
西班牙文 (墨西哥)	es-MX	
西班牙文 (美國)	es-US	
瑞典文	sv-SE	
土耳其文	tr-TR	
威爾斯文	cy-GB	

阿拉伯文 (arb)

下表列出國際音標 (IPA) 音素、拓展音標字母評估法 (X-SAMPA) 符號，以及 Amazon Polly 支援的 Zeina 阿拉伯語聲音對應視素。

音素/視素表

IPA	X-SAMPA	描述	範例	視素
子音				
ʔ	ʔ	聲門塞音	أنا	
ħ	ħ\	咽部擦音	عمر	k

IPA	X-SAMPA	描述	範例	視素
b	b	濁雙唇塞音	بَلَد	p
d	d	濁齒齦塞音	دَارِي	t
d ^ʁ	d_?\\	加強語氣的濁齒齦塞音	ضَوء	t
ḏ	dZ	濁顎齦塞擦音	جَمِيل	S
ð	D	濁齒擦音	ذَلِكَ	T
ð ^ʁ	D_?\\	加強語氣的濁齒擦音	ظَالِم	T
f	f	清唇齒擦音	فَصْل	f
g	g	濁軟顎塞音	إِنْجَلْتِ رَا	k
ɣ	G	濁軟顎擦音	غَرَب	k
h	h	清聲門擦音	هَذَا	k
j	j	硬顎近音	يَمَشِي	i
k	k	清軟顎塞音	كَالْب	k
l	l	齒齦邊近音	لَاقِي	t
l ^ʁ	l_G	加強語氣的齒齦邊近音	哨哨哨哨哨哨哨哨哨哨	t
m	m	雙唇鼻音	مَإِذَا	p
n	n	齒齦鼻音	نُور	t
p	p	清雙唇塞音	حَبَسَ	p
q	q	清小舌塞音	قَرِيبَ	k
r	r	齒齦顫音	رَمَل	r

IPA	X-SAMPA	描述	範例	視素
s	s	清齒齶擦音	سُؤَال	s
s ^ʕ	s_?\\	加強語氣的清齒齶擦音	صَاحِب	s
ʃ	S	清齶後擦音	شُكْر	S
t	t	清齒齶塞音	تَمَر	t
t ^ʕ	t_?\\	加強語氣的清齒齶塞音	طَالِب	t
θ	T	清齒擦音	ثَلَاث	T
v	v	濁唇齒擦音	فِي تَامِيْن	f
w	w	唇軟顎近音	وَلَد	u
x	x	清軟顎擦音	خَوْف	k
ħ	X\\	清咽部擦音	حَوْلَ	k
z	z	濁齒齶擦音	زُهْر	s
母音				
a	a	開前不圓唇母音	بَرَد	a
a:	a:	長開前不圓唇母音	دَار	a
a ^ʕ	A_?\\	加強語氣的開後不圓唇母音	طَابَل	a
a ^ʕ :	A_?\\:	加強語氣的長開後不圓唇母音	طَالِم	a
u	u	閉後圓唇母音	شُرْب	u
u:	u:	長閉後圓唇母音	سُور	u

IPA	X-SAMPA	描述	範例	視素
u ^ɾ	u_?\ 	加強語氣的閉後圓唇母音	بَدَّ	u
u ^ɾ :	u_?\ :	加強語氣的長閉後圓唇母音	طول	u
i	i	閉前不圓唇母音	بِنَت	i
i:	i:	長閉前不圓唇母音	حَزِين	i
i ^ɾ	i_?\ 	加強語氣的閉前不圓唇母音	ضَدَّ	i
i ^ɾ :	i_?\ :	加強語氣的長閉前不圓唇母音	ماضي	i
e	e	半閉前不圓唇母音	ماركَة	e
e:	e:	長半閉前不圓唇母音	موديل	e
ɔ	O	半開後圓唇母音	تكنولوجيا	O
ɔ:	O:	長半開後圓唇母音	تلفزيون	O

阿拉伯文（海灣）(ar-AE)

下表列出國際音標 (IPA) 音素、拓展音標字母評估法 (X-SAMPA) 符號，以及 Amazon Polly 支援的阿拉伯語 Hala 聲音對應視素。

音素/視素表

IPA	X-SAMPA	描述	範例	發音	視素
子音					
b	b	濁雙唇塞音	「ر」	/ˈba.lad/	b

IPA	X-SAMPA	描述	範例	發音	視素
d	d	濁齒齦塞音	沙烏地阿拉伯文	/ " r a d /	d
dʕ	d_ʔ\	pharyngealized voiced alveolar plosive	「」	/ " d_ʔ\ a w ʔ /	D
f	f	清唇齒擦音	1911 年 12 月 11 日	/ " f l . r l n /	f
g	g	濁軟顎塞音	-----	/ " g a : l /	k
j	j	語音方言近似	-----	/ " j l m . S i : /	i
k	k	清軟顎塞音	「」	/ " k a : . m i l /	k
l	l	語音橢圓側向近似值	「」	/ " l e : l /	t
lʕ	l_G	pharyngealized voiced alveolar lateral approximant	寮寮寮寮寮寮	/ ʔ \ a b . 「 d A_ʔ \ l_G . l_G A_ʔ \ /	t
m	m	雙唇鼻塞	相比之下，	/ " m l j . j a /	p
n	n	齒槽鼻塞	نور	/ " n u : r /	t
p	p	清雙唇塞音	「」	/ " ʔ O . p e . r a : /	p
q	q	清小舌塞音	-----	/ " q A_ʔ \ s_ʔ \ r /	k
r	r	齒齦顫音	「」	/ " r a . m l l /	r

IPA	X-SAMPA	描述	範例	發音	視素
s	s	清齒齦擦音	別名：	/ " s l m . s l m /	s
s ^ɸ	s_? \	pharyngealized 無語音立體聲 複寫	-----	/ " s_? \ A_? : 。 X l l b /	s
t	t	清齒齦塞音	傳兀兀兀	/ " t a . m a r /	t
t ^ɸ	t_? \	pharyngealized 無語音立體聲 複寫	停滯不前	/ " t_? \ A_? : . l l b /	t
v	v	濁唇齒擦音	في تامين	/ v i : . t A 。 「 m i : n /	f
w	w	語音迷宮近似 值	「」	/ " w a : . j l d /	u
x	x	清軟顎擦音	「」	/ x a 。 「 r u : f /	k
z	z	清軟顎擦音	「」	/ " z h u : r /	s
ð	D	語音齒間仿造	「」	/ " D a : . l l k /	D
ð ^ɸ	D_? \	語音化語音齒 間虛構	「」	/ D_? \ A_? \ 。 「 l a : m /	D
ħ	X \	清咽部擦音	————	/ ? l 。 「 X \ i : n /	k
ŋ	N	小聲鼻塞	第 2 版 第 2 版 第 2 版 第 2 版	/ h O N 。 「 k O N g /	k
ɣ	G	濁軟顎擦音	「」	/ G l 。 「 r i : . b a /	k

IPA	X-SAMPA	描述	範例	發音	視素
ʃ	S	清齦後擦音	橋樑	/ " S a m s /	S
ʒ	Z	濁齦後擦音	井字號	/ Z a 。 「 k e : t /	S
ʔ	ʔ	聲門塞音	「」	/ m u 。 「 ʔ a s . s a . s a /	
ʕ	ʔ\	咽部擦音	停滯不前	/ " ʔ \ a : m m /	k
dʒ	dZ	濁顎齦塞擦音	錯誤地說， 錯誤地說， 錯誤地說，錯 誤地說，錯誤 地說，錯誤 地說，錯誤地 說，	/ " dZ a : m . ʔ \ a /	S
θ	T	語音齒間仿造	「」	/ T a 。 「 l a : 。 T a /	T
ħ	h	清喉擦音	「」	/ " h l a : l /	k
母音					
æ	a	中開前無圓短母音	https : //	/ " s a . f a r /	a
ɑ̣	A_ʔ\	pharyngealized 露背不圓短母音	停滯不前	/ " s _ ʔ \ A _ ʔ \ l b /	a
æ:	a:	中開前不圓長母音	樟腦樟腦樟腦 樟腦樟腦樟	/ " b a : b /	a

IPA	X-SAMPA	描述	範例	發音	視素
ɑː	A_?ː	pharyngealized 露背不圓長母音	井字號	/ " n A_? :。 D_? \ i_? \ dZ /	a
a	A	開啟中央無圓 短母音	Wi-Fi	/ " w A j . f A j /	a
i	i	tense close front unrounded 短 母音 (MSA)	嘯嘯嘯嘯嘯嘯	/ ? i s。 「 X \ A_? \ : q /	i
ɪ	ɪ	鬆弛前閉無環 短母音	「」	/ " b l n t /	i
ɪ̤	i_? \	pharyngealized 閉前不圓短母音	停滯狀態停滯 狀態	/ " t_? \ i_? \ f l l /	i
iː	iː	關閉正面無環 長母音	https : //O ;	/ s a。 「 b i : l /	i
ɪː	i_? :	pharyngealized close front unrounded long vowel	「」	/ r A_? \。 「 t_? \ i_? : b /	i
u	u	拉緊閉後圓短 母音 (MSA)	「」	/ " m u x . t a . r i ? \ /	u
ʊ	U	鬆弛露背圓唇 短母音	「」	/ r U。 「 s u : m /	u
ṳ	u_? \	pharyngealized 閉後圓短母音	Mi- \ \ \ \ \ \ \ \	/ ? \ u_? \ s_? \。 「 f u : r /	u

IPA	X-SAMPA	描述	範例	發音	視素
u:	u:	閉後圓唇長母音	婆婆婆婆婆婆	/ " t u : t /	u
u ^ɸ :	u_? \:	pharyngealized 閉後圓唇長母音	停頓停頓停頓 停頓	/ " s_ ? \ u_ ? \ : r /	u
e	e	中前方無圓短母音	嘯嘯嘯嘯嘯嘯 嘯嘯嘯嘯嘯	/ " s e n t /	e
e:	e:	中前方無圓長母音	沙烏地那	/ " ? e : S /	e
ɔ	O	開中後圓短母音	「」	/ d O 。 「 l A r /	O
ɔ:	O:	中開後圓長母音	「」	/ " l O : n /	O

加泰隆尼亞文 (ca-ES)

下表列出國際音標 (IPA) 音素、拓展音標字母評估法 (X-SAMPA) 符號，以及 Amazon Polly 支援的加泰隆尼亞語 Arlet 聲音對應的視素。

音素/視素表

IPA	X-SAMPA	描述	範例	視素
子音				
p	p	清雙唇塞音	p loure	p
t	t	清齒齶塞音	T arscopea	t
k	k	清軟顎塞音	c om	k
b	b	濁雙唇塞音	b ata	p

IPA	X-SAMPA	描述	範例	視素
d	d	濁齒齶塞音	end oll	t
g	g	濁軟顎塞音	g ros	k
m	m	語音雙唇鼻音	m Anera	p
n	n	語音聲道鼻聲	don ar	t
ɲ	J	語音顎鼻音	any	J
ŋ	N	語音小唇鼻音	pin güí	k
ɫ	5	語音化腔體側向近似值 (暗 l)	al bercoc	l
ʎ	L	語音顎側近似值	ll op	J
r	r	語音 alveolar trill	parr a	r
ɾ	4	語音聲道點擊	分頁 a	t
f	f	清唇齒擦音	émf asi	f
s	s	清齒齶擦音	s ac	s
z	z	濁齒齶擦音	calz es	s
ʃ	S	清齶後擦音	guix	S
ʒ	Z	濁齶後擦音	col·leg i	S
ɬ	tS	清齶後塞擦音	cotx e	S
ɮ	dZ	濁顎齶塞擦音	platj a	S
β	B	語音雙工近似值	ob ert	B
ð	D	語音牙科近似值	bed oll	T
j	j	語音方言近似	noi a	i

IPA	X-SAMPA	描述	範例	視素
ɣ	G	語音小說近似值	pega	k
v	v	濁唇齒擦音	af gà	f
w	w	語音迷宮近似值	a	u
x	x	清軟顎擦音	J iménez	k
j	j\	濁硬顎擦音	y eso	J
l	l	語音橢圓側向近似值	al ondra	t
θ	T	清齒擦音	Gonz ález	T
母音				
a	a	開後母音	casa	a
e	e	半閉前不圓唇母音	lle nya	e
ɛ	E	半開前不圓唇母音	xe c	E
i	i	前閉無環母音	vi sca	i
o	o	半閉後圓唇母音	go s	o
ɔ	O	半開後圓唇母音	jo c	O
u	u	閉後圓唇母音	u n	u
ə	@	中央母音	casa	@
其他符號				
'	"	主要重音	Alabama	
,	%	次要重音	Alabama	
.	.	音節邊界	A.la.ba.ma	

中文（中文）(yue-CN)

下表列出 Amazon Polly 所支援之廣語語音的 Jyutping 和國際音標 (IPA) 音素。Jyutping 是廣式的羅馬化系統，常用於學術界和廣式說話者之間。IPA 和 X-SAMPA 並不常用，但可支援英文。表格中的 IPA 和 X-SAMPA 符號僅供參考，不應用於中文轉錄。也會顯示變形範例和對應的視覺效果。

若要讓 Amazon Polly 搭配 Jyutping 使用音標發音，請使用 `phoneme alphabet="x-amazon-jyutping"` 標籤。

以下範例顯示此情況與每個標準。

Jyutping：

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

IPA：

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

X-SAMPA：

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

Note

Amazon Polly 僅接受以 UTF-8 編碼的廣式輸入。

音素/視素表

標點符號	IPA	X-SAMPA	描述	變形範例	視素
子音					
b	p	p	清雙唇塞音	巴, b aa1	p
c	TSH	ts_h	送氣清齒齶塞擦音	叉, c aa1	s
d	t	t	清齒齶塞音	打, d aa2	t
f	f	f	清唇齒擦音	花, f aa1	f
g	k	k	清軟顎塞音	家, g aa1	k
gw	kw	k_w	迷宮化無聲小說複數	瓜, gw aa1	u
h	h	h	清聲門擦音	哈, h aa1	k
k	k ^h	k_h	送氣清軟顎塞音	卡, k aa1	k
kw	kwh	k_wh	迷宮化吸音無聲小白齒倍數	YouTube, kw aa1	u
l	l	l	齒齶邊近音	「」, l aa1	t
m	m	m	雙唇鼻音	「」, m aa1	p
m	m	m=	音節雙唇鼻音	「」, 「m4」	p
ng	ŋ	N	軟顎鼻音	牙, ng aa4	k
ng	ŋ	N=	syllabic 小唇鼻	「」, 「ng4」	k
n	n	n	齒齶鼻音	拿, n aa4	t
p	p ^h	p_h	送氣清雙唇塞音	趴, p aa1	p
s	s	s	清齒齶擦音	沙, s aa1	s

標點符號	IPA	X-SAMPA	描述	變形範例	視素
t	t ^h	t_h	送氣清齒齶塞音	他、t aa1	t
w	w	w	唇軟顎近音	無條件，有 aa1	u
y	j	j	硬顎近音	z、j aa5	i
z	ts	ts	清齒齶塞擦音	da、z aa1	s
母音					
a	ɐ	6	次開央母音	吉、ga t1	a
aa	ɑ	A	開後不圓唇母音	家，gaa1	a
aaɪ	◌◌i	Ai	diphthong	街，gaai1	a
aaʊ	◌◌u	Au	diphthong	交、gaau1	a
ai	「哈明」	6i	diphthong	雞、gai1	a
au	◌魯	6u	diphthong	溝、kau1	a
e	ɛ	E	半開前不圓唇母音	爹、de1	E
ei	ei	ei	diphthong	基、gei1	e
eo	ɵ	8	半閉央圓唇母音	春、ceo n1	o
eoɪ	貝尼	8y	雙母音	居留、geoi1	o
eu	沙烏	Eu	雙母音	掉入 掉垃圾、deu6	E
i	i	i	閉前不圓唇母音	斯，si1	i
ɪ	ɪ	ɪ	次閉次前不圓唇母音	激、gik1	i

標點符號	IPA	X-SAMPA	描述	變形範例	視素
iu	iu	iu	雙母音	「」、 「giu1」	i
o	ɔ	O	半開後圓唇母音	哥、go1	O
oe	œ	9	半開前圓唇母音	鋸、goe3	O
oi	ɔi	Oi	diphthong	該 goi1	O
ou	ou	ou	diphthong	高、gou1	o
u	u	u	閉後圓唇母音	姑、gu1	u
u	ʊ	U	次閉次後圓唇母音	谷、gu k5	u
ui	ui	ui	diphthong	刼、gui6	u
yu	y	y	閉前圓唇母音	於、jyu1	u
音調標記和其他符號					
1			高階	詩文、si1	
2			中上升	史、si2	
3			中層級	試、si3	
4			非常低的層級	時、si4	
5			低上升	市、si5	
6			低階	是、si6	
-	.	.	音節邊界	語音 jyu5-jam1	

中文（國語）(cmn-CN)

下表列出 Amazon Polly 所支援中文語音的拼音和國際音標 (IPA) 音素。拼音是標準中文羅馬化的國際標準。IPA 和 X-SAMPA 並不常用，但可支援英文。表格中的 IPA 和 X-SAMPA 符號僅供參考，不應用於中文轉錄。也會顯示拼音範例和對應視素。

若要讓 Amazon Polly 搭配 Pinyin 使用音標發音，請使用 `phoneme alphabet="x-amazon-phonetic standard used"` 標籤。

以下範例顯示此情況與每個標準。

拼音：

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

IPA：

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

X-SAMPA：

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

Note

Amazon Polly 僅接受以 UTF-8 編碼的中文輸入。Amazon Polly 目前不支援 GB 18030 編碼標準。

音素/視素表

拼音	IPA	X-SAMPA	描述	拼音範例	視素
子音					
f	f	f	清唇齒擦音	發，fa1	f
h	h	h	清聲門擦音	和，he2	k
g	k	k	清軟顎塞音	古，gu3	k
k	k ^h	k_h	送氣清軟顎塞音	苦，ku3	k
l	l	l	齒齦邊近音	拉，la1	t
m	m	m	雙唇鼻音	罵，ma4	p
n	n	n	齒齦鼻音	那，na4	t
ng	ŋ	N	軟顎鼻音	正，zheng4	k
b	p	p	清雙唇塞音	爸，ba4	p
p	p ^h	p_h	送氣清雙唇塞音	怕，pa4	p
s	s	s	清齒齦擦音	四，si4	s
x	ç	s\	清硬顎擦音	西，xi1	J
sh	ʃ	s`	清捲舌擦音	是，shi4	S
d	t	t	清齒齦塞音	打，da3	t
t	t ^h	t_h	送氣清齒齦塞音	他，ta1	t
zh	ʈʂ	t`s`	清捲舌塞擦音	之，zhi1	S
ch	ʈʂ ^h	t`s`_h	清送氣捲舌塞擦音	吃，chi1	S
s	ʈʂ	ts	清齒齦塞擦音	字，zi4	s

拼音	IPA	X-SAMPA	描述	拼音範例	視素
j		ts\	清齶顎塞擦音	鸡, ji1	J
q		ts_h	送氣清齶顎塞擦音	七, qi1	J
c		ts_h	送氣清齒齶塞擦音	次, ci4	s
w	w	w	唇軟顎近音	我, wo3	u
r		z`	濁捲舌擦音	日, ri4	S
"er" 和 "r" 色彩音節					
er		@`	r 色彩中央母音	二, er4	@
-r			r 色彩音節	馅儿, xianr4	@
母音					
e		7	半閉後不圓唇母音	恶, e4	e
e		@	中央母音	恩, en1	@
a	a	a	開前不圓唇母音	安, an1	a
ai	aɪ	aɪ	雙母音	爱, ai4	a
ao	aʊ	aU	雙母音	奥, ao4	a
ei	eɪ	e	雙母音	诶, ei4	e
e		E	半開前不圓唇母音	姐, jie3	E
i	i	i	閉前不圓唇母音	鸡, ji1	i
ou	oʊ	oU	雙母音	欧, ou1	o
o		O	半開後圓唇母音	哦, o4	o
u	u	u	閉後圓唇母音	主, zhu3	u

拼音	IPA	X-SAMPA	描述	拼音範例	視素
yu	y	y	閉前圓唇母音	于, yu2	u

音調標記和其他符號

1			高階音調	淤, yu1	
2			升高音調	鱼, yu2	
3			低 (降低-升高) 音調	语, yu3	
4			降低音調	育, yu4	
0			中性音調	的, de0	
-	.	.	音節邊界	语音 yu3-yin1	

捷克文 (cs-CZ)

下表列出國際音標 (IPA) 音素、拓展音標字母評估法 (X-SAMPA) 符號，以及 Amazon Polly 所支援捷克文聲音的對應視素。

音素/視素表

IPA	X-SAMPA	描述	範例	視素
子音				
p	p	清雙唇塞音	p es	p
t	t	清齒齶塞音	t 確定	t
c	c	清硬顎塞音	t'uk	J
k	k	清軟顎塞音	k os	k
b	b	濁雙唇塞音	b ez	p

IPA	X-SAMPA	描述	範例	視素
d	d	濁齒齦塞音	d 正常	t
ɟ	J\	濁硬顎塞音	d'as	J
g	g	濁軟顎塞音	g um	k
f	f	清唇齒擦音	f ilm	f
v	v	濁唇齒擦音	v es	f
s	s	清齒齦擦音	en	s
z	z	濁齒齦擦音	z el	s
ʃ	S	清齶後擦音	šel	S
ʒ	Z	濁齶後擦音	žen	S
x	x	清軟顎擦音	chat	k
ɦ	h	清喉擦音	h us	k
ts	ts	清齒齦塞擦音	c o	s
tʃ	tS	清齶後塞擦音	čin	S
dz	dz	濁齒齦塞擦音	špic berský	s
dʒ	dZ	清齒齦塞擦音	džin	S
m	m	雙唇鼻音	m 或	p
n	n	齒齦鼻音	n os	t
ɲ	J	硬顎鼻音	ňader	J
ŋ	N	軟顎鼻音	ban ka	k
r	r	語音 alveolar trill	r ys	r

IPA	X-SAMPA	描述	範例	視素
ɹ	r_r	語音引發的齒槽虛構三角	řez	r
ɹcsv	r_0_r	無聲引發的立體聲浮水印	keř	r
l	l	齒齦邊近音	l es	t
j	j	硬顎近音	j en	i
w	w	labiovelar 近似值	W atson	u
ɾ	r_ =	syllabic voiced alveolar trill	kr k	r
ɹ̥	l_ =	音節齒齦邊近音	vl na	t
母音				
a	a	開前不圓唇母音	la n	a
a:	a:	長開前不圓唇母音	lá n	a
ɛ	E	半開前不圓唇母音	le t	E
ɛ:	E:	長半開前不圓唇母音	lé t	E
ɪ	l	次閉次前不圓唇母音	bi t	i
i:	i:	長閉前不圓唇母音	bít	i
o	o	半閉後圓唇母音	ho l	o
o:	o:	長半閉後圓唇母音	gól	o
u	u	閉後圓唇母音	pu l	u

IPA	X-SAMPA	描述	範例	視素
u:	u:	長閉後圓唇母音	pʊl	u
a↯u	au	雙母音	au 至	a
夏爾汪 汪	Eu	雙母音	eu ro	E
o↯u	ou	雙母音	mou k	o
其他符號				
'	“	主要重音		
.	.	音節邊界		

丹麥文 (da-DK)

下表列出國際音標 (IPA) 音素、拓展音標字母評估法 (X-SAMPA) 符號，以及 Amazon Polly 所支援丹麥文聲音的對應視素。

音素/視素表

IPA	X-SAMPA	描述	範例	視素
子音				
b	b	濁雙唇塞音	bat	p
d	d	濁齒齦塞音	da	t
ð	D	濁齒擦音	mad、thriller	T
f	f	清唇齒擦音	fat	f
g	g	濁軟顎塞音	gat	k
h	h	清聲門擦音	hat	k

IPA	X-SAMPA	描述	範例	視素
j	j	硬顎近音	jo	i
k	k	清軟顎塞音	kat	k
l	l	齒齦邊近音	ladt	t
m	m	雙唇鼻音	mat	p
n	n	齒齦鼻音	nay	t
ŋ	N	軟顎鼻音	lang	k
p	p	清雙唇塞音	pande	p
r	r	齒齦顫音	thriller、story	r
ʁ	R	濁小舌擦音	rat	k
s	s	清齒齦擦音	sat	s
t	t	清齒齦塞音	tal	t
v	v	濁唇齒擦音	vat	f
w	w	圓唇軟顎近音	hav、weekend	u
母音				
ø	2	半閉前圓唇母音	øst	o
ø:	2:	長半閉前圓唇母音	øse	o
e	6	次開央母音	mor	a
œ	9	半開前圓唇母音	skøn、grønt	O
œ:	9:	長半開前圓唇母音	høne、gøre	O
ə	@	中央母音	ane	@

IPA	X-SAMPA	描述	範例	視素
æ:	{:	長次開前不圓唇母音	male	a
a	a	開前不圓唇母音	man	a
æ	{	次開前不圓唇母音	adresse	a
ɑ	A	開後不圓唇母音	lak、tak	a
ɑ:	A:	長開後不圓唇母音	rase	a
e	e	半閉前不圓唇母音	midt	e
e:	e:	長半閉前不圓唇母音	mele	e
ɛ	E	半開前不圓唇母音	mæt	E
ɛ:	E:	長半開前不圓唇母音	mæle	E
i	i	閉前不圓唇母音	mit	i
i:	i:	長閉前不圓唇母音	mile	i
o	o	半閉後圓唇母音	foto	o
o:	o:	長半閉後圓唇母音	mole	o
ɔ	O	半開後圓唇母音	mund	O
ɔ:	O:	長半開後圓唇母音	måle	O
ɒ:	Q:	長開後圓唇母音	morse	O
u	u	閉後圓唇母音	lusk	u
u:	u:	長閉後圓唇母音	mule	u
ʌ	V	半開後不圓唇	kører	E

IPA	X-SAMPA	描述	範例	視素
y	y	閉前圓唇母音	yt	u
y:	y:	長閉前圓唇母音	hyle	u
其他符號				
'	"	主要重音	Alabama	
,	%	次要重音	Alabama	
.	.	音節邊界	A.la.ba.ma	

荷蘭文（比利時）(nl-BE)

下表列出國際音標 (IPA) 音素、拓展音標字母評估法 (X-SAMPA) 符號，以及 Amazon Polly 所支援比利時荷蘭文（法林）聲音對應的視素。

音素/視素表

IPA	X-SAMPA	描述	範例	視素
子音				
b	b	濁雙唇塞音	bak	p
d	d	濁齒齦塞音	dak	t
ɖʒ	dʒ	濁顎齦塞擦音	manager	S
f	f	清唇齒擦音	fel	f
g	g	濁軟顎塞音	goal	k
ɣ	G	濁軟顎擦音	hoed	k
ɦ	h\	清喉擦音	hand	k
j	j	硬顎近音	ja	i

IPA	X-SAMPA	描述	範例	視素
k	k	清軟顎塞音	kap	k
l	l	齒齦邊近音	land	t
m	m	雙唇鼻音	met	p
n	n	齒齦鼻音	net	t
ŋ	N	軟顎鼻音	ba ng	k
p	p	清雙唇塞音	pak	p
r	r	齒齦顫音	rand	r
s	s	清齒齦擦音	sein	s
ʃ	S	清齶後擦音	show	S
t	t	清齒齦塞音	tak	t
v	v	濁唇齒擦音	vel	f
ʋ	v\	唇齒近音	wit	f
x	x	清軟顎擦音	到 ch	k
z	z	濁齒齦擦音	ziin	s
ʒ	Z	濁齶後擦音	bagage	S
母音				
ø:	2:	長半閉前圓唇母音	neus	o
œy	9y	diphthong	buit	O
ə	@	中央母音	de	@
a:	a:	長開前不圓唇母音	baad	a

IPA	X-SAMPA	描述	範例	視素
ɑ:	A	開後不圓唇母音	bad	a
e:	e:	長半閉前不圓唇母音	beet	e
ɜ:	3:	長半開央不圓唇音	barrière	E
ɛ	E	半開前不圓唇母音	bed	E
ɛi	Ei	diphthong	beet	E
i	i	閉前不圓唇母音	vier	i
ɪ	ɪ	次閉次前不圓唇母音	pit	i
o:	o:	長半閉後圓唇母音	boot	o
ɔ	O	半開後圓唇母音	pot	O
u	u	閉後圓唇母音	hoed	u
ʌu	Vu	diphthong	fout	E
y:	y:	長閉前圓唇母音	fuut	u
ʏ	Y	次閉次前圓唇母音	hut	u

其他符號

'	"	主要重音	Alabama	
,	%	次要重音	Alabama	
.	.	音節邊界	A.la.ba.ma	

荷蘭文 (nl-NL)

下表列出國際音標 (IPA) 音素、拓展音標字母評估法 (X-SAMPA) 符號，以及 Amazon Polly 所支援荷蘭文聲音對應的視素。

音素/視素表

IPA	X-SAMPA	描述	範例	視素
子音				
b	b	濁雙唇塞音	bak	p
d	d	濁齒齦塞音	dak	t
ɖʒ	dʒ	濁顎齦塞擦音	manager	S
f	f	清唇齒擦音	fel	f
g	g	濁軟顎塞音	goal	k
ɣ	G	濁軟顎擦音	hoed	k
ɦ	h\	清喉擦音	hand	k
j	j	硬顎近音	ja	i
k	k	清軟顎塞音	kap	k
l	l	齒齦邊近音	land	t
m	m	雙唇鼻音	met	p
n	n	齒齦鼻音	net	t
ŋ	N	軟顎鼻音	ba ng	k
p	p	清雙唇塞音	pak	p
r	r	齒齦顫音	rand	r
s	s	清齒齦擦音	sein	s

IPA	X-SAMPA	描述	範例	視素
ʃ	S	清齶後擦音	show	S
t	t	清齒齶塞音	tak	t
v	v	濁唇齒擦音	vel	f
u	v\	唇齒近音	wit	f
x	x	清軟顎擦音	到 ch	k
z	z	濁齒齶擦音	ziin	s
ʒ	Z	濁齶後擦音	bagage	S
母音				
ø:	2:	長半閉前圓唇母音	neus	o
œy	9y	diphthong	buit	O
ə	@	中央母音	de	@
a:	a:	長開前不圓唇母音	baad	a
ɑ:	A	開後不圓唇母音	bad	a
e:	e:	長半閉前不圓唇母音	beet	e
ɜ:	3:	長半開央不圓唇音	barrière	E
ɛ	E	半開前不圓唇母音	bed	E
ɛi	Ei	diphthong	beet	E
i	i	閉前不圓唇母音	vier	i
ɪ	l	次閉次前不圓唇母音	pit	i

IPA	X-SAMPA	描述	範例	視素
o:	o:	長半閉後圓唇母音	boot	o
ɔ	O	半開後圓唇母音	pot	O
u	u	閉後圓唇母音	hoed	u
ʌu	Vu	diphthong	fout	E
y:	y:	長閉前圓唇母音	fuut	u
ʏ	Y	次閉次前圓唇母音	hut	u
其他符號				
'	"	主要重音	Alabama	
,	%	次要重音	Alabama	
.	.	音節邊界	A.la.ba.ma	

英文 (美國) (en-US)

下表列出國際音標 (IPA) 音素、拓展音標字母評估法 (X-SAMPA) 符號，以及 Amazon Polly 所支援美式英文聲音對應的視素。

音素/視素表

IPA	X-SAMPA	描述	範例	視素
子音				
b	b	濁雙唇塞音	bed	p
d	d	濁齒齦塞音	dig	t
ɹ̥dʒ	dʒ	濁顎齦塞擦音	jump	S
ð	D	濁齒擦音	then	T

IPA	X-SAMPA	描述	範例	視素
f	f	清唇齒擦音	five	f
g	g	濁軟顎塞音	game	k
h	h	清聲門擦音	house	k
j	j	硬顎近音	yes	i
k	k	清軟顎塞音	cat	k
l	l	齒齦邊近音	lay	l
m	m	雙唇鼻音	mouse	p
n	n	齒齦鼻音	nap	t
ŋ	N	軟顎鼻音	thing	k
p	p	清雙唇塞音	speak	p
ɹ	r\	齒齦近音	red	r
s	s	清齒齦擦音	seem	s
ʃ	S	清齶後擦音	ship	S
t	t	清齒齦塞音	trap	t
tʃ	tS	清齶後塞擦音	chart	S
θ	T	清齒擦音	thin	T
v	v	濁唇齒擦音	vest	f
w	w	圓唇軟顎近音	west	u
z	z	濁齒齦擦音	zero	s
ʒ	Z	濁齶後擦音	vision	S

IPA	X-SAMPA	描述	範例	視素
母音				
ə	@	中央母音	arena	@
ə̃	@`	中央 r 色彩母音	reader	@
æ	{	次開前不圓唇母音	trap	a
aɪ	al	雙母音	price	a
aʊ	aU	雙母音	mouth	a
ɑ	A	長開後不圓唇母音	father	a
eɪ	el	雙母音	face	e
ɜ̃	3`	半開央不圓唇 r 色彩母音	nurse	E
ɛ	E	半開前不圓唇母音	dress	E
i	i	長閉前不圓唇母音	fleece	i
ɪ	l	次閉次前不圓唇母音	kit	i
oʊ	oU	雙母音	goat	o
ɔ	O	長半開後圓唇母音	thought	O
ɔɪ	OI	雙母音	choice	O
u	u	長閉後圓唇母音	goose	u
ʊ	U	次閉次後圓唇母音	foot	u
ʌ	V	半開後不圓唇母音	strut	E
其他符號				

IPA	X-SAMPA	描述	範例	視素
'	"	主要重音	Alabama	
,	%	次要重音	Alabama	
.	.	音節邊界	A.la.ba.ma	

英文 (澳洲) (en-AU)

下表列出國際音標 (IPA) 音素、拓展音標字母評估法 (X-SAMPA) 符號，以及 Amazon Polly 支援的澳洲英文聲音對應視素。

音素/視素表

IPA	X-SAMPA	描述	範例	視素
子音				
b	b	濁雙唇塞音	bed	p
d	d	濁齒齶塞音	dig	t
ɹ̥dʒ	dʒ	濁顎齶塞擦音	jump	S
ð	D	濁齒擦音	then	T
f	f	清唇齒擦音	five	f
g	g	濁軟顎塞音	game	k
h	h	清聲門擦音	house	k
j	j	硬顎近音	yes	i
k	k	清軟顎塞音	cat	k
l	l	齒齶邊近音	lay	t
ɫ	l=	音節齒齶邊近音	battle	t

IPA	X-SAMPA	描述	範例	視素
m	m	雙唇鼻音	mouse	p
ᵿ	m=	音節雙唇鼻音	anthem	p
n	n	齒齶鼻音	nap	t
ṇ	n=	音節齒齶鼻音	button	t
ŋ	N	軟顎鼻音	thing	k
p	p	清雙唇塞音	pin	p
ɹ	r\	齒齶近音	red	r
s	s	清齒齶擦音	seem	s
ʃ	S	清齶後擦音	ship	S
t	t	清齒齶塞音	task	t
tʃ	tS	清齶後塞擦音	chart	S
θ	T	清齒擦音	thin	T
v	v	濁唇齒擦音	vest	f
w	w	圓唇軟顎近音	west	u
z	z	濁齒齶擦音	zero	s
ʒ	Z	濁齶後擦音	vision	S
母音				
ə	@	中央母音	arena	@
əʊ	@U	雙母音	goat	@
æ	{	次開前不圓唇母音	trap	a

IPA	X-SAMPA	描述	範例	視素
aɪ	al	雙母音	price	a
aʊ	aU	雙母音	mouth	a
ɑ:	A:	長開後不圓唇母音	father	a
eɪ	el	雙母音	face	e
ɜ:	3:	長半開央不圓唇音	nurse	E
ɛ	E	半開前不圓唇母音	dress	E
ɛə	E@	雙母音	square	E
i:	i	長閉前不圓唇母音	fleece	i
ɪ	l	次閉次前不圓唇母音	kit	i
ɪə	l@	雙母音	near	i
ɔ:	OI	長半開後圓唇母音	thought	O
ɔɪ	OI	雙母音	choice	O
ɒ	Q	開後圓唇母音	lot	O
u:	u:	長閉後圓唇母音	goose	u
ʊ	U	次閉次後圓唇母音	foot	u
ʊə	U@	雙母音	cure	u
ʌ	V	半開後不圓唇母音	strut	E
其他符號				
'	"	主要重音	Alabama	
,	%	次要重音	Alabama	

IPA	X-SAMPA	描述	範例	視素
.	.	音節邊界	A.la.ba.ma	

英文 (英國) (en-GB)

下表列出國際音標 (IPA) 音素、拓展音標字母評估法 (X-SAMPA) 符號，以及 Amazon Polly 所支援英國英文聲音對應的視素。

音素/視素表

IPA	X-SAMPA	描述	範例	視素
子音				
b	b	濁雙唇塞音	bed	p
d	d	濁齒齦塞音	dig	t
dʒ	dZ	濁顎齦塞擦音	jump	S
ð	D	濁齒擦音	then	T
f	f	清唇齒擦音	five	f
g	g	濁軟顎塞音	game	k
h	h	清聲門擦音	house	k
j	j	硬顎近音	yes	i
k	k	清軟顎塞音	cat	k
l	l	齒齦邊近音	lay	t
ɫ	l=	音節齒齦邊近音	battle	t
m	m	雙唇鼻音	mouse	p
ɱ	m=	音節雙唇鼻音	anthem	p

IPA	X-SAMPA	描述	範例	視素
n	n	齒齦鼻音	nap	t
ŋ	n=	音節齒齦鼻音	button	t
ŋ	N	軟顎鼻音	thing	k
p	p	清雙唇塞音	pin	p
ɹ	r\	齒齦近音	red	r
s	s	清齒齦擦音	seem	s
ʃ	S	清齶後擦音	ship	S
t	t	清齒齦塞音	task	t
tʃ	tS	清齶後塞擦音	chart	S
θ	T	清齒擦音	thin	T
v	v	濁唇齒擦音	vest	f
w	w	圓唇軟顎近音	west	u
z	z	濁齒齦擦音	zero	s
ʒ	Z	濁齶後擦音	vision	S
母音				
ə	@	中央母音	arena	@
əʊ	@U	雙母音	goat	@
æ	{	次開前不圓唇母音	trap	a
aɪ	al	雙母音	price	a
aʊ	aU	雙母音	mouth	a

IPA	X-SAMPA	描述	範例	視素
ɑ:	A:	長開後不圓唇母音	father	a
eɪ	eɪ	雙母音	face	e
ɜ:	3:	長半開央不圓唇音	nurse	E
ɛ	E	半開前不圓唇母音	dress	E
ɛə	E@	雙母音	square	E
i:	i	長閉前不圓唇母音	fleece	i
ɪ	ɪ	次閉次前不圓唇母音	kit	i
ɪə	ɪ@	雙母音	near	i
ɔ:	O:	長半開後圓唇母音	thought	O
ɔɪ	Oɪ	雙母音	choice	O
ɒ	Q	開後圓唇母音	lot	O
u:	u:	長閉後圓唇母音	goose	u
ʊ	U	次閉次後圓唇母音	foot	u
ʊə	U@	雙母音	cure	u
ʌ	V	半開後不圓唇母音	strut	E
其他符號				
'	"	主要重音	Alabama	
,	%	次要重音	Alabama	
.	.	音節邊界	A.la.ba.ma	

英文 (印度) (en-IN)

下表列出國際音標 (IPA) 音素、拓展音標字母評估法 (X-SAMPA) 符號，以及 Amazon Polly 所支援印度英文聲音對應的視素。

如需搭配印度英文使用之其他音素的詳細資訊，請參閱[印地語 \(hi-IN\)](#)。

音素/視素表

IPA	X-SAMPA	描述	範例	視素
子音				
b	b	濁雙唇塞音	bed	p
d	d	濁齒齦塞音	dig	t
ɖʒ	dʒ	濁顎齦塞擦音	jump	S
ð	D	濁齒擦音	then	T
f	f	清唇齒擦音	five	f
g	g	濁軟顎塞音	game	k
h	h	清聲門擦音	house	k
j	j	硬顎近音	yes	i
k	k	清軟顎塞音	cat	k
l	l	齒齦邊近音	lay	t
ɭ	l=	音節齒齦邊近音	battle	t
m	m	雙唇鼻音	mouse	p
ɱ	m=	音節雙唇鼻音	anthem	p
n	n	齒齦鼻音	nap	t
ɳ	n=	音節齒齦鼻音	nap	t

IPA	X-SAMPA	描述	範例	視素
ŋ	N	軟顎鼻音	thing	k
p	p	清雙唇塞音	pin	p
ɹ	r\	齒齶近音	red	r
s	s	清齒齶擦音	seem	s
ʃ	S	清齶後擦音	ship	S
t	t	清齒齶塞音	task	t
ʧ	tS	清齶後塞擦音	chart	S
θ	T	清齒擦音	thin	T
v	v	濁唇齒擦音	vest	f
w	w	圓唇軟顎近音	west	u
z	z	濁齒齶擦音	zero	s
ʒ	Z	濁齶後擦音	vision	S
母音				
ə	@	中央母音	arena	@
əʊ	@U	雙母音	goat	@
æ	{	次開前不圓唇母音	trap	a
aɪ	al	雙母音	price	a
aʊ	aU	雙母音	mouth	a
ɑ:	A:	長開後不圓唇母音	father	a
eɪ	el	雙母音	face	e

IPA	X-SAMPA	描述	範例	視素
ɜ:	3:	長半開央不圓唇音	nurse	E
ɛ	E	半開前不圓唇母音	dress	E
ɛə	E@	雙母音	square	E
i:	i	長閉前不圓唇母音	fleece	i
ɪ	ɪ	次閉次前不圓唇母音	kit	i
ɪə	ɪ@	雙母音	near	i
ɔ:	Oɪ	長半開後圓唇母音	thought	O
ɔɪ	Oɪ	雙母音	choice	O
ɒ	Q	開後圓唇母音	lot	O
u:	u:	長閉後圓唇母音	goose	u
ʊ	U	次閉次後圓唇母音	foot	u
ʊə	U@	雙母音	cure	u
ʌ	V	半開後不圓唇母音	strut	E
其他符號				
'	"	主要重音	Alabama	
,	%	次要重音	Alabama	
.	.	音節邊界	A.la.ba.ma	

英文（愛爾蘭）(en-IE)

下表列出國際音標 (IPA) 音素、拓展音標字母評估法 (X-SAMPA) 符號，以及 Amazon Polly 所支援愛爾蘭英文聲音對應的視素。

音素/視素表

IPA	X-SAMPA	描述	範例	視素
子音				
b	b	濁雙唇塞音	bed	p
d	d	濁齒齦塞音	dig	t
ɹ̥dʒ	dʒ	濁顎齦塞擦音	jump	S
ð	D	濁齒擦音	then	T
f	f	清唇齒擦音	five	f
g	g	濁軟顎塞音	game	k
h	h	清聲門擦音	house	k
j	j	硬顎近音	yes	i
k	k	清軟顎塞音	cat	k
l	l	齒齦邊近音	lay	t
m	m	雙唇鼻音	mouse	p
n	n	齒齦鼻音	nap	t
ŋ	N	軟顎鼻音	thing	k
p	p	清雙唇塞音	speak	p
ɹ	r\	齒齦近音	red	r
s	s	清齒齦擦音	seem	s
ʃ	S	清齶後擦音	ship	S
t	t	清齒齦塞音	trap	t
tʃ	tS	清齶後塞擦音	chart	S

IPA	X-SAMPA	描述	範例	視素
θ	T	清齒擦音	thin	T
v	v	濁唇齒擦音	vest	f
w	w	圓唇軟顎近音	west	u
z	z	濁齒齶擦音	zero	s
ʒ	Z	濁齶後擦音	vision	S
母音				
ə	@	中央母音	arena	@
æ̃	@`	中央 r 色彩母音	reader	@
æ	{	次開前不圓唇母音	trap	a
aɪ	al	雙母音	price	a
aʊ	aU	雙母音	mouth	a
ɑ	A	長開後不圓唇母音	father	a
eɪ	el	雙母音	face	e
ɜ̃	3`	半開央不圓唇 r 色彩母音	nurse	E
ɛ	E	半開前不圓唇母音	dress	E
i	i	長閉前不圓唇母音	fleece	i
ɪ	l	次閉次前不圓唇母音	kit	i
oʊ	oU	雙母音	goat	o
ɔ	O	長半開後圓唇母音	thought	O

IPA	X-SAMPA	描述	範例	視素
ɔɪ	Oɪ	雙母音	choice	O
u	u	長閉後圓唇母音	goose	u
ʊ	U	次閉次後圓唇母音	foot	u
ʌ	V	半開後不圓唇母音	strut	E
其他符號				
'	"	主要重音	Alabama	
,	%	次要重音	Alabama	
.	.	音節邊界	A.la.ba.ma	

英文（紐西蘭）(en-NZ)

下表列出國際音標 (IPA) 音素、拓展音標字母評估法 (X-SAMPA) 符號，以及 Amazon Polly 所支援紐西蘭英文聲音對應的視素。

音素/視素表

IPA	X-SAMPA	描述	範例	視素
子音				
b	b	濁雙唇塞音	bed	p
d	d	濁齒齦塞音	dig	t
ɹ̥dʒ	dʒ	濁顎齦塞擦音	jump	S
ð	D	濁齒擦音	then	T
f	f	清唇齒擦音	five	f
g	g	濁軟顎塞音	game	k

IPA	X-SAMPA	描述	範例	視素
h	h	清聲門擦音	house	k
j	j	硬顎近音	yes	i
k	k	清軟顎塞音	cat	k
l	l	齒齦邊近音	lay	t
ɭ	l=	音節齒齦邊近音	battle	t
m	m	雙唇鼻音	mouse	p
ṁ	m=	音節雙唇鼻音	anthem	p
n	n	齒齦鼻音	nap	t
ṅ	n=	音節齒齦鼻音	button	t
ŋ	N	軟顎鼻音	thing	k
p	p	清雙唇塞音	pin	p
ɹ	r\	齒齦近音	red	r
s	s	清齒齦擦音	seem	s
ʃ	S	清齶後擦音	ship	S
t	t	清齒齦塞音	task	t
ʧ	tS	清齶後塞擦音	chart	S
θ	T	清齒擦音	thin	T
v	v	濁唇齒擦音	vest	f
w	w	圓唇軟顎近音	west	u
z	z	濁齒齦擦音	zero	s

IPA	X-SAMPA	描述	範例	視素
ʒ	Z	濁齶後擦音	vision	S
母音				
ə	@	中央母音	arena	@
əʊ	@U	雙母音	goat	@
æ	{	次開前不圓唇母音	trap	a
aɪ	al	雙母音	price	a
aʊ	aU	雙母音	mouth	a
ɑ:	A:	長開後不圓唇母音	father	a
eɪ	el	雙母音	face	e
ɜ:	3:	長半開央不圓唇音	nurse	E
ɛ	E	半開前不圓唇母音	dress	E
ɛə	E@	雙母音	square	E
i:	i	長閉前不圓唇母音	fleece	i
ɪ	l	次閉次前不圓唇母音	kit	i
ɪə	l@	雙母音	near	i
ɔ:	O:	長半開後圓唇母音	thought	O
ɔɪ	Oɪ	雙母音	choice	O
ɒ	Q	開後圓唇母音	lot	O
u:	u:	長閉後圓唇母音	goose	u
ʊ	U	次閉次後圓唇母音	foot	u

IPA	X-SAMPA	描述	範例	視素
ʊə	U@	雙母音	cure	u
ʌ	V	半開後不圓唇母音	strut	E
其他符號				
'	"	主要重音	Alabama	
,	%	次要重音	Alabama	
.	.	音節邊界	A.la.ba.ma	

Aria 語音會說紐西蘭英文，並提供 Maori 的有限支援。它可以發音以下毛利語彙和片語。Maori 片語區分大小寫。

英文	毛利
Hello/cheers	Kia ora
歡迎（收件人）	Nau mai (ki)
您好（一個人）/謝謝您	Tēnā koe
您好（三人或更多人）/謝謝您	Tēnā koutou
早安	Ata mārie
早安	Mōrena
謝謝您	Ngā mihi
請小心	Ngā Manaakitanga
見	Ka 風箏
稍後見	Mā te wā
祝您有個愉快的一天	Kia pai tō rā

英文	毛利
Merry 聖誕	Meri Kirihimete
毛利	毛利
毛利語言	te reo Māori
毛利語言週	Te wiki o te reo Māori
紐西蘭	阿杜卡羅拉
毛利新年	Mātariki
紐西蘭的 Town / Waitangi Day 是紐西蘭的國定日	Waitangi
一	太極
Two	rua
三個	toru
四個	whā
五	利馬
六個	ono
七	魏爾圖
八個	waru
九	iwa
十	特考爾文
二十	rua tekau
三十	Toru tekau

英文（新加坡）(en-SG)

下表列出國際音標 (IPA) 音素、拓展音標字母評估法 (X-SAMPA) 符號，以及 Amazon Polly 支援的新加坡英文聲音對應的視素。

音素/視素表

IPA	X-SAMPA	描述	範例	視素
子音				
b	b	濁雙唇塞音	bed	p
d	d	濁齒齦塞音	dig	t
ɖʒ	dʒ	濁顎齦塞擦音	jump	S
ð	D	濁齒擦音	then	T
f	f	清唇齒擦音	five	f
g	g	濁軟顎塞音	game	k
h	h	清聲門擦音	house	k
j	j	硬顎近音	yes	i
k	k	清軟顎塞音	cat	k
l	l	齒齦邊近音	lay	t
ɭ	l=	音節齒齦邊近音	battle	t
m	m	雙唇鼻音	mouse	p
ɱ	m=	音節雙唇鼻音	anthem	p
n	n	齒齦鼻音	nap	t
ɳ	n=	音節齒齦鼻音	button	t
ŋ	N	軟顎鼻音	thing	k

IPA	X-SAMPA	描述	範例	視素
p	p	清雙唇塞音	pin	p
ɹ	r\	齒齶近音	red	r
s	s	清齒齶擦音	seem	s
ʃ	S	清齶後擦音	ship	S
t	t	清齒齶塞音	task	t
tʃ	tS	清齶後塞擦音	chart	S
θ	T	清齒擦音	thin	T
v	v	濁唇齒擦音	vest	f
w	w	圓唇軟顎近音	west	u
z	z	濁齒齶擦音	zero	s
ʒ	Z	濁齶後擦音	vision	S

母音

ə	@	中央母音	arena	@
əʊ	@U	雙母音	goat	@
æ	{	次開前不圓唇母音	trap	a
aɪ	al	雙母音	price	a
aʊ	aU	雙母音	mouth	a
ɑ:	A:	長開後不圓唇母音	father	a
eɪ	el	雙母音	face	e
ɜ:	3:	長半開央不圓唇音	nurse	E

IPA	X-SAMPA	描述	範例	視素
ɛ	E	半開前不圓唇母音	dress	E
ɛə	E@	雙母音	square	E
i:	i	長閉前不圓唇母音	fleece	i
ɪ	ɪ	次閉次前不圓唇母音	kit	i
ɪə	ɪ@	雙母音	near	i
ɔ:	O:	長半開後圓唇母音	thought	O
ɔɪ	Oɪ	雙母音	choice	O
ɒ	Q	開後圓唇母音	lot	O
u:	u:	長閉後圓唇母音	goose	u
ʊ	U	次閉次後圓唇母音	foot	u
ʊə	U@	雙母音	cure	u
ʌ	V	半開後不圓唇母音	strut	E
其他符號				
'	"	主要重音	Alabama	
,	%	次要重音	Alabama	
.	.	音節邊界	A.la.ba.ma	

英文（南非）(en-ZA)

下表列出國際音標 (IPA) 音素、拓展音標字母評估法 (X-SAMPA) 符號，以及 Amazon Polly 所支援南非英文聲音對應的視素。

音素/視素表

IPA	X-SAMPA	描述	範例	視素
子音				
b	b	濁雙唇塞音	bed	p
d	d	濁齒齦塞音	dig	t
ɖʒ	dʒ	濁顎齦塞擦音	jump	S
ð	D	濁齒擦音	then	T
f	f	清唇齒擦音	five	f
g	g	濁軟顎塞音	game	k
h	h	清聲門擦音	house	k
j	j	硬顎近音	yes	i
k	k	清軟顎塞音	cat	k
l	l	齒齦邊近音	lay	t
ɫ	l=	音節齒齦邊近音	battle	t
ʘ	K	無聲橫向仿造	umhl anga	t
m	m	雙唇鼻音	mouse	p
ɱ	m=	音節雙唇鼻音	anthem	p
n	n	齒齦鼻音	nap	t
ɳ	n=	音節齒齦鼻音	button	t
ŋ	N	軟顎鼻音	thing	k
p	p	清雙唇塞音	pin	p
ɹ	r\	齒齦近音	red	r

IPA	X-SAMPA	描述	範例	視素
r	r	齒齶顫音	分頁 eis	r
s	s	清齒齶擦音	seem	s
ʃ	S	清齶後擦音	ship	S
t	t	清齒齶塞音	task	t
ʧ	tS	清齶後塞擦音	chart	S
θ	T	清齒擦音	thin	T
v	v	濁唇齒擦音	vest	f
w	w	圓唇軟顎近音	west	u
x	x	清軟顎擦音	g auteng	k
z	z	濁齒齶擦音	zero	s
!	!\	後聲道點擊	gq eberha	k
	\	dental click	nc 立方體	t
	\	橫向按一下	xh osa	t
母音				
ə	@	中央母音	Rena	@
ɪɪ	@i	雙母音	nelsprui t	i
əʊ	@U	雙母音	goat	@
æ	{	次開前不圓唇母音	trap	a
aɪ	al	雙母音	price	a
aʊ	aU	雙母音	mouth	a

IPA	X-SAMPA	描述	範例	視素
ɑ:	A:	長開後不圓唇母音	father	a
eɪ	el	雙母音	face	e
ɜ:	3:	長半開央不圓唇音	nurse	E
ɛ	E	半開前不圓唇母音	dress	E
ɛə	E@	雙母音	square	E
i:	i	長閉前不圓唇母音	fleece	i
i◊◊	l@	雙母音	du preez	i
ɪ	l	次閉次前不圓唇母音	kit	i
ɪə	l@	雙母音	near	i
ɔ:	O:	長半開後圓唇母音	thought	O
ɔɪ	Oɪ	雙母音	choice	O
ɒ	Q	開後圓唇母音	lot	O
u:	u:	長閉後圓唇母音	goose	u
ʊ	U	次閉次後圓唇母音	foot	u
ʊə	U@	雙母音	cure	u
ʌ	V	半開後不圓唇母音	strut	E
y	y	閉前圓唇母音	van vuu ren	u
其他符號				
'	"	主要重音	Alabama	
,	%	次要重音	Alabama	

IPA	X-SAMPA	描述	範例	視素
.	.	音節邊界	A.la.ba.ma	

英文 (威爾斯) (en-GB-WLS)

下表列出國際音標 (IPA) 音素、拓展音標字母評估法 (X-SAMPA) 符號，以及 Amazon Polly 所支援威爾斯英文聲音的對應視素。

音素/視素表

IPA	X-SAMPA	描述	範例	視素
子音				
b	b	濁雙唇塞音	bed	p
d	d	濁齒齦塞音	dig	t
dʒ	dZ	濁顎齦塞擦音	jump	S
ð	D	濁齒擦音	then	T
f	f	清唇齒擦音	five	f
g	g	濁軟顎塞音	game	k
h	h	清聲門擦音	house	k
j	j	硬顎近音	yes	i
k	k	清軟顎塞音	cat	k
l	l	齒齦邊近音	lay	t
ɫ	l=	音節齒齦邊近音	battle	t
m	m	雙唇鼻音	mouse	p
ɱ	m=	音節雙唇鼻音	anthem	p

IPA	X-SAMPA	描述	範例	視素
n	n	齒齶鼻音	nap	t
ŋ	n=	音節齒齶鼻音	nap	t
ŋ	N	軟顎鼻音	thing	k
p	p	清雙唇塞音	pin	p
ɹ	r\	齒齶近音	red	r
s	s	清齒齶擦音	seem	s
ʃ	S	清齶後擦音	ship	S
t	t	清齒齶塞音	task	t
tʃ	tS	清齶後塞擦音	chart	S
θ	T	清齒擦音	thin	T
v	v	濁唇齒擦音	vest	f
w	w	圓唇軟顎近音	west	u
z	z	濁齒齶擦音	zero	s
ʒ	Z	濁齶後擦音	vision	S
母音				
ə	@	中央母音	arena	@
əʊ	@U	雙母音	goat	@
æ	{	次開前不圓唇母音	trap	a
aɪ	al	雙母音	price	a
aʊ	aU	雙母音	mouth	a

IPA	X-SAMPA	描述	範例	視素
ɑ:	A:	長開後不圓唇母音	father	a
eɪ	eɪ	雙母音	face	e
ɜ:	3:	長半開央不圓唇音	nurse	E
ɛ	E	半開前不圓唇母音	dress	E
ɛə	E@	雙母音	square	E
i:	i	長閉前不圓唇母音	fleece	i
ɪ	ɪ	次閉次前不圓唇母音	kit	i
ɪə	ɪ@	雙母音	near	i
ɔ:	Oɪ	長半開後圓唇母音	thought	O
ɔɪ	Oɪ	雙母音	choice	O
ɒ	Q	開後圓唇母音	lot	O
u:	u:	長閉後圓唇母音	goose	u
ʊ	U	次閉次後圓唇母音	foot	u
ʊə	U@	雙母音	cure	u
ʌ	V	半開後不圓唇母音	strut	E
其他符號				
'	"	主要重音	Alabama	
,	%	次要重音	Alabama	
.	.	音節邊界	A.la.ba.ma	

芬蘭文 (fi-FI)

下表列出國際音標 (IPA) 音素、拓展音標字母評估法 (X-SAMPA) 符號，以及 Amazon Polly 所支援芬蘭文聲音對應的視素。

音素/視素表

IPA	X-SAMPA	描述	範例	視素
-----	---------	----	----	----

芬蘭國家/地區

p	p	清雙唇塞音	【p】 ankki	p
t	t	清齒齶塞音	【t】 alo	t
k	k	清軟顎塞音	【k】 aali	k
d	d	濁齒齶塞音	【d】 ata	t
s	s	清齒齶擦音	【s】 ali	s
h	h	清聲門擦音	【h】 attu	k
u	v\	語音周邊近似值	【v】 aivaquin	v
j	j	硬顎近音	【j】 oki	i
l	l	齒齶邊近音	【l】 異常	t
r	r	語音 alveolar trill	【r】 iita	r
m	m	雙唇鼻音	【m】 ato	p
n	n	齒齶鼻音	【n】 enää	t
ŋ	N	軟顎鼻音	he 【n】 ki	k

在貸款單字中找到的共義詞

b	b	濁雙唇塞音	【b】 ussi	p
---	---	-------	-----------------	---

IPA	X-SAMPA	描述	範例	視素
f	f	清唇齒擦音	【f】 irma	v
w	w	圓唇軟顎近音	【w】 iki	u
z	z	濁齒齶擦音	【z】 ulu	s
g	g	濁軟顎塞音	【g】 aala	k
ʃ	S	清齶後擦音	【sh】 akki	S
ʒ	Z	濁齶後擦音	【g】 enre	S
θ	T	清齒擦音	ear 【th】	T
ð	D	濁齒擦音	ei 【th】 er	T
短母音				
i	i	閉前不圓唇母音	k 【i】 lo	i
ɛ	E	半開前不圓唇母音	k 【e】 sä	E
æ	{	次開前不圓唇母音	k 【ä】 ly	A
y	y	閉前圓唇母音	k 【y】 lä	u
ø	2	關閉前中圓母音	p 【ö】 ly	O
u	u	閉後圓唇母音	k 【u】 lo	u
ɔ	O	開啟中後圓母音	k 【o】 lo	O
ɑ	A	開後不圓唇母音	k 【a】 la	A
長母音				
i:	i:	長閉前不圓唇母音	s 【ii】 li	i
ɛ:	E:	長開前中無環母音	【ee】 tu	E

IPA	X-SAMPA	描述	範例	視素
æ:	{:	接近前開無圓母音的長音	t 【äa】 llä	A
y:	y:	長閉前不圓唇母音	t 【yy】 li	u
ø:	2:	長閉前中圓母音	t 【öö】 lö	O
u:	u:	長閉後圓唇母音	t 【uu】 li	u
ɔ:	O:	長半開後圓唇母音	r 【oo】 li	O
ɑ:	A:	長開後不圓唇母音	k 【aa】 su	A
Diphthongs				
ɛi	Ei	diphthong	l 【ei】 pä	E
æi	{i	diphthong	【äi】 ti	A
ui	ui	diphthong	k 【ui】 n	u
◆◆i	Ai	diphthong	k 【ai】 kki	A
ɔi	Oi	diphthong	p 【oi】 ka	O
øi	2i	diphthong	s 【öi】 n	O
yi	yi	diphthong	l 【yi】 jy	u
◆◆u	Au	diphthong	s 【au】 na	A
ɔu	Ou	diphthong	k 【ou】 lu	O
沙烏	Eu	diphthong	r 【eu】 na	E
iu	iu	diphthong	v 【iu】 lu	i
æy	{y	diphthong	t 【äy】 nnä	A
øy	2y	diphthong	k 【öy】 hä	O

IPA	X-SAMPA	描述	範例	視素
哈  y	Ey	diphthong	pes 【ey】 tyä	E
iy	iy	diphthong	käär 【iy】 tyä	i
i  	iE	diphthong	t 【ie】	i
yø	y2	diphthong	【yö】	u
uExample uO		diphthong	t 【uo】	u
在英文貸款單中找到的 Vowel				
ɪ	ɪ	次閉次前不圓唇母音	b 【i】 t	i
ʊ	U	次閉次後圓唇母音	b 【oo】 k	u
ə	@	中央母音	【a】 bout	@
ʌ	V	半開後不圓唇母音	c 【u】 t	E

法文 (fr-FR)

下表列出國際音標 (IPA) 音素、拓展音標字母評估法 (X-SAMPA) 符號，以及 Amazon Polly 所支援法文聲音對應的視素。

音素/視素表

IPA	X-SAMPA	描述	範例	視素
子音				
b	b	濁雙唇塞音	boire	p
d	d	濁齒齦塞音	madame	t
f	f	清唇齒擦音	femme	f

IPA	X-SAMPA	描述	範例	視素
g	g	濁軟顎塞音	grand	k
ɥ	H	唇硬顎近音	bruit	u
j	j	硬顎近音	meilleur	i
k	k	清軟顎塞音	quatre	k
l	l	齒齶邊近音	malade	t
m	m	雙唇鼻音	maison	p
n	n	齒齶鼻音	astronome	t
ɲ	J	硬顎鼻音	baigner	J
ŋ	N	軟顎鼻音	parking	k
p	p	清雙唇塞音	pomme	p
ʁ	R	濁小舌擦音	amoureux	k
s	s	清齒齶擦音	santé	s
ʃ	S	清齶後擦音	chat	S
t	t	清齒齶塞音	téléphone	t
v	v	濁唇齒擦音	vrai	f
w	w	圓唇軟顎近音	soir	u
z	z	濁齒齶擦音	raison	s
ʒ	Z	濁齶後擦音	aubergine	S
母音				
ø	2	半閉前圓唇母音	deux	o

IPA	X-SAMPA	描述	範例	視素
œ	9	半開前圓唇母音	neuf	O
œ̃	9~	鼻半開前圓唇母音	brun	O
ə	@	中央母音	je	@
a	a	開前不圓唇母音	table	a
ã	A~	鼻開後不圓唇母音	camembert	a
e	e	半閉前不圓唇母音	marché	e
ɛ	E	半開前不圓唇母音	neige	E
ẽ	E~	鼻半開前不圓唇母音	sapin	E
i	i	閉前不圓唇母音	mille	i
o	o	半閉後圓唇母音	hôpital	o
ɔ	O	半開後圓唇母音	homme	O
õ	O~	鼻半開後圓唇母音	bon	O
u	u	閉後圓唇母音	sous	u
y	y	閉前圓唇母音	dur	u
其他符號				
'	"	主要重音	Alabama	
,	%	次要重音	Alabama	
.	.	音節邊界	A.la.ba.ma	

法文（比利時）(fr-BE)

下表列出國際音標 (IPA) 音素、拓展音標字母評估法 (X-SAMPA) 符號，以及 Amazon Polly 支援的比利時法文聲音對應的視素。

音素/視素表

IPA	X-SAMPA	描述	範例	視素
子音				
b	b	濁雙唇塞音	boire	p
d	d	濁齒齦塞音	madame	t
f	f	清唇齒擦音	femme	f
g	g	濁軟顎塞音	grand	k
ɥ	H	唇硬顎近音	bruit	u
j	j	硬顎近音	meilleur	i
k	k	清軟顎塞音	quatre	k
l	l	齒齦邊近音	malade	t
m	m	雙唇鼻音	maison	p
n	n	齒齦鼻音	astronome	t
ɲ	J	硬顎鼻音	baigner	J
ŋ	N	軟顎鼻音	parking	k
p	p	清雙唇塞音	pomme	p
ʁ	R	濁小舌擦音	amoureux	k
s	s	清齒齦擦音	santé	s
ʃ	S	清齶後擦音	chat	S

IPA	X-SAMPA	描述	範例	視素
t	t	清齒齶塞音	téléphone	t
v	v	濁唇齒擦音	vrai	f
w	w	圓唇軟顎近音	soir	u
z	z	濁齒齶擦音	raison	s
ʒ	Z	濁齶後擦音	aubergine	S
母音				
ø	2	半閉前圓唇母音	deux	o
œ	9	半開前圓唇母音	neuf	O
œ̃	9~	鼻半開前圓唇母音	brun	O
ə	@	中央母音	je	@
a	a	開前不圓唇母音	table	a
ã	A~	鼻開後不圓唇母音	camembert	a
e	e	半閉前不圓唇母音	marché	e
ɛ	E	半開前不圓唇母音	neige	E
ẽ	E~	鼻半開前不圓唇母音	sapin	E
i	i	閉前不圓唇母音	mille	i
o	o	半閉後圓唇母音	hôpital	o
ɔ	O	半開後圓唇母音	homme	O
õ	O~	鼻半開後圓唇母音	bon	O
u	u	閉後圓唇母音	sous	u

IPA	X-SAMPA	描述	範例	視素
y	y	閉前圓唇母音	dur	u

其他符號

'	"	主要重音	Alabama	
,	%	次要重音	Alabama	
.	.	音節邊界	A.la.ba.ma	

法文 (加拿大) (fr-CA)

下表列出國際音標 (IPA) 音素、拓展音標字母評估法 (X-SAMPA) 符號，以及 Amazon Polly 所支援加拿大法文聲音對應的視素。

音素/視素表

IPA	X-SAMPA	描述	範例	視素
子音				
b	b	濁雙唇塞音	boire	p
d	d	濁齒齦塞音	madame	t
f	f	清唇齒擦音	femme	f
g	g	濁軟顎塞音	grand	k
ɥ	H	唇硬顎近音	bruit	u
j	j	硬顎近音	meilleur	i
k	k	清軟顎塞音	quatre	k
l	l	齒齦邊近音	malade	t
m	m	雙唇鼻音	maison	p

IPA	X-SAMPA	描述	範例	視素
n	n	齒齦鼻音	astronome	t
ɲ	J	硬顎鼻音	baigner	J
ŋ	N	軟顎鼻音	parking	k
p	p	清雙唇塞音	pomme	p
ʁ	R	濁小舌擦音	amoureux	k
s	s	清齒齦擦音	santé	s
ʃ	S	清齶後擦音	chat	S
t	t	清齒齦塞音	téléphone	t
v	v	濁唇齒擦音	vrai	f
w	w	圓唇軟顎近音	soir	u
z	z	濁齒齦擦音	raison	s
ʒ	Z	濁齶後擦音	aubergine	S
母音				
ø	2	半閉前圓唇母音	deux	o
œ	9	半開前圓唇母音	neuf	O
œ̃	9~	鼻半開前圓唇母音	brun	O
ə	@	中央母音	je	@
a	a	開前不圓唇母音	table	a
ɑ̃	A~	鼻開後不圓唇母音	camembert	a
e	e	半閉前不圓唇母音	marché	e

IPA	X-SAMPA	描述	範例	視素
ɛ	E	半開前不圓唇母音	neige	E
ẽ	E~	鼻半開前不圓唇母音	sapin	E
i	i	閉前不圓唇母音	mille	i
o	o	半閉後圓唇母音	hôpital	o
ɔ	O	半開後圓唇母音	homme	O
õ	O~	鼻半開後圓唇母音	bon	O
u	u	閉後圓唇母音	sous	u
y	y	閉前圓唇母音	dur	u

其他符號

'	"	主要重音	Alabama	
,	%	次要重音	Alabama	
.	.	音節邊界	A.la.ba.ma	

德文 (de-DE)

下表列出國際音標 (IPA) 音素、拓展音標字母評估法 (X-SAMPA) 符號，以及 Amazon Polly 所支援德文聲音對應的視素。

音素/視素表

IPA	X-SAMPA	描述	範例	視素
子音				
ʔ	ʔ	聲門塞音		

IPA	X-SAMPA	描述	範例	視素
b	b	濁雙唇塞音	Bier	p
d	d	濁齒齦塞音	Dach	t
ç	C	清硬顎擦音	ich	k
ɖʒ	dʒ	濁顎齦塞擦音	Dschungel	S
f	f	清唇齒擦音	Vogel	f
g	g	濁軟顎塞音	Gabel	k
h	h	清聲門擦音	Haus	k
j	j	清聲門擦音	jemand	i
k	k	清軟顎塞音	Kleid	k
l	l	齒齦邊近音	Loch	t
m	m	雙唇鼻音	Milch	p
n	n	齒齦鼻音	Natur	t
ŋ	N	軟顎鼻音	klingen	k
p	p	清雙唇塞音	Park	p
ɸf	pf	清唇齒塞擦音	Apfel	
ʀ	R	小舌顫音	Regen	
s	s	清齒齦擦音	Messer	s
ʃ	S	清齶後擦音	Fischer	S
t	t	清齒齦塞音	Topf	T
ʦ	Ts	清齒齦塞擦音	Zahl	

IPA	X-SAMPA	描述	範例	視素
ʦ	tS	清齶後塞擦音	deutsch	S
v	v	濁唇齒擦音	Wasser	f
x	x	清軟顎擦音	kochen	k
z	z	濁齒齶擦音	See	s
ʒ	Z	濁齶後擦音	Orange	S
母音				
ø:	2:	長半閉前圓唇母音	böse	o
ɐ	6	次開央母音	besser	a
ɐ̯	6_^	非音節次開央母音	Klar	a
œ	9	半開前圓唇母音	können	O
ə	@	中央母音	Rede	@
a	a	開前不圓唇母音	Salz	a
a:	a:	長開前不圓唇母音	Sahne	a
aɪ	al	雙母音	nein	a
aʊ	aU	雙母音	Augen	a
ã	A~	鼻開後不圓唇母音	Restaurant	a
e:	e:	長半閉前不圓唇母音	Rede	e
ɛ	E	半開前不圓唇母音	Keller	E
ẽ	E~	鼻半開前不圓唇母音	Terrain	E

IPA	X-SAMPA	描述	範例	視素
i:	i:	長閉前不圓唇母音	Lied	i
ɪ	ɪ	次閉次前不圓唇母音	bitte	i
o:	o:	長半閉後圓唇母音	Kohl	o
ɔ	O	半開後圓唇母音	Koffer	O
õ	O~	鼻半開後圓唇母音	Annonce	O
ɔʏ	OY	雙母音	neu	O
u:	u:	長閉後圓唇母音	Bruder	u
ʊ	U	次閉次後圓唇母音	Wunder	u
y:	y:	長閉前圓唇母音	kühl	u
ʏ	Y	次閉次前圓唇母音	Küche	u
其他符號				
'	"	主要重音	Alabama	
,	%	次要重音	Alabama	
.	.	音節邊界	A.la.ba.ma	

德文（奧地利）(de-AT)

下表列出國際音標 (IPA) 音素、拓展音標字母評估法 (X-SAMPA) 符號，以及 Amazon Polly 支援的奧地利德文聲音對應的視素。

音素/視素表

IPA	X-SAMPA	描述	範例	視素
子音				
ʔ	ʔ	聲門塞音		
b	b	濁雙唇塞音	Bier	p
d	d	濁齒齦塞音	Dach	t
ç	C	清硬顎擦音	ich	k
ɖʒ	dZ	濁顎齦塞擦音	Dschungel	S
f	f	清唇齒擦音	Vogel	f
g	g	濁軟顎塞音	Gabel	k
h	h	清聲門擦音	Haus	k
j	j	清聲門擦音	jemand	i
k	k	清軟顎塞音	Kleid	k
l	l	齒齦邊近音	Loch	t
m	m	雙唇鼻音	Milch	p
n	n	齒齦鼻音	Natur	t
ŋ	N	軟顎鼻音	klingen	k
p	p	清雙唇塞音	Park	p
ɸf	pf	清唇齒塞擦音	Apfel	
ʀ	R	小舌顫音	Regen	
s	s	清齒齦擦音	Messer	s
ʃ	S	清齦後擦音	Fischer	S

IPA	X-SAMPA	描述	範例	視素
t	t	清齒齶塞音	Topf	T
ʈs	Ts	清齒齶塞擦音	Zahl	
ʈʃ	tS	清齶後塞擦音	deutsch	S
v	v	濁唇齒擦音	Wasser	f
x	x	清軟顎擦音	kochen	k
z	z	濁齒齶擦音	See	s
ʒ	Z	濁齶後擦音	Orange	S
母音				
ø:	2:	長半閉前圓唇母音	böse	o
ɐ	6	次開央母音	besser	a
ɐ̯	6_^	非音節次開央母音	Klar	a
œ	9	半開前圓唇母音	können	O
ə	@	中央母音	Rede	@
a	a	開前不圓唇母音	Salz	a
a:	a:	長開前不圓唇母音	Sahne	a
aɪ	al	雙母音	nein	a
aʊ	aU	雙母音	Augen	a
ã	A~	鼻開後不圓唇母音	Restaurant	a
e:	e:	長半閉前不圓唇母音	Rede	e
ɛ	E	半開前不圓唇母音	Keller	E

IPA	X-SAMPA	描述	範例	視素
ẽ	E~	鼻半開前不圓唇母音	Terrain	E
i:	i:	長閉前不圓唇母音	Lied	i
ɪ	ɪ	次閉次前不圓唇母音	bitte	ɪ
o:	o:	長半閉後圓唇母音	Kohl	o
ɔ	O	半開後圓唇母音	Koffer	O
õ	O~	鼻半開後圓唇母音	Annonce	O
ɔʏ	OY	雙母音	neu	O
u:	u:	長閉後圓唇母音	Bruder	u
ʊ	U	次閉次後圓唇母音	Wunder	u
y:	y:	長閉前圓唇母音	kühl	u
ʏ	Y	次閉次前圓唇母音	Küche	u
其他符號				
'	"	主要重音	Alabama	
,	%	次要重音	Alabama	
.	.	音節邊界	A.la.ba.ma	

德文（瑞士標準）(de-CH)

下表列出國際音標 (IPA) 音素、拓展音標字母評估法 (X-SAMPA) 符號，以及 Amazon Polly 支援的德文（瑞士標準）聲音對應的視素。

音素/視素表

IPA	X-SAMPA	描述	範例	視素
子音				
ʔ	ʔ	聲門塞音		
b	b	濁雙唇塞音	Bier	p
d	d	濁齒齦塞音	Dach	t
ç	C	清硬顎擦音	ich	k
ɖʒ	dZ	濁顎齦塞擦音	Dschungel	S
f	f	清唇齒擦音	Vogel	f
g	g	濁軟顎塞音	Gabel	k
h	h	清聲門擦音	Haus	k
j	j	清聲門擦音	jemand	i
k	k	清軟顎塞音	Kleid	k
l	l	齒齦邊近音	Loch	t
m	m	雙唇鼻音	Milch	p
n	n	齒齦鼻音	Natur	t
ŋ	N	軟顎鼻音	klingen	k
p	p	清雙唇塞音	Park	p
pf	pf	清唇齒塞擦音	Apfel	
ʀ	R	小舌顫音	Regen	
s	s	清齒齦擦音	Messer	s
ʃ	S	清齶後擦音	Fischer	S

IPA	X-SAMPA	描述	範例	視素
t	t	清齒齶塞音	Topf	T
ʦ	Ts	清齒齶塞擦音	Zahl	
ʦ̥	tS	清齶後塞擦音	deutsch	S
v	v	濁唇齒擦音	Wasser	f
x	x	清軟顎擦音	kochen	k
z	z	濁齒齶擦音	See	s
ʒ	Z	濁齶後擦音	Orange	S
母音				
ø:	2:	長半閉前圓唇母音	böse	o
ɐ	6	次開央母音	besser	a
ɐ̯	6_^	非音節次開央母音	Klar	a
œ	9	半開前圓唇母音	können	O
ə	@	中央母音	Rede	@
a	a	開前不圓唇母音	Salz	a
a:	a:	長開前不圓唇母音	Sahne	a
aɪ	al	雙母音	nein	a
aʊ	aU	雙母音	Augen	a
ã	A~	鼻開後不圓唇母音	Restaurant	a
e:	e:	長半閉前不圓唇母音	Rede	e
ɛ	E	半開前不圓唇母音	Keller	E

IPA	X-SAMPA	描述	範例	視素
ẽ	E~	鼻半開前不圓唇母音	Terrain	E
i:	i:	長閉前不圓唇母音	Lied	i
ɪ	ɪ	次閉次前不圓唇母音	bitte	ɪ
o:	o:	長半閉後圓唇母音	Kohl	o
ɔ	O	半開後圓唇母音	Koffer	O
õ	O~	鼻半開後圓唇母音	Annonce	O
ɔʏ	OY	雙母音	neu	O
u:	u:	長閉後圓唇母音	Bruder	u
ʊ	U	次閉次後圓唇母音	Wunder	u
y:	y:	長閉前圓唇母音	kühl	u
ʏ	Y	次閉次前圓唇母音	Küche	u
其他符號				
'	“	主要重音	Alabama	
,	%	次要重音	Alabama	
.	.	音節邊界	A.la.ba.ma	

印地語 (hi-IN)

下表列出國際音標 (IPA) 音素、拓展音標字母評估法 (X-SAMPA) 符號，以及 Amazon Polly 所支援印地語聲音的音素類型。

如需搭配印地語使用的其他音素的詳細資訊，請參閱[英文 \(印度\) \(en-IN\)](#)。

音素/視素表

IPA	X-SAMPA	描述	範例
子音			
p ^h	p_h	清送氣雙唇塞音	फूल (phool)
b ^h	b_h	濁送氣雙唇塞音	भारी (bhaari)
t̪	t_d	清齒擦音	तापमान (taapmaan)
t̪ ^h	t_d_h	清送氣齒擦音	थोड़ा (thoda)
d̪	d_d	濁齒擦音	दिल्ली (dilli)
d̪ ^h	d_d_h	濁送氣齒擦音	धोबी (dhobi)
t̪	t`	清捲舌塞音	कटोरा (katora)
t̪ ^h	t`_h	清送氣捲舌塞音	ठंड (thand)
d̪	d`	濁捲舌塞音	डर (darr)
d̪ ^h	d`_h	濁送氣捲舌塞音	ढाल (dhal)
tʃ ^h	tS_h	清送氣塞擦音	छाल (chaal)
dʒ ^h	dZ_h	濁送氣塞擦音	झाल (jhaal)
k ^h	k_h	清送氣軟顎塞音	खान (khan)
g ^h	g_h	濁送氣軟顎塞音	घान (ghaan)
ŋ	n`	捲舌鼻音	क्षण (kshan)
r	ɽ	齒齦閃音	राम (ram)
ɽ	r`	平捲舌閃音	बड़ा (bada)
ɽ ^h	r`_h	濁送氣捲舌閃音	बढ़ी (barhi)
ʋ	v\	雙唇近音	वसूल (wasool)

IPA	X-SAMPA	描述	範例
母音			
ə	@_o	中央母音	अच्छा (achhaa)
ẽ	@~	鼻音化中央母音	हँसना (hansnaa)
a	A_o	開前不圓唇母音	आग (aag)
ã	A~	鼻音化開前不圓唇母音	घड़ियाँ (ghariyaan)
ɪ	I_o	次閉次前不圓唇母音	इक्कीस (ikkees)
ĩ	I~	鼻音化次閉次前不圓唇母音	संचिआई (sinchai)
i	i_o	閉前不圓唇母音	बिल्ली (billee)
ĩ	i~	鼻音化閉前不圓唇母音	नहीं (nahin)
ʊ	U_o	次閉次後圓唇母音	उलूल (ullu)
ũ	U~	鼻音化次閉次後圓唇母音	मुँह (munh)
u	u_o	閉後圓唇母音	फूल (phool)
ũ	u~	鼻音化閉後圓唇母音	ऊँट (oont)
ɔ	O_o	半開後圓唇母音	कौन (kaun)
õ	O~	鼻音化半開後圓唇母音	भौ (bhaun)
o	o	半閉後圓唇母音	सोना (sona)
õ	o~	鼻音化半閉後圓唇母音	क्यों (kyon)
ɛ	E_o	半開前不圓唇母音	पैसा (paisa)
ẽ	E~	鼻音化半開前不圓唇母音	मैं (main)
e	e	半閉前不圓唇母音	एक (ek)

IPA	X-SAMPA	描述	範例
ẽ	e~	鼻音化半閉前不圓唇母音	कतिबे (kitabein)

冰島文 (is-IS)

下表列出國際音標 (IPA) 音素、拓展音標字母評估法 (X-SAMPA) 符號，以及 Amazon Polly 所支援冰島文聲音對應的視素。

音素/視素表

IPA	X-SAMPA	描述	範例	視素
子音				
b	b	濁雙唇塞音	grasbakkanum	0
c	c	清硬顎塞音	pakkin	k
c ^h	c_h	送氣清硬顎塞音	anarkistai	k
ç	C	清硬顎擦音	héðan	k
d	d	濁齒齶塞音	bóndi	t
ð	D	濁齒擦音	borð	T
f	f	清唇齒擦音	duft	f
g	g	濁軟顎塞音	holgóma	k
ɣ	G	濁軟顎擦音	hugur	k
h	h	清聲門擦音	heili	k
j	j	硬顎近音	jökull	i
k ^h	k_h	送氣清軟顎塞音	ósköpunum	k
l	l	齒齶邊近音	gólf	t

IPA	X-SAMPA	描述	範例	視素
ɹ	l_0	清齒齦邊近音	fólk	t
m	m	雙唇鼻音	september	p
m̥	m_0	清雙唇鼻音	kompa	p
n	n	齒齦鼻音	númer	t
n̥	n_0	清齒齦鼻音	pöntun	t
ɲ	J	硬顎鼻音	pælingar	J
ŋ	N	軟顎鼻音	söngvarann	k
ŋ̥	N_0	清軟顎鼻音	frænka	k
pʰ	p_h	送氣清雙唇塞音	afplánun	p
r	r	齒齦顫音	afskrifta	r
ɾ	r_0	清齒齦顫音	andvörpum	r
s	s	清齒齦擦音	baðhús	s
tʰ	t_h	送氣清齒齦塞音	tanki	t
θ	T	清齒擦音	þeldökki	T
v	v	濁唇齒擦音	silfur	f
w	w	圓唇軟顎近音		u
x	x	清軟顎擦音	samfélags	k
母音				
œ	9	半開前圓唇母音	þröskuldinum	O
œ:	9:	長半開前圓唇母音	tvö	O

IPA	X-SAMPA	描述	範例	視素
a	a	開前不圓唇母音	nefna	a
a:	a:	長開前不圓唇母音	fara	a
au	au	雙母音	átta	a
au:	au:	雙母音	átján	a
ɛ	E	半開前不圓唇母音	kennari	E
ɛ:	E:	長半開前不圓唇母音	dreka	E
i	i	閉前不圓唇母音	Gúliver	i
i:	i:	長閉前不圓唇母音	þír	i
ɪ	ɪ	次閉次前不圓唇母音	samspil	i
ɪ:	ɪ:	長次閉次前不圓唇母音	stig	i
ɔ	O	半開後圓唇母音	regndropar	O
ɔ:	O:	長半開後圓唇母音	ullarbolur	O
ou	Ou	雙母音	tólf	O
ou:	Ou:	雙母音	fjórir	O
u	u	閉後圓唇母音	stúlkan	u
u:	u:	長閉後圓唇母音	frú	u
ʏ	Y	次閉次前圓唇母音	tíu	u
ʏ:	Y	長次閉次前圓唇母音	gruninn	u

IPA	X-SAMPA	描述	範例	視素
其他符號				
ˈ	"	主要重音	Alabama	
ˌ	%	次要重音	Alabama	
ˌ	.	音節邊界	A.la.ba.ma	

義大利文 (it-IT)

下表列出國際音標 (IPA) 音素、拓展音標字母評估法 (X-SAMPA) 符號，以及 Amazon Polly 支援的義大利文聲音對應的視素。

音素/視素表

IPA	X-SAMPA	描述	範例	視素
子音				
b	b	濁雙唇塞音	bacca	p
d	d	濁齒齦塞音	dama	t
ɖz	dz	濁齒齦塞擦音	zero	s
ɖʒ	dʒ	濁顎齦塞擦音	giro	S
f	f	清唇齒擦音	famiglia	f
g	g	濁軟顎塞音	gatto	k
h	h	清聲門擦音	horror	k
j	j	硬顎近音	dieci	i
k	k	清軟顎塞音	campo	k
l	l	齒齦邊近音	lido	t

IPA	X-SAMPA	描述	範例	視素
ʎ	L	硬顎邊近音	aglio	J
m	m	雙唇鼻音	mille	p
n	n	齒齦鼻音	nove	t
ɲ	J	硬顎鼻音	lasagne	J
p	p	清雙唇塞音	pizza	p
r	r	齒齦顫音	risata	r
s	s	清齒齦擦音	sei	s
ʃ	S	清齶後擦音	scienza	S
t	t	清齒齦塞音	tavola	t
ʦ	ts	清齒齦塞擦音	forza	s
tʃ	tS	清齶後塞擦音	cielo	S
v	v	濁唇齒擦音	venti	f
w	w	圓唇軟顎近音	quattro	u
z	z	濁齒齦擦音	bisogno	s
ʒ	Z	濁齶後擦音	bijou	S
母音				
a	a	開前不圓唇母音	arco	a
e	e	半閉前不圓唇母音	tre	e
ɛ	E	半開前不圓唇母音	ettaro	E
i	i	閉前不圓唇母音	impero	i

IPA	X-SAMPA	描述	範例	視素
o	o	半閉後圓唇母音	cento	o
ɔ	O	半開後圓唇母音	otto	O
u	u	閉後圓唇母音	uno	u

其他符號

'	"	主要重音	Alabama	
,	%	次要重音	Alabama	
.	.	音節邊界	A.la.ba.ma	

日文 (ja-JP)

Amazon Polly 支援日文的發音 Kana 和 Yomigana 字母。若要讓 Amazon Polly 搭配這些字母使用音標發音，請使用 `phoneme alphabet="x-amazon-phonetic standard used"` 屬性。

- `x-amazon-pron-kana` – 表示使用發音 Kana。發音 Kana 是用於音標轉錄的特殊 Katakana 字元，可以編碼音調重音。
- `x-amazon-yomigana` – 表示已使用 Yomigana。Yomigana 可以是傳統的 Katakana、Hiragana 和拉丁字母，解釋為 *hepburn romanization*。

下列範例示範如何使用這些範例：

發音 Kana

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

Yomigana

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

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

下表列出國際音標 (IPA) 音素、拓展音標字母評估法 (X-SAMPA) 符號，以及 Amazon Polly 所支援日文聲音對應的視素。

IPA	X-SAMPA	描述	範例	視素
子音				
r	4	齒齦閃音	練習，+renshuu	t
ʔ	ʔ	聲門塞音	あっつ，atsu '	
b	b	濁雙唇塞音	舞踊，buyou	p
β	B	濁雙唇擦音	ヴィンテージ，vinteeji	B
c	c	清硬顎塞音	ききょう，kikyou	k
ç	C	清硬顎擦音	人，hito	k
d	d	濁齒齦塞音	濁点，dakuten	t
ɖʑ	dz\	濁齶顎塞擦音	純，jun	J
g	g	濁軟顎塞音	ご飯，gohan	k
h	h	清聲門擦音	本，hon	k
j	j	硬顎近音	屋根，yane	i
ɟ	J\	濁硬顎塞音	行儀，gyougi	J
k	k	清軟顎塞音	漢字，kanji	k
ʃ	l\	齒齦邊閃音	釣り，tsuri	r
ʃj	lj	齒齦邊閃音、硬顎近音	流行，ryuukou	r

IPA	X-SAMPA	描述	範例	視素
m	m	雙唇鼻音	飯，meshi	p
n	n	齒齶鼻音	猫，neko	t
ɲ	J	硬顎鼻音	日本，nippon	J
ɳ	N\	小舌鼻音	缶，kan	k
p	p	清雙唇塞音	パン，pan	p
ɸ	p\	清雙唇擦音	福，huku	f
s	s	清齒齶擦音	層，sou	s
ɕ	s\	清硬顎擦音	書簡，shokan	J
t	t	清齒齶塞音	手紙，tegami	t
ʦ	ts	清齒齶塞擦音	釣り，tsuri	s
ʧ	ts\	清齶顎塞擦音	吉，kichi	J
w	w	圓唇軟顎近音	電話，denwa	u
z	z	濁齒齶擦音	座敷，zashiki	s
母音				
ä:	a:_"	長開央不圓唇母音	羽蟻，haari	a
ä	a_"	開央不圓唇母音	仮名，kana	a
e:	e:_o	長半開前不圓唇母音	学生，gakusei	@
e	e_o	半開前不圓唇母音	歴，reki	@
i	i	閉前不圓唇母音	気，ki	i
i:	i:	長閉前不圓唇母音	詩歌，shiika	i

IPA	X-SAMPA	描述	範例	視素
ʊ	M	閉後不圓唇母音	運, un	i
ʊ:	M:	長閉後不圓唇母音	宗教, shuukyou	i
o:	o:_o	長半後圓唇母音	購讀, koodoku	o
o	o_o	半後圓唇母音	讀者, dokusha	o

韓文 (ko-KR)

下表列出國際音標 (IPA) 音素、拓展音標字母評估法 (X-SAMPA) 符號，以及 Amazon Polly 所支援韓文聲音的對應視素。

IPA	X-SAMPA	描述	範例	視素
子音				
k	k	清軟顎塞音	강, [g]ang	k
k#	k_t	強清軟顎塞音	깨, [kk]e	k
n	n	齒齶鼻音	남, [n]am	t
t	t	清齒齶塞音	도, [d]o	t
t#	t_t	強清齒齶塞音	때, [tt]e	t
r	4	齒齶閃音	사랑, sa[r]ang	t
l	l	齒齶邊近音	돌, do[l]	t
m	m	雙唇鼻音	무, [m]u	p
p	p	清雙唇塞音	봄, [b]om	p
p#	p_t	強清雙唇塞音	빨, [pp]eol	p
s	s	清齒齶擦音	새, [s]e	s

IPA	X-SAMPA	描述	範例	視素
s#	s_t	強清齒齶擦音	씨, [ss]i	s
ŋ	N	軟顎鼻音	방, ba[ng]	k
ʈʂ	ts\	清齶顎塞擦音	조, [j]o	J
ʈʂʰ	ts_t	強清齶顎塞擦音	찌, [jj]i	J
ʈʂʰ	ts_h	送氣清齶顎塞擦音	차, [ch]a	J
kʰ	k_h	送氣清軟顎塞音	코, [k]o	k
tʰ	t_h	送氣清齒齶塞音	통, [t]ong	t
pʰ	p_h	送氣清雙唇塞音	패, [p]e	p
h	h	清聲門擦音	힘, [h]im	k
j	j	硬顎近音	양, [y]ang	i
w	w	圓唇軟顎近音	왕, [w]ang	u
ɰ	M\	軟顎近音>	의, [w]i	i
母音				
a	a	開前不圓唇母音	밥, b[a]b	a
ʌ	V	半開後不圓唇母音	정, j[eo]ng	E
ɛ	E	半開前不圓唇母音	배, b[e]	E
o	o	半閉後圓唇母音	노, n[o]	o
u	u	閉後圓唇母音	둘, d[u]l	u
ɯ	M	閉後不圓唇母音	은, [eu]n	i
i	i	閉前不圓唇母音	김, k[i]m	i

挪威文 (nb-NO)

下表列出完整的國際音標 (IPA) 音素和拓展音標字母評估法 (X-SAMPA) 符號，以及 Amazon Polly for Norwegian 語言聲音支援的對應視素。

IPA	X-SAMPA	描述	範例	視素
子音				
r	4	齒齶閃音	prøv	t
b	b	濁雙唇塞音	labb	p
ç	C	清硬顎擦音	kino	k
d	d	濁齒齶塞音	ladd	t
ɖ	d`	濁捲舌塞音	verdi	t
f	f	清唇齒擦音	fot	f
g	g	濁軟顎塞音	tagg	k
h	h	清聲門擦音	ha	k
j	j	硬顎近音	gi	i
k	k	清軟顎塞音	takk	k
l	l	齒齶邊近音	fall、ball	t
ɭ	l`	捲舌邊近音	ærlig	t
m	m	雙唇鼻音	lam	p
n	n	齒齶鼻音	vann	t
ɳ	n`	捲舌鼻音	garn	t
ŋ	N	軟顎鼻音	sang	k

IPA	X-SAMPA	描述	範例	視素
p	p	清雙唇塞音	hopp	p
s	s	清齒齶擦音	lass	s
ʃ	s`	清捲舌擦音	års	S
ʃ	S	清齶後擦音	skyt	S
t	t	清齒齶塞音	lat	t
t̪	t`	清捲舌塞音	hardt	t
u	v\	唇齒近音	vin	f
w	w	圓唇軟顎近音	will	x
母音				
ø:	2:	長半閉前圓唇母音	søt	o
œ	9	半開前圓唇母音	søtt	O
ə	@	中央母音	ape	@
æ:	{:	長次開前不圓唇母音	vær	a
ʊ	}	閉央圓唇母音	lund	u
ʊ:	}::	長閉央圓唇母音	lun	u
æ	{	次開前不圓唇母音	vært	a
ɑ	A	開後不圓唇母音	hatt	a
ɑ:	A:	長開後不圓唇母音	hat	a
e:	e:	長半閉前不圓唇母音	sen	e

IPA	X-SAMPA	描述	範例	視素
ɛ	E	半開前不圓唇母音	send	E
i:	i:	長閉前不圓唇母音	vin	i
ɪ	ɪ	次閉次前不圓唇母音	vind	ɪ
o:	o:	長半閉後圓唇母音	våt	o
ɔ	O	半開後圓唇母音	vått	O
u:	u:	長閉後圓唇母音	bok	u
ʊ	U	次閉次後圓唇母音	bukk	u
y:	y:	長閉前圓唇母音	lyn	u
ʏ	Y	次閉次前圓唇母音	lynne	u
其他符號				
'	"	主要重音	Alabama	
,	%	次要重音	Alabama	
.	.	音節邊界	A.la.ba.ma	

波蘭文 (pl-PL)

下表列出國際音標 (IPA) 音素、拓展音標字母評估法 (X-SAMPA) 符號，以及 Amazon Polly 所支援波蘭文聲音對應的視素。

音素/視素表

IPA	X-SAMPA	描述	範例	視素
子音				

IPA	X-SAMPA	描述	範例	視素
b	b	濁雙唇塞音	bobas、belka	p
d	d	濁齒齦塞音	dar、do	t
ᵈ̥	dz	濁齒齦塞擦音	dzwon、wiedzowie	s
ᵈ̥	dz\	濁齶塞擦音	dźwięk	J
ᵈ̥	dz`	濁捲舌塞擦音	dżem、dżungla	S
f	f	清唇齒擦音	furtka、film	f
g	g	濁軟顎塞音	gazeta、waga	k
h	h	清聲門擦音	chleb、handel	k
j	j	硬顎近音	jak、maja	i
k	k	清軟顎塞音	kura、marek	k
l	l	齒齦邊近音	lipa、alicja	t
m	m	雙唇鼻音	matka、molo	p
n	n	齒齦鼻音	norka	t
ɲ	J	硬顎鼻音	koń、toruń	J
p	p	清雙唇塞音	pora、stop	p
r	r	齒齦顫音	rok、park	r
s	s	清齒齦擦音	sum、pas	s
ʃ	s\	清硬顎擦音	śruba、śnieg	J
ʂ	s`	清捲舌擦音	szum、masz	S
t	t	清齒齦塞音	tok、stół	t

IPA	X-SAMPA	描述	範例	視素
ts	ts	清齒齶塞擦音	car、co	s
tʃ	ts\	清齶顎塞擦音	ćma、mieć	J
tʂ	ts`	清捲舌塞擦音	czas、raczej	S
v	v	濁唇齒擦音	worek、mewa	f
w	w	圓唇軟顎近音	łaska、mało	u
z	z	濁齒齶擦音	zero	s
ʒ	z\	濁齶顎擦音	żrebię、bieliźnie	J
ʐ	z`	濁捲舌擦音	żar、żona	S
母音				
a	a	開前不圓唇母音	ja	a
ɛ	E	半開前不圓唇母音	echo	E
ẽ	E~	鼻半開前不圓唇母音	węże	E
i	i	閉前不圓唇母音	ile	i
ɔ	O	半開後圓唇母音	oczy	O
õ	O~	鼻半開後圓唇母音	wąż	O
u	u	閉後圓唇母音	uczta	u
ɨ	1	閉央不圓唇母音	byk	i
其他符號				
'	"	主要重音	Alabama	
,	%	次要重音	Alabama	

IPA	X-SAMPA	描述	範例	視素
.	.	音節邊界	A.la.ba.ma	

葡萄牙文 (pt-PT)

下表列出國際音標 (IPA) 音素、拓展音標字母評估法 (X-SAMPA) 符號，以及 Amazon Polly 支援的葡萄牙文聲音對應的視素。

音素/視素表

IPA	X-SAMPA	描述	範例	視素
子音				
r	4	齒齶閃音	pira	t
b	b	濁雙唇塞音	dato	p
d	d	濁齒齶塞音	dato	t
f	f	清唇齒擦音	facto	f
g	g	濁軟顎塞音	gato	k
j	j	硬顎近音	paraguay	i
k	k	清軟顎塞音	cacto	k
l	l	齒齶邊近音	galo	t
ʎ	L	硬顎邊近音	galho	J
m	m	雙唇鼻音	mato	p
n	n	齒齶鼻音	nato	t
ɲ	J	硬顎鼻音	pinha	J
p	p	清雙唇塞音	pato	p

IPA	X-SAMPA	描述	範例	視素
ɾ	ɾ\	小舌顫音	barroso	k
s	s	清齒齶擦音	saca	s
ʃ	S	清齶後擦音	chato	S
t	t	清齒齶塞音	tacto	t
v	v	濁唇齒擦音	vaca	f
w	w	圓唇軟顎近音	mau	u
z	z	濁齒齶擦音	zaca	s
ʒ	Z	濁齶後擦音	jacto	S
母音				
a	a	開前不圓唇母音	parto	a
ã	a~	鼻開前不圓唇母音	pega	a
e	e	半閉前不圓唇母音	pega	e
ẽ	e~	鼻半閉前不圓唇母音	movem	e
ɛ	E	半開前不圓唇母音	café	E
i	i	閉前不圓唇母音	lingueta	i
ĩ	i~	鼻閉前不圓唇母音	cinto	i
o	o	半閉後圓唇母音	poder	o
õ	o~	鼻半閉後圓唇母音	compra	o
ɔ	O	半開後圓唇母音	cotó	O
u	u	閉後圓唇母音	fui	u

IPA	X-SAMPA	描述	範例	視素
ũ	u~	鼻閉後圓唇母音	sunto	u
其他符號				
'	"	主要重音	Alabama	
,	%	次要重音	Alabama	
.	.	音節邊界	A.la.ba.ma	

葡萄牙文 (巴西) (pt-BR)

下表列出國際音標 (IPA) 音素、拓展音標字母評估法 (X-SAMPA) 符號，以及 Amazon Polly 所支援巴西葡萄牙文聲音對應的視素。

音素/視素表

IPA	X-SAMPA	描述	範例	視素
子音				
r	4	齒齶閃音	pira	t
b	b	濁雙唇塞音	bato	p
d	d	濁齒齶塞音	dato	t
ɾ̃	dZ	濁顎齶塞擦音	idade	S
f	f	清唇齒擦音	facto	f
g	g	濁軟顎塞音	gato	k
j	j	硬顎近音	paraguay	i
k	k	清軟顎塞音	cacto	k
l	l	齒齶邊近音	galo	t

IPA	X-SAMPA	描述	範例	視素
ʎ	L	硬顎邊近音	galho	J
m	m	雙唇鼻音	mato	p
n	n	齒齦鼻音	nato	t
ɲ	J	硬顎鼻音	pinha	J
p	p	清雙唇塞音	pato	p
s	s	清齒齦擦音	saca	s
ʃ	S	清齶後擦音	chato	S
t	t	清齒齦塞音	tacto	t
ɥ	tS	清齶後塞擦音	noite	S
v	v	濁唇齒擦音	vaca	f
w	w	圓唇軟顎近音	mau	u
χ	X	清小舌擦音	carro	k
z	z	濁齒齦擦音	zaca	s
ʒ	Z	濁齶後擦音	jacto	S
母音				
a	a	開前不圓唇母音	parto	a
ã	a~	鼻開前不圓唇母音	pensamos	a
e	e	半閉前不圓唇母音	pega	e
ẽ	e~	鼻半閉前不圓唇母音	movem	e
ɛ	E	半開前不圓唇母音	café	E

IPA	X-SAMPA	描述	範例	視素
i	i	閉前不圓唇母音	lingueta	i
ĩ	i~	鼻閉前不圓唇母音	cinto	i
o	o	半閉後圓唇母音	poder	o
õ	o~	鼻半閉後圓唇母音	compra	o
ɔ	O	半開後圓唇母音	cotó	O
u	u	閉後圓唇母音	fui	u
ũ	u~	鼻閉後圓唇母音	sunto	u
其他符號				
'	"	主要重音	Alabama	
,	%	次要重音	Alabama	
.	.	音節邊界	A.la.ba.ma	

羅馬尼亞文 (ro-RO)

下表列出國際音標 (IPA) 音素、拓展音標字母評估法 (X-SAMPA) 符號，以及 Amazon Polly 所支援羅馬尼亞文聲音對應的視素。

音素/視素表

IPA	X-SAMPA	描述	範例	視素
子音				
b	b	濁雙唇塞音	bubă	p
d	d	濁齒齦塞音	după	t
ɖʒ	dʒ	濁顎齦塞擦音	george	S

IPA	X-SAMPA	描述	範例	視素
f	f	清唇齒擦音	afacere	f
g	g	濁軟顎塞音	agri#	k
h	h	清聲門擦音	harpă	k
j	j	硬顎近音	baie	i
k	k	清軟顎塞音	co#	k
l	l	齒齦邊近音	lampa	t
m	m	雙唇鼻音	mama	p
n	n	齒齦鼻音	nor	t
p	p	清雙唇塞音	pilă	p
r	r	齒齦顫音	rampă	r
s	s	清齒齦擦音	soare	s
ʃ	S	清齶後擦音	ma#ină	S
t	t	清齒齦塞音	tata	t
ʦ	ts	清齒齦塞擦音	#ară	s
tʃ	tS	清齶後塞擦音	ceai	S
v	v	濁唇齒擦音	via#ă	f
w	w	圓唇軟顎近音	beau	u
z	z	濁齒齦擦音	mozol	s
ʒ	Z	濁齶後擦音	joacă	S
母音				

IPA	X-SAMPA	描述	範例	視素
ə	@	中央母音	babă	@
a	a	開前不圓唇母音	casa	a
e	e	半閉前不圓唇母音	elan	e
ɐ	e_^	非音節半閉前不圓唇母音	beau	e
i	i	閉前不圓唇母音	mie	i
o	o	半閉後圓唇母音	oră	o
oa	o_^a	雙母音	oare	o
u	u	閉後圓唇母音	unde	u
ɨ	1	閉央不圓唇母音	România	i
其他符號				
'	"	主要重音	Alabama	
,	%	次要重音	Alabama	
.	.	音節邊界	A.la.ba.ma	

俄文 (ru-RU)

下表列出國際音標 (IPA) 音素、拓展音標字母評估法 (X-SAMPA) 符號，以及 Amazon Polly 所支援俄文聲音對應的視素。

音素/視素表

IPA	X-SAMPA	描述	範例	視素
子音				

IPA	X-SAMPA	描述	範例	視素
b	b	濁雙唇塞音	борт	p
b ^j	b'	顎音化濁雙唇塞音	бюро	p
d	d	濁齒齦塞音	дом	t
d ^j	d'	顎音化濁齒齦塞音	дядя	t
f	f	清唇齒擦音	флаг	f
f ^j	f'	顎音化清唇齒擦音	февраль	f
g	g	濁軟顎塞音	нога	k
g ^j	g'	顎音化濁軟顎塞音	герой	k
j	j	硬顎近音	дизайн、ящик	i
k	k	清軟顎塞音	кот	k
k ^j	k'	顎音化清軟顎塞音	кино	k
l	l	齒齦邊近音	лампа	t
l ^j	l'	顎音化齒齦邊近音	лес	t
m	m	雙唇鼻音	мама	p
m ^j	m'	顎音化雙唇鼻音	мяч	p
n	n	齒齦鼻音	нос	t
n ^j	n'	顎音化齒齦鼻音	няня	t
p	p	清雙唇塞音	папа	p
p ^j	p'	顎音化清雙唇塞音	перо	p
r	r	齒齦顫音	роза	r

IPA	X-SAMPA	描述	範例	視素
r ^j	r'	顎音化齒齶顫音	рюмка	r
s	s	清齒齶擦音	сыр	s
s ^j	s'	顎音化清齒齶擦音	сердце、русь	s
ʃ:	s\:	長清齶顎擦音	щека	J
ʂ	s`	清捲舌擦音	шум	S
t	t	清齒齶塞音	точка	t
t ^j	t'	顎音化清齒齶塞音	тётя	t
ʈs	ts	清齒齶塞擦音	царь	s
ʈʃ	ts\	清齶顎塞擦音	час	J
v	v	濁唇齒擦音	вор	f
v ^j	v'	顎音化濁唇齒擦音	верфь	f
x	x	清軟顎擦音	хор	k
x ^j	x'	顎音化清軟顎擦音	химия	k
z	z	濁齒齶擦音	зуб	s
z ^j	z'	顎音化濁齒齶擦音	зима	s
ʒ:	z\:	長濁齶顎擦音	уезжать	J
ʐ	z`	濁捲舌擦音	жена	S
母音				
ə	@	中央母音	канарейка	@
a	a	開前不圓唇母音	два、яблоко	a

IPA	X-SAMPA	描述	範例	視素
e	e	半閉前不圓唇母音	печь	e
ɛ	E	半開前不圓唇母音	это	E
i	i	閉前不圓唇母音	один、четыре	i
o	o	半閉後圓唇母音	кот	o
u	u	閉後圓唇母音	муж、вьюга	u
ɨ	1	閉央不圓唇母音	мышь	ɨ

西班牙文 (es-ES)

下表列出國際音標 (IPA) 音素、拓展音標字母評估法 (X-SAMPA) 符號，以及 Amazon Polly 所支援西班牙文聲音對應的視素。

音素/視素表

IPA	X-SAMPA	描述	範例	視素
子音				
r	4	齒齶閃音	pero、bravo、amor、	t
b	b	濁雙唇塞音	bestia	p
β	B	濁雙唇擦音	bebé	B
d	d	濁齒齶塞音	cuando	t
ð	D	濁齒擦音	arder	T
f	f	清唇齒擦音	fase、café	f
g	g	濁軟顎塞音	gato、lengua、guerra	k
ɣ	G	濁軟顎擦音	trigo、Argos	k

IPA	X-SAMPA	描述	範例	視素
j	j	硬顎近音	hacia、tierra、radio、	i
ɟ	ɟ\	濁硬顎擦音	enhielar、sayo、inye syerba	J
k	k	清軟顎塞音	caña、laca、quisimo	k
l	l	齒齦邊近音	lino、calor、princi pal	t
ʎ	ʎ	硬顎邊近音	llave、pollo	J
m	m	雙唇鼻音	madre、comer、anfil	p
n	n	齒齦鼻音	nido、anillo、sin	t
ɲ	ɲ	硬顎鼻音	cabaña、ñoquis	J
ŋ	ŋ	軟顎鼻音	cinco、venga	k
p	p	清雙唇塞音	pozo、topo	p
r	r	齒齦顫音	perro、enrachado	r
s	s	清齒齦擦音	saco、casa、puertas	s
t	t	清齒齦塞音	tamiz、átomo	t
ʈʂ	ʈʂ	清齶後塞擦音	chubasco	S
θ	θ	清齒擦音	cereza、zorro、lacer	T
w	w	圓唇軟顎近音	fuego、fuimos、cuot	u
x	x	清軟顎擦音	jamón、general、su je、reloj	k
z	z	濁齒齦擦音	rasgo、mismo	s
母音				

IPA	X-SAMPA	描述	範例	視素
a	a	開前不圓唇母音	tanque	a
e	e	半閉前不圓唇母音	peso	e
i	i	閉前不圓唇母音	cinco	i
o	o	半閉後圓唇母音	bosque	o
u	u	半閉前不圓唇母音	publicar	u
其他符號				
'	"	主要重音	Alabama	
,	%	次要重音	Alabama	
.	.	音節邊界	A.la.ba.ma	

西班牙文 (墨西哥) (es-MX)

下表列出國際音標 (IPA) 音素、拓展音標字母評估法 (X-SAMPA) 符號，以及 Amazon Polly 支援的墨西哥西班牙文聲音對應的視素。

音素/視素表

IPA	X-SAMPA	描述	範例	視素
子音				
r	4	齒齶閃音	pero、bravo、amor、	t
b	b	濁雙唇塞音	bestia	p
β	B	濁雙唇擦音	bebé	B
d	d	濁齒齶塞音	cuando	t
ð	D	濁齒擦音	arder	T

IPA	X-SAMPA	描述	範例	視素
f	f	清唇齒擦音	fase、café	f
g	g	濁軟顎塞音	gato、lengua、guerra	k
ɣ	G	濁軟顎擦音	trigo、Argos	k
j	j	硬顎近音	hacia、tierra、radio、	i
ɟ	j\	濁硬顎擦音	enhielar、sayo、inye syerba	J
k	k	清軟顎塞音	caña、laca、quisimo	k
l	l	邊齒齦近音	lino、calor、princi pal	t
m	m	雙唇鼻音	madre、comer、anfil	p
n	n	齒齦鼻音	nido、anillo、sin	t
ɲ	J	硬顎鼻音	cabaña、ñoquis	J
ŋ	N	軟顎鼻音	angosto、increíble	k
p	p	清雙唇塞音	pozo、topo	p
r	r	齒齦顫音	perro、enrachado	r
s	s	清齒齦擦音	saco、casa、puertas	s
ʃ	S	清齶後擦音	show、flash	S
t	t	清齒齦塞音	tamiz、átomo	t
ɬ	tS	清齶後塞擦音	chubasco	S
w	w	圓唇軟顎近音	fuego、fuimos、cuot	u
x	x	清軟顎擦音	jamón、general、 peaje、reloj	k

IPA	X-SAMPA	描述	範例	視素
z	z	濁齒齶擦音	rasgo、mismo	s
母音				
a	a	央開不圓唇母音	tanque	a
e	e	半閉前不圓唇母音	peso	e
i	i	閉前不圓唇母音	cinco	i
o	o	半閉後圓唇母音	bosque	o
u	u	閉後圓唇母音	publicar	u
其他符號				
'	"	主要重音	Alabama	
,	%	次要重音	Alabama	
.	.	音節邊界	A.la.ba.ma	

西班牙文 (美國) (es-US)

下表列出國際音標 (IPA) 音素、拓展音標字母評估法 (X-SAMPA) 符號，以及 Amazon Polly 所支援美國西班牙文聲音對應的視素。

音素/視素表

IPA	X-SAMPA	描述	範例	視素
子音				
r	4	齒齶閃音	pero、bravo、amor、	t
b	b	濁雙唇塞音	bestia	p
β	B	濁雙唇擦音	bebé	B

IPA	X-SAMPA	描述	範例	視素
d	d	濁齒齦塞音	cuando	t
ð	D	濁齒擦音	arder	T
f	f	清唇齒擦音	fase、café	f
g	g	濁軟顎塞音	gato、lengua、guerra	k
ɣ	G	濁軟顎擦音	trigo、Argos	k
j	j	硬顎近音	hacia、tierra、radio、	i
ɲ	j\	濁硬顎擦音	enhielar、sayo、inye syerba	J
k	k	清軟顎塞音	caña、laca、quisimo	k
l	l	邊齒齦近音	lino、calor、princi pal	t
m	m	雙唇鼻音	madre、comer、anfil	p
n	n	齒齦鼻音	nido、anillo、sin	t
ɲ	J	硬顎鼻音	cabaña、ñoquis	J
ŋ	N	軟顎鼻音	angosto、increíble	k
p	p	清雙唇塞音	pozo、topo	p
r	r	齒齦顫音	perro、enrachado	r
s	s	清齒齦擦音	saco、casa、puertas	s
ʃ	S	清齶後擦音	show、flash	S
t	t	清齒齦塞音	tamiz、átomo	t
ɬ	tS	清齶後塞擦音	chubasco	S

IPA	X-SAMPA	描述	範例	視素
w	w	圓唇軟顎近音	fuego、fuimos、cuot	u
x	x	清軟顎擦音	jamón, general, peaje, reloj	k
z	z	濁齒齶擦音	rasgo、mismo	s

母音

a	a	央開不圓唇母音	tanque	a
e	e	半閉前不圓唇母音	peso	e
i	i	閉前不圓唇母音	cinco	i
o	o	半閉後圓唇母音	bosque	o
u	u	閉後圓唇母音	publicar	u

其他符號

'	"	主要重音	Alabama	
,	%	次要重音	Alabama	
.	.	音節邊界	A.la.ba.ma	

瑞典文 (sv-SE)

下表列出國際音標 (IPA) 音素、拓展音標字母評估法 (X-SAMPA) 符號，以及 Amazon Polly 所支援瑞典文聲音的對應視素。

音素/視素表

IPA	X-SAMPA	描述	範例	視素
子音				

IPA	X-SAMPA	描述	範例	視素
b	b	濁雙唇塞音	bil	p
d	d	濁齒齦塞音	dal	t
ɖ	d`	濁捲舌塞音	bord	t
f	f	清唇齒擦音	fil	f
g	g	濁軟顎塞音	gås	k
h	h	清聲門擦音	hal	k
j	j	硬顎近音	jag	i
k	k	清軟顎塞音	kal	k
l	l	齒齦邊近音	lös	t
ɭ	l`	捲舌邊近音	härlig	t
m	m	雙唇鼻音	mil	p
n	n	齒齦鼻音	nålar	t
ɳ	n`	捲舌鼻音	barn	t
ŋ	N	軟顎鼻音	ring	k
p	p	清雙唇塞音	pil	p
r	r	齒齦顫音	ris	r
s	s	清齒齦擦音	sil	s
ɸ	s\	清硬顎擦音	tjock	J
ʂ	s`	清捲舌擦音	fors、schlager	S
t	t	清齒齦塞音	tal	t

IPA	X-SAMPA	描述	範例	視素
t	t`	清捲舌塞音	hjort	t
v	v	濁唇齒擦音	vår	f
w	w	圓唇軟顎近音	aula、airways	u
ɧ	x\	清硬顎軟顎擦音	sjuk	k
母音				
ø	2	半閉前圓唇母音	föll、förr	o
ø	2:	長半閉前圓唇母音	föl、nöt、för	o
ɵ	8	半閉央圓唇母音	buss、full	o
ə	@	中央母音	pojken	@
ʉ:	}:	長閉央圓唇母音	hus、ful	u
a	a	開前不圓唇母音	hall、matt	a
æ	{	次開前不圓唇母音	herr	a
ɑ:	A:	長開後不圓唇母音	hal、mat	a
e:	e:	長半閉前不圓唇母音	vet、hel	e
ɛ	E	半開前不圓唇母音	vett、rätt、hetta、hä	E
ɛ:	E:	長半開前不圓唇母音	säl、häl、här	E:
i:	i:	長閉前不圓唇母音	vit、sil	i:
ɪ	ɪ	次閉次前不圓唇母音	vitt、sill	i
o:	o:	長半閉後圓唇母音	håll、mål	o

IPA	X-SAMPA	描述	範例	視素
ɔ	O	半開後圓唇母音	háll、moll	O
u:	u:	長閉後圓唇母音	sol、bot	u
ʊ	U	次閉次後圓唇母音	bott	u
y	y	閉前圓唇母音	bytt	u
y:	y:	長閉前圓唇母音	syl、syl	u
其他符號				
'	"	主要重音	Alabama	
,	%	次要重音	Alabama	
.	.	音節邊界	A.la.ba.ma	

土耳其文 (tr-TR)

下表列出國際音標 (IPA) 音素、拓展音標字母評估法 (X-SAMPA) 符號，以及 Amazon Polly 所支援土耳其文聲音對應的視素。

音素/視素表

IPA	X-SAMPA	描述	範例	視素
子音				
r	4	齒齶閃音	durum	t
ɾ	4_0_r	清擦齒齶閃音	bir	t
ɽ	4_r	擦齒齶閃音	raf	t
b	b	濁雙唇塞音	raf	p
c	c	清硬顎塞音	kedı	k

IPA	X-SAMPA	描述	範例	視素
d	d	濁齒齦塞音	dede	t
ɖʒ	dʒ	濁顎齦塞擦音	cam	S
f	f	清唇齒擦音	fare	f
g	g	濁軟顎塞音	galibi	k
h	h	清聲門擦音	hasta	k
j	j	硬顎近音	yat	i
ʝ	J\	濁硬顎塞音	genç	J
k	k	清軟顎塞音	akıl	k
l	l	齒齦邊近音	lale	t
ɭ	5	軟顎化齒齦邊近音	labirent	t
m	m	雙唇鼻音	maaş	p
n	n	齒齦鼻音	anı	t
p	p	清雙唇塞音	ip	p
s	s	清齒齦擦音	ses	s
ʃ	S	清齶後擦音	aşı	S
t	t	清齒齦塞音	ütü	t
ɬʃ	tS	清齶後塞擦音	çaba	S
v	v	濁唇齒擦音	ekvator、kahveci、al	f
z	z	濁齒齦擦音	ver	s
ʒ	Z	濁齶後擦音	azık	S

IPA	X-SAMPA	描述	範例	視素
母音				
ø	2	半閉前圓唇母音	göl	0
œ	9	半開前圓唇母音	banliyö	O
a	a	開前不圓唇母音	kal	a
a:	a:	長開前不圓唇母音	davacı	a
æ	{	次開前不圓唇母音	özlem、güvenlik、gü	a
e	e	半閉前不圓唇母音	keçi	e
ɛ	E	半開前不圓唇母音	dede	E
i	i	閉前不圓唇母音	bir	i
i:	i:	長閉前不圓唇母音	izah	i
ɪ	ɪ	次閉次前不圓唇母音	keçi	i
ʊ	ʊ	閉後不圓唇母音	kıl	i
o	o	半閉後圓唇母音	kol	o
o:	o:	長半閉後圓唇母音	dolar	o
u	u	閉後圓唇母音	durum	u
u:	u:	長閉後圓唇母音	ruhum	u
ʊ	U	次閉次後圓唇母音	dolu	u
y	y	閉前圓唇母音	güvenlik	u
ɣ	Y	次閉次前圓唇母音	aşı	u
其他符號				

IPA	X-SAMPA	描述	範例	視素
'	"	主要重音	Alabama	
,	%	次要重音	Alabama	
.	.	音節邊界	A.la.ba.ma	

威爾斯文 (cy-GB)

下表列出國際音標 (IPA) 音素、拓展音標字母評估法 (X-SAMPA) 符號，以及 Amazon Polly 所支援威爾斯語的對應視素。

音素/視素表

IPA	X-SAMPA	描述	範例	視素
子音				
b	b	濁雙唇塞音	baban	p
d	d	濁齒齦塞音	deg	t
ɖʒ	dʒ	濁顎齦塞擦音	garej	s
ð	D	濁齒擦音	deuddeg	T
f	f	清唇齒擦音	ffacs	f
g	g	濁軟顎塞音	gadael	k
h	h	清聲門擦音	haearn	k
j	j	硬顎近音	astudio	i
k	k	清軟顎塞音	cant	k
l	l	齒齦邊近音	lan	t
ɬ	K	清齒齦邊擦音	llan	t

IPA	X-SAMPA	描述	範例	視素
m	m	雙唇鼻音	mae	p
ᵹ	m_0	清雙唇鼻音	ymhen	p
n	n	齒齦鼻音	naw	t
ɳ	n_0	清齒齦鼻音	anhawster	t
ŋ	N	軟顎鼻音	argyfwng	k
ɲ	N_0	清軟顎鼻音	anghenion	k
p	p	清雙唇塞音	pump	p
r	r	齒齦顫音	rhoi	r
ɾ	r_0	清齒齦顫音	garw	r
s	s	清齒齦擦音	saith	s
ʃ	S	清齶後擦音	siawns	S
t	t	清齒齦塞音	tegan	t
ɥ	tS	清齶後塞擦音	cytsain	S
θ	T	清齒擦音	aberth	T
v	v	濁唇齒擦音	prawf	f
w	w	圓唇軟顎近音	rhagweld	u
χ	X	清小舌擦音	chwech	k
z	z	濁齒齦擦音	aids	s
ʒ	Z	濁齶後擦音	rouge	S
母音				

IPA	X-SAMPA	描述	範例	視素
ə	@	中央母音	ychwanega	@
a	a	開前不圓唇母音	acen	a
ai	ai	雙母音	dau	a
au	au	雙母音	awdur	a
ɑ:	A:	長開後不圓唇母音	mab	a
ɑ:i	A:1	雙母音	aelod	a
e:	e:	長半閉前不圓唇母音	peth	e
ɛ	E	半開前不圓唇母音	pedwar	E
ɛi	Ei	雙母音	beic	E
i:	i:	長閉前不圓唇母音	tri	i
ɪ	ɪ	次閉次前不圓唇母音	miliwn	i
ɪu	ɪu	雙母音	unigryw	i
o:	o:	長半閉後圓唇母音	oddi	o
ɔ	O	半開後圓唇母音	oddieithr	O
ɔi	Oi	雙母音	troi	O
ɔu	Ou	雙母音	rownd	O
u:	u:	長閉後圓唇母音	cwch	u
ʊ	U	次閉次後圓唇母音	acwstig	u
ʊi	Ui	雙母音	wyth	u

IPA	X-SAMPA	描述	範例	視素
其他符號				
'	"	主要重音	Alabama	
,	%	次要重音	Alabama	
.	.	音節邊界	A.la.ba.ma	

Amazon Polly 語音引擎

Amazon Polly 有四個語音引擎，可將輸入文字轉換為逼真的語音。這些包括：生成式、長式、神經式和標準式。若要使用 Amazon Polly 語音，請選取引擎和語音合成 API 操作。然後提供輸入文字讓引擎合成，然後選取音訊輸出格式。根據這些輸入，Amazon Polly 會將提供的文字合成為高品質的語音音訊串流。

下列各節包含 Amazon Polly 提供之語音引擎的詳細資訊。

主題

- [生成式語音](#)
- [長形語音](#)
- [神經語音](#)
- [標準語音](#)
- [選擇語音引擎](#)

生成式語音

Amazon Polly 的生成文字text-to-speech(TTS) 引擎提供最人性化、情感參與和適應性的對話語音，可透過 Amazon Polly 主控台使用。

生成式引擎是迄今為止最大的 Amazon Polly TTS 模型。它會部署十億參數轉換器，將原始文字轉換為語音程式碼，然後部署以卷積為基礎的解碼器，以增量、可串流的方式將這些語音程式碼轉換為波形。此方法顯示大型語言模型 (LLMs) 在訓練越來越多的公開可用和專屬資料時廣泛報告的緊急功能，包括各種語音、語言和風格。

生成式引擎會建立合成語音，以與人類語音非常相似的方式進行情感參與、自信和高度共通。您可以使用這些語音做為知識淵博的客戶助理、虛擬訓練師或具有近乎人類合成語音的廣告商。

Note

以這些語音為基礎的state-of-the-art技術位於生成式 AI 語言和語音建模的範式中。技術的副作用是，訓練資料和模型的任何更新都可能導致聲音聲音的音量稍微變化，即使整體品質隨著模型更新而改善。這可能會影響長時間合成不同內容組件的使用案例，例如播客季節。

可用的生成語音

Amazon Polly 目前在生成變體中提供 43 個語音。

	Language	語言代碼	名稱/ID	Gender
1	英文 (澳洲)	en-AU	奧利維亞	女性
2	英文 (英國)	en-GB	Amy	女性
			Brian	男性
3	英文 (印度)	en-IN	卡加爾文	女性
4	英文 (愛爾蘭)	en-IE	Niamh	女性
5	英文 (紐西蘭)	en-NZ	Aria	女性
6	英文 (新加坡)	en-SG	茉莉花	女性
7	英文 (南非)	en-ZA	Ayanda	女性
8	英文 (美國)	zh-TW	丹尼爾文	女性
			Joanna	女性
			Matthew	男性
			Ruth	女性
			Salli	女性
			Stephen	男性
			愛寧達	女性
9	荷蘭文 (比利時)	nl-BE	Lisa	女性
10	荷蘭文 (荷蘭)	nl-NL	月亮	女性
11	法文 (比利時)	fr-BE	伊薩布爾	女性

	Language	語言代碼	名稱/ID	Gender
12	法文 (加拿大)	fr-CA	加布里爾文	女性
			Liam	男性
13	法文 (法國)	fr-FR	Ambre	女性
			Céline	女性
			佛羅里安	男性
			Léa	女性
			Rémi	男性
14	德文 (奧地利)	de-AT	漢納	女性
15	德文 (德國)	de-DE	丹尼爾	男性
			利納特	男性
			Vicki	女性
16	德文 (瑞士)	de-CH	薩布林納	女性
17	義大利文 (義大利)	it-IT	披頭四	女性
			Bianca	女性
			Lorenzo	男性
18	韓文 (韓國)	ko-KR	Seoyeon	女性
19	波蘭文 (波蘭)	pl-PL	Ewa	女性
			奧拉	女性
20	葡萄牙文 (巴西)	pt-BR	Camila	女性

	Language	語言代碼	名稱/ID	Gender
21	西班牙文 (墨西哥)	es-MX	安德列	男性
			Mía	女性
22	西班牙文 (西班牙)	es-ES	Lucia	女性
			Sergio	男性
23	西班牙文 (美國)	es-US	Lupe	女性
			Pedro	男性

Note

生成語音成本會在 [Amazon Polly 定價資訊頁面上](#) 指定。

功能和區域相容性

Amazon Polly 生成語音可在下列區域使用：

- 美國東部 (維吉尼亞北部)：us-east-1
- 歐洲 (法蘭克福)：eu-central-1
- 美國西部 (奧勒岡)：us-west-2
- 亞太區域 (東京)：ap-northeast-1
- 亞太區域 (首爾)：ap-northeast-2
- 亞太區域 (新加坡)：ap-southeast-1
- 歐洲 (倫敦)：eu-west-2
- 加拿大 (中部)：ca-central-1
- 歐洲 (蘇黎世)：eu-central-2
- 其他區域無法使用

生成語音支援下列功能：

- 雙向串流 API 現已在生成式引擎中提供，並允許同時串流輸入和輸出。此 API 適用於下列 AWS 區域：美國東部（維吉尼亞北部）、歐洲（法蘭克福）、美國西部（奧勒岡）、亞太區域（新加坡）、歐洲（倫敦）、加拿大（中部）和歐洲（蘇黎世）。請造訪 [文件](#) 以進一步了解如何使用。
- 即時和非同步語音合成操作。
- 生成式引擎不支援新聞播報員說話風格。
- Amazon Polly 支援許多（但不是全部）SSML 標籤。如需 NTTTS 支援的 SSML 標籤的詳細資訊，請參閱 [支援的 SSML 標籤](#)
- 如同標準語音，您可以從各種取樣率中選擇，為您的應用程式最佳化頻寬和音訊品質。標準和神經語音的有效取樣率為 8 kHz、16 kHz、22 kHz 或 24 kHz。標準語音的預設值為 22 kHz。生成語音的預設值為 24 kHz。Amazon Polly 支援 MP3、OGG (Vorbis) 和原始 PCM 音訊串流格式。

目前無法支援產生語音標記。

Note

目前，歐洲（倫敦）、加拿大（中部）和歐洲（蘇黎世）區域僅支援下列生成語音：Joanna (en-US)、Ruth (en-US)、Salli (en-US)、Stephen (en-US)、INTI (en-US)、Amy (en-GB)、Brian (en-GB)、Olivia (en-AU)、Florian (fr-FR)、Ambre (fr-FR)、Lorenzo (it-IT)、Beatrice (it-IT)、Jasmine (en-SG)、Aria (en-NZ)、Lennart (de-DE)、Vicki (de-DE)、Sabbrina (de-CH)、Hannah (

Note

在極少數的模型幻覺情況下，（以及隨著生成式引擎透過字符轉譯語音字符的模型行為）會實施強制的緊急停止機制。內建機制會阻止模型進一步轉譯語音。此安全功能是以資料分析為基礎，其中模型可能幻覺，通常是在句子結尾。

在某些情況下，模型可能會認為它會嫉妒，然後最終可能會在產生步驟中剪下單字，從而渲染一半的單字。這可能會產生不適當的結果。

長形語音

Amazon Polly 具有長形引擎，可產生類似人類、高度表達和情緒熟練的語音。長形語音旨在吸引聆聽者注意較長的內容，例如新聞文章、訓練資料或行銷影片。

Amazon Polly 長形語音是使用尖端深度學習 TTS 技術所開發。此模型會學習複寫人類語言的音素、音素、語調和其他音素和聲音層面，進而產生高度自然的語音輸出。

長形引擎使用文字內嵌來解釋文字的意義。使用文字內嵌，長形引擎可以產生自然聲音的正確強調、暫停和色調。結果是語音，結合存在於人類溝通中的完整情緒元素範圍。這包括模擬意外或區分對話與敘述。這共同建立了一種高級語音產品，聽起來像活人。

Note

以這些語音為基礎的state-of-the-art技術位於生成式 AI 語言和語音建模的範式中。技術的副作用是，任何訓練資料和模型的更新都可能導致聲音聲音聲音的些微變化，即使整體品質隨著模型更新而改善。這可能會影響長時間合成不同內容組件的使用案例，例如播客季節。

可用的長格式語音

Amazon Polly 目前提供四個 en-US 和兩個 es-ES 長格式語音。兩種語言都提供女性和男性語音。英文長格式語音 Daniel、Gregory 和 Ruth 也提供對話式 NTTS 變體。

	Language	語言代碼	名稱/ID	Gender
1	英文 (美國)	zh-TW	丹尼爾文	女性
			Gregory	男性
			Ruth	女性
			派翠克	男性
2	西班牙文 (西班牙)	es-ES	Alba	女性
			Raúl	男性

功能和區域相容性

Amazon Polly 長格式語音可在下列區域使用：

- 美國東部 (維吉尼亞北部)：us-east-1
- 其他區域無法使用

Amazon Polly Long-form 引擎支援下列功能：

- 即時和非同步語音合成操作。
- 所有[語音標記](#)。
- Amazon Polly 支援許多（但不是全部）SSML 標籤。如需 NTTS 支援的 SSML 標籤的詳細資訊，請參閱[支援的 SSML 標籤](#)
- 如同標準語音，您可以從各種取樣率中選擇，為您的應用程式最佳化頻寬和音訊品質。標準語音、長式語音和神經語音的有效取樣率為：8 kHz、16 kHz、22kHz 或 24 kHz。標準語音的預設值為 22 kHz。長格式和神經語音的預設值為 24 kHz。Amazon Polly 支援 MP3、OGG (Vorbis) 和原始 PCM 音訊串流格式。

Note

長格式語音成本是在 [Amazon Polly 定價資訊頁面上](#) 指定。

神經語音

Amazon Polly 具有神經text-to-speech(NTTS) 引擎，可產生比標準語音更高品質的語音。標準 TTS 語音使用串連合成。標準引擎會串連錄製語音的音素，產生非常自然的音調合成語音。然而，語音中不可避免的變化以及用於分段波形的技術會限制語音品質。Amazon Polly NTTS 引擎不使用標準串連合成來產生語音。它有兩個部分：

- 神經網路 — 將一系列音素（最基本的語言單位）轉換為一系列光譜。(Spectrogram 是不同頻帶中能源層級的快照。)
- vocoder — 將光譜圖轉換為幾乎連續的音訊訊號。

類神經 TTS 系統的第一個元件是序列對序列的模型。此模型不會只從對應的輸入建立其結果，也會考慮輸入元素序列如何共同運作。模型會選擇其輸出的光譜圖，以便其頻帶強調人類大腦在處理語音時使用的聲學特徵。

接著，此模型的輸出會傳遞至神經聲碼器。這會將光譜圖轉換為語音波形。在用於建置一般用途串連合成系統的大型資料集上訓練時，這種sequence-to-sequence方法會產生更高品質、更自然的語音。

可用的神經語音

神經語音提供 36 種語言和語言變體。下表列出了這些語音。

	語言和語言變體	語言代碼	名稱/ID	Gender
1	阿拉伯文 (海灣)	ar-AE	Hala	女性
			Zayd	男性
2	比利時荷蘭文 (瑞典文)	nl-BE	Lisa	女性
3	加泰隆尼亞文	ca-ES	Arlet	女性
4	捷克文	cs-CZ	吉特卡	女性
5	中文 (中文)	yue-CN	海津	女性
6	中文 (國語)	cmn-CN	Zhiyu	女性
7	丹麥文	da-DK	Sofie	女性
8	荷蘭文	nl-NL	月亮	女性
9	英文 (澳洲)	en-AU	奧利維亞	女性
10	英文 (英國)	en-GB	Amy*	女性
			Emma	女性
			Brian	男性
			Arthur	男性
11	英文 (印度)	en-IN	卡加爾文	女性
12	英文 (愛爾蘭)	en-IE	Niamh	女性
13	英文 (紐西蘭)	en-NZ	Aria	女性
14	英文 (新加坡)	en-SG	茉莉花	女性
15	英文 (南非)	en-ZA	Ayanda	女性
16	英文 (美國)	zh-TW	丹尼爾文	女性

	語言和語言變體	語言代碼	名稱/ID	Gender
			Gregory	男性
			Ivy	女性 (子)
			Joanna*	女性
			Kendra	女性
			Kimberly	女性
			Salli	女性
			Joey	男性
			Justin	男性 (兒童)
			Kevin	男性 (兒童)
			Matthew*	男性
			Ruth	女性
			Stephen	男性
17	芬蘭文	fi-FI	Suvi	女性
18	法文 (比利時)	fr-BE	伊薩布爾	女性
19	法文 (加拿大)	fr-CA	加布里爾文	女性
			Liam	男性
20	法文	fr-FR	Léa	女性
			Rémi	男性
21	德文	de-DE	Vicki	女性
			丹尼爾	男性
22	德文 (奧地利)	de-AT	漢納	女性

	語言和語言變體	語言代碼	名稱/ID	Gender
23	德文 (瑞士)	de-CH	薩布林納	女性
24	印地文	hi-IN	卡加爾文	女性
25	義大利文	it-IT	Bianca	女性
			Adriano	男性
26	日文	ja-JP	Takumi	男性
			卡茲哈	女性
			Tomoko	女性
27	韓文	ko-KR	Seoyeon	女性
			Jihye	女性
28	挪威文	nb-NO	Ida	女性
29	波蘭文	pl-PL	奧拉	女性
30	葡萄牙文 (巴西)	pt-BR	Camila	女性
			Vitória/Vitoria	女性
			Thiago	男性
31	葡萄牙文 (歐洲)	pt-PT	Inês/Ines	女性
32	西班牙文 (西班牙)	es-ES	Lucia	女性
			Sergio	男性
33	西班牙文 (墨西哥)	es-MX	Mia	女性
			安德列	男性

	語言和語言變體	語言代碼	名稱/ID	Gender
34	西班牙文（美國）	es-US	Lupe*	女性
			Pedro	男性
35	瑞典文	sv-SE	Elin	女性
36	土耳其文	tr-TR	爆量	女性

*Amy、Joanna、Lupe 和 Matthew 語音可與新聞播報員說話風格搭配使用。如需詳細資訊，請參閱[套用新聞播報員語音](#)。

功能和區域相容性

並非所有 AWS 區域都提供神經語音，也不支援所有 Amazon Polly 功能。

下列區域支援神經語音：

- 美國東部 (維吉尼亞北部)：us-east-1
- 美國西部 (奧勒岡)：us-west-2
- 非洲（開普敦）：af-south-1
- 亞太區域（東京）：ap-northeast-1
- 亞太區域（首爾）：ap-northeast-2
- 亞太區域（大阪）：ap-northeast-3
- 亞太區域（孟買）：ap-south-1
- 亞太區域（新加坡）：ap-southeast-1
- 亞太區域 (雪梨)：ap-southeast-2
- 亞太區域（馬來西亞）：ap-southeast-5
- 亞太區域（泰國）：ap-southeast-7
- 加拿大（中部）：ca-central-1
- 歐洲（法蘭克福）：eu-central-1
- 歐洲（愛爾蘭）：eu-west-1
- 歐洲（倫敦）：eu-west-2
- 歐洲（巴黎）：eu-west-3

- 歐洲（西班牙）：eu-south-2
- 歐洲（蘇黎世）：eu-central-2
- AWS GovCloud（美國西部）：us-gov-west-1

這些區域的端點和協定與用於標準語音的端點和協定相同。如需詳細資訊，請參閱 [Amazon Polly 端點和配額](#)。

神經語音支援以下功能：

- 即時和非同步語音合成操作。
- 新聞播報員風格。如需有關說話風格的詳細資訊，請參閱 [套用新聞播報員語音](#)。
- 所有語音標記。
- Amazon Polly 支援的許多（但不是全部）SSML 標籤。如需 NTTS 支援的 SSML 標籤的詳細資訊，請參閱支援的標籤。

如同標準語音，您可以從各種取樣率中選擇，為您的應用程式最佳化頻寬和音訊品質。標準和神經語音的有效取樣率為 8 kHz、16 kHz、22 kHz 或 24 kHz。標準語音的預設值為 22 kHz。神經語音的預設值為 24 kHz。Amazon Polly 支援 MP3、OGG (Vorbis) 和原始 PCM 音訊串流格式。

標準語音

Amazon Polly 具有使用串連合成的標準引擎。標準引擎會串連錄製語音的音素，產生非常自然的音調合成語音。

可用的標準語音

Amazon Polly 目前提供 29 種語言和語言變體的 40 個女性和 20 個男性標準語音。

	Language	語言代碼	名稱/ID	Gender
1	阿拉伯文	arb	Zeina	女性
2	中文（國語）	cmn-CN	Zhiyu	女性
3	丹麥文	da-DK	Naja	女性
			Mads	男性

	Language	語言代碼	名稱/ID	Gender
4	荷蘭文	nl-NL	Lotte	女性
			Ruben	男性
5	英文 (澳洲)	en-AU	Nicole	女性
			Russell	男性
6	英文 (英國)	en-GB	Amy	女性
			Emma	女性
			Brian	男性
7	英文 (印度)	en-IN	Aditi	女性
			Raveena	女性
8	英文 (美國)	zh-TW	Ivy	女性
			Joanna	女性
			Kendra	女性
			Kimberly	女性
			Salli	女性
			Joey	男性
			Kevin	男性
9	英文 (威爾)	en-GB-WLS	Geraint	男性
10	法文	fr-FR	Céline/Celine	女性
			Léa	女性
			Mathieu	男性
11	法文 (加拿大)	fr-CA	Chantal	女性

	Language	語言代碼	名稱/ID	Gender
12	德文	de-DE	Marlene	女性
			Vicki	女性
			Hans	男性
13	印地文	hi-IN	Aditi	女性
14	冰島文	is-IS	Dóra/Dora	女性
			Karl	男性
15	義大利文	it-IT	Carla	女性
			Bianca	女性
			Giorgio	男性
16	日文	ja-JP	Mizuki	女性
			Takumi	男性
17	韓文	ko-KR	Seoyeon	女性
18	挪威文	nb-NO	Liv	女性
19	波蘭文	pl-PL	Ewa	女性
			Maja	女性
			Jacek	男性
			Jan	男性
20	葡萄牙文 (巴西)	pt-BR	Camila	女性
			Vitória/Vitoria	女性
			Ricardo	男性

	Language	語言代碼	名稱/ID	Gender
21	葡萄牙文 (歐洲)	pt-PT	Inês/Ines	女性
			Cristiano	男性
22	羅馬尼亞文	ro-RO	Carmen	女性
23	俄文	ru-RU	Tatyana	女性
			Maxim	男性
24	西班牙文 (西班牙)	es-ES	Conchita	女性
			Lucia	女性
			Enrique	男性
25	西班牙文 (墨西哥)	es-MX	Mia	女性
26	西班牙文 (美國)	es-US	Lupe	女性
			Penélope/ Penelope	女性 男性
			Miguel	
27	瑞典文	sv-SE	Astrid	女性
28	土耳其文	tr-TR	Filiz	男性
29	威爾斯文	cy-GB	Gwyneth	女性

功能和區域相容性

Amazon Polly 標準語音可在下列 Amazon Polly 區域使用：

- 美國東部 (維吉尼亞北部)：us-east-1
- 美國東部 (俄亥俄)：us-east-2

- 美國西部 (加利佛尼亞北部) : us-west-1
- 美國西部 (奧勒岡) : us-west-2
- 非洲 (開普敦) : af-south-1
- 亞太區域 (香港) : ap-east-1
- 亞太區域 (東京) : ap-northeast-1
- 亞太區域 (首爾) : ap-northeast-2
- 亞太區域 (大阪) : ap-northeast-3
- 亞太區域 (孟買) : ap-south-1
- 亞太區域 (新加坡) : ap-southeast-1
- 亞太區域 (雪梨) : ap-southeast-2
- 亞太區域 (馬來西亞) : ap-southeast-5
- 中國 (寧夏) : cn-northwest-1 ;
- 加拿大 (中部) : ca-central-1
- 歐洲 (法蘭克福) : eu-central-1
- 歐洲 (愛爾蘭) : eu-west-1
- 歐洲 (倫敦) : eu-west-2
- 歐洲 (巴黎) : eu-west-3
- 歐洲 (西班牙) : eu-south-2
- 歐洲 (斯德哥爾摩) : eu-north-1
- 中東 (巴林) : me-south-1
- 南美洲 (聖保羅) : sa-east-1
- AWS GovCloud (美國西部) : us-gov-west-1

這些區域的端點和通訊協定與用於神經語音的端點和通訊協定相同。如需詳細資訊，請參閱 [Amazon Polly 端點和配額](#)。

Amazon Polly 標準引擎支援下列功能 (待定) :

- 即時和非同步語音合成操作。
- 所有 [語音標記](#)。
- Amazon Polly 支援許多 (但非全部) SSML 標籤。如需 NTTS 支援的 SSML 標籤的詳細資訊，請參閱 [支援的 SSML 標籤](#)。

- 您可以選擇各種取樣率，以最佳化應用程式的頻寬和音訊品質。標準語音的預設取樣率為 22 kHz。Amazon Polly 支援 MP3、OGG (Vorbis) 和原始 PCM 音訊串流格式。

Note

標準語音成本在 [Amazon Polly 定價資訊頁面上](#) 指定。

選擇語音引擎

您可以透過 Amazon Polly 主控台或 存取 Amazon Polly 語音 AWS CLI。

在主控台上選擇語音引擎

1. 開啟位於 <https://console.aws.amazon.com/polly/> 的 Amazon Polly 主控台。
2. 從 Amazon Polly 主控台中，選擇所需的語音引擎。
3. 從語音下拉式功能表中選擇所需的語音。
4. 使用您選擇的文字產生 TTS 音訊。

若要在 中選擇語音引擎 AWS CLI，請在 SynthesizeSpeech 或 StartSpeechSynthesisTask API 操作 VoiceId 中指定 Engine 和 。如需一些範例，請參閱 [快速入門程式碼範例](#) 和 [Python 範例](#)。

語音標記

語音標記是描述您所合成的語音的中繼資料，例如句子或文字在音訊串流中開始和結束的地方。當您請求文字的語音標記時，Amazon Polly 會傳回此中繼資料，而不是合成的語音。透過使用語音標記與合成語音音訊串流，您可以為您的應用程式提供增強的視覺化體驗。

例如，將中繼資料與文字中的音訊串流結合，可讓您將語音與臉部動畫（唇同步）同步，或在說話時反白文字。

使用神經、長形或標準text-to-speech引擎時，可以使用語音標記。

主題

- [語音標記類型](#)
- [Visemes 和 Amazon Polly](#)
- [語音標記輸出](#)
- [請求語音標記](#)
- [不含 SSML 的語音標記範例](#)
- [使用 SSML 的語音標記範例](#)

語音標記類型

您可以使用 [SynthesizeSpeech](#) 或 `StartSpeechSynthesisTask` 命令的 [SpeechMarkTypes](#) 選項來請求語音標記。[StartSpeechSynthesisTask](#) 您指定要從您的輸入文字傳回的中繼資料元素。您可以請求最多四種類型的中繼資料，但是每個請求必須至少指定一個。請求不會產生任何音訊輸出。

在 中 AWS CLI，例如：

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

Amazon Polly 使用以下元素產生語音標記：

- 句子 – 表示輸入文字中的句子元素。
- 字詞 – 表示文字中的字詞元素。
- 視素 – 描述對應於每個說話音素的臉部和嘴巴動作。如需詳細資訊，請參閱[Visemes 和 Amazon Polly](#)。
- ssml – 從 SSML 輸入文字描述 `<mark>` 元素。如需詳細資訊，請參閱[從 SSML 文件產生語音](#)。

Visemes 和 Amazon Polly

視素代表說出文字時臉部和嘴部的位置。它同等視覺上的音素，是形成單字的基本聲音單位。視素是語音的基本視覺化建置區塊。

每種語言都有一組與其特定音素對應的視覺效果。在語言中，每個音素都有對應的視素，其代表形成聲音時的嘴形。不過，並非所有視素都可對應到特定得音素，因為許多音素在說出時看起來相同，即使聽起來不同。例如，英文中的「pet」和「bet」這兩個詞在聲音上不同。但是，在視覺上觀察 (沒有聲音) 時，它們看起來完全相同。

下表列出國際音標 (IPA) 音素和拓展音標字母評估法 (X-SAMPA) 符號的部分清單，以及美式英文聲音對應的視素

如需所有可用語言的完整表格，請參閱[Amazon Polly 中的語言](#)。

IPA	X-SAMPA	Description	範例	視素
子音				
b	b	濁雙唇塞音	bed	p
d	d	濁齒齦塞音	dig	t
ɟʒ	dʒ	濁顎齦塞擦音	jump	S
ð	D	濁齒擦音	then	T
f	f	清唇齒擦音	five	f
g	g	濁軟顎塞音	game	k
h	h	清聲門擦音	house	k
...	...	...	...	...

語音標記輸出

Amazon Polly 會在以行分隔的 JSON 串流中傳回語音標記物件。語音標記物件包含下列欄位：

- 時間 – 從對應的音訊串流開始的時間戳記 (以毫秒為單位)

- type – 語音標記的類型（句子、字詞、視素或 ssmi）
- start – 輸入文字中物件開頭的位元組（而非字元）偏移量（不包括視覺標記）
- end – 輸入文字中物件結尾的位元組（而非字元）偏移量（不包括視覺標記）
- 值 – 這會根據語音標記的類型而異
 - SSML：<mark> SSML 標籤
 - 視素：視素名稱
 - 文字或句子：輸入文字的子字串，由開始和結束欄位分隔

例如，Amazon Polly 會從文字「Mary has a little lamb」產生下列word語音標記物件：

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

所述文字（「had」）在音訊串流開始後 373 毫秒起始，並在輸入文字的位元組 5 開始位元組 8 結束。

Note

此中繼資料是用於 Joannavoice-id。如果您使用另一個具相同輸入文字的語音，中繼資料可能不同。

請求語音標記

您可以使用 主控台或 synthesize-speech 命令向 Amazon Polly 請求語音標記。然後，您可以檢視中繼資料或將其儲存至檔案。

Console

在主控台上產生語音標記

1. 登入 AWS 管理主控台，並在 <https://console.aws.amazon.com/polly/>：// 開啟 Amazon Polly 主控台。
2. 選擇 Text-to-Speech (文字轉換語音) 索引標籤。
3. 開啟 SSML 以使用 SSML。
4. 輸入文字或將其貼到輸入方塊。
5. 針對語言，選擇文字的語言。

6. 針對語音，選擇您要使用的語音。
7. 若要變更文字發音、展開其他設定、開啟自訂發音，以及針對套用語彙，選擇所需的語彙。
8. 若要驗證語音，請選擇接聽。
9. 開啟語音檔案格式設定。

 Note

下載 MP3、OGG、PCM、Mu-law 或 A-law 格式不會產生語音標記。

10. 針對檔案格式，選擇語音標記。
11. 對於語音標記類型，選擇要產生的語音標記類型。選擇 SSML 中繼資料的選項僅在 SSML 開啟時可用。如需搭配 Amazon Polly 使用 SSML 的詳細資訊，請參閱 [從 SSML 文件產生語音](#)。
12. 選擇 Download (下載)。

AWS CLI

除了輸入文字之外，需要下列元素才能傳回此中繼資料：

- `output-format`

傳回語音標記時，Amazon Polly 僅支援 JSON 格式。

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

如果您使用不支援的輸出格式，Amazon Polly 會擲回例外狀況。

- `voice-id`

為確保中繼資料符合關聯的音訊串流，請指定相同的語音用來產生合成的語音音訊串流。可用的語音沒有相同的語音速率。如果您使用用於產生語音以外的聲音，中繼資料不會與音訊串流相符。

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

- `speech-mark-types`

指定您想要的語音標記的類型。您可以請求任何或所有語音標記類型，但是必須至少指定一個類型。

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

- **text-type**

純文字是 Amazon Polly 的預設輸入文字，因此 `text-type ssml` 如果您想要傳回 SSML 語音標記，則必須使用。

- **outfile**

指定要寫入中繼資料的輸出檔案。

```
MaryLamb.txt
```

下列 AWS CLI 範例已針對 Unix、Linux 和 macOS 格式化。對於 Windows，將每一行尾端的反斜線 (\) Unix 接續字元取代為插入符號 (^)，並在輸入文字周圍使用完整引號 (") 取代內部標籤的單引號 (')。

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

不含 SSML 的語音標記範例

以下範例顯示您請求的簡單句子：「Mary had a little lamb.」的中繼資料在畫面上的呈現。為簡單起見，我們在這個範例中不包含 SSML 語音標記。

下列 AWS CLI 範例已針對 Unix、Linux 和 macOS 格式化。對於 Windows，將每一行尾端的反斜線 (\) Unix 接續字元取代為插入符號 (^)，並在輸入文字周圍使用完整引號 (") 取代內部標籤的單引號 (')。

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

當您提出此請求時，Amazon Polly 會在 .txt 檔案中傳回下列項目：

```
{
  "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"}
```

在這個輸出中，文字的每個部分是以語音標記分開：

- 句子「Mary had a little lamb.」
- 句子中的每個字：「Mary」、「had」、「a」、「little」和「lamb.」
- 對應的音訊串流中每個聲音的視素：「p」、「E」、「r」、「i」以此類推。如需有關視素的詳細資訊，請參閱 [Visemes](#) 和 [Amazon Polly](#)。

使用 SSML 的語音標記範例

從 SSML 增強文字產生語音標記的程序與沒有 SSML 的程序類似。使用 `synthesize-speech` 命令，並指定 SSML 增強的文字和語音標記的類型，如下例所示。為了讓範例更容易閱讀，我們不包含視覺語音標記，但這些也可以包含。

下列 AWS CLI 範例已針對 Unix、Linux 和 macOS 格式化。對於 Windows，將每一行尾端的反斜線 (\) Unix 接續字元取代為插入符號 (^)，並在輸入文字周圍使用完整引號 (") 取代內部標籤的單引號 (')。

```
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
```

當您提出此請求時，Amazon Polly 會在 .txt 檔案中傳回下列項目：

```
{"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"}
```

從 SSML 文件產生語音

您可以使用 Amazon Polly 從純文字或使用語音合成標記語言 (SSML) 標記的文件產生語音。使用 SSML 增強的文字可讓您進一步控制 Amazon Polly 如何從您提供的文字產生語音。

使用 SSML 標籤，您可以自訂和控制語音的各方面，例如發音、音量和語音速率。在 [中 AWS 管理主控台](#)，您要轉換為音訊的 SSML 增強文字會在 Text-to-Speech 頁面的 SSML 索引標籤上輸入。雖然以純文字輸入的文字依賴於您選擇的語言和語音的預設設定，但使用 SSML 增強的文字不僅告訴 Amazon Polly 您想要說的內容，也告訴 Amazon Polly 您想要如何說出它。除了新增的 SSML 標籤之外，Amazon Polly 會以與合成純文字相同的方式合成 SSML 增強的文字。如需詳細資訊，請參閱[使用 Amazon Polly 合成語音範例](#)。

使用 SSML 時，請將整個文字括在 < speak > 標籤中，讓 Amazon Polly 知道您使用的是 SSML。例如：

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

然後，您使用特定的 SSML 標籤針對 < speak > 標籤內部文字來自訂說出文字的方式。您可以增加說話停頓點，改變更說話節奏、降低或提高音量，或新增其他許多自訂方式，讓文字說出的聲音完全符合您。如需您可使用的完整 SSML 標籤清單，請參閱[支援的 SSML 標籤](#)。

例如，您可以在文字內加入長時間停頓，或變更語音速度或音調。其他選項包括：

- 強調特定字詞或片語
- 使用音標發音
- 包括呼吸聲
- 輕聲低語
- 使用新聞播報員的說話風格。

如需 Amazon Polly 支援的 SSML 標籤及其使用方式的完整詳細資訊，請參閱[支援的 SSML 標籤](#)

使用 SSML 時，有幾個保留字元需要特殊處理。這是因為 SSML 使用這些字元做為其程式碼的一部分。為了使用它們，您可以使用特定實體來逸出它們。如需詳細資訊，請參閱[SSML 中的預留字元](#)

Amazon Polly 為這些類型的控制項提供[語音合成標記語言 \(SSML\) 1.1 版 W3C 建議所定義的 SSML 標記標籤子集](#)。

您可以在 Amazon Polly 主控台中使用 SSML，或使用 AWS CLI。下列主題示範如何使用 SSML 產生語音和控制輸出，讓它完全符合您的需求。

主題

- [SSML 中的預留字元](#)
- [在主控台上使用 SSML](#)
- [搭配 Synthesize-Speech 命令使用 SSML](#)
- [合成 SSML 增強文件](#)
- [支援的 SSML 標籤](#)

SSML 中的預留字元

有五個預先定義的字元不能在 SSML 陳述式中正常使用。這些實體由語言規格保留。這些字元如下所示：

語
音
程
式
碼

號
(雙
引
號)

'
符
號

j
引
號

j
於

名稱 程式碼 符號

& 符號

由於 SSML 使用這些字元做為其程式碼的一部分，若要在 SSML 中使用這些符號，您必須在使用時逸出該字元。您使用逸出碼而非實際字元，因此它可以正確顯示並且仍建立有效的 SSML 文件。例如，以下句子

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

將在 SSML 轉譯為

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

在此情況下，單引號和 & 符號的特殊字元會逸出，因此 SSML 文件仍然有效。

對於 &、< 和 > 符號，當您使用 SSML 時，一律需要逸出碼。此外，當您使用單引號 (') 做為單引號時，也必須使用逸出碼。

不過，當您使用雙引號 (") 或單引號 (') 做為引號時，是否使用逸出碼將取決於內容。

雙引號

- 在以雙引號分隔的屬性值中，必須逸出。例如，在以下 AWS CLI 程式碼中

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

- 在文字內容中不需要逸出。例如，在下列中

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

- 使用單引號分隔的屬性值時，不需要逸出。例如，在以下 AWS CLI 程式碼中

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

單引號

- 當用作單引號時，必須逸出。例如，在下列中

```
We've got to leave quickly.
```

- 在文字內容中不需要逸出。例如，在下列中

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

- 在以雙引號分隔的程式碼屬性中，不需要逸出。例如，在以下 AWS CLI 程式碼中

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

在主控台上使用 SSML

在下列範例中，您會使用 SSML 標籤，指示 Amazon Polly 在說出簡短段落時，將「全球資訊網協會」取代為「W3C」。您也可以使用標籤來引進停頓和輕聲低語。比較本練習的結果和[套用語彙（合成語音）](#)

如需包含範例的 SSML 的詳細資訊，請參閱[支援的 SSML 標籤](#)。

若要從 SSML 增強文字合成語音 (主控台)

1. 登入 AWS 管理主控台，並在 <https://console.aws.amazon.com/polly/>：// 開啟 Amazon Polly 主控台。
2. 如果尚未顯示，請選擇 Text-to-Speech (文字轉語音) 標籤。
3. 開啟 SSML。
4. 請在文字方塊中輸入或貼上下列文字：

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

SSML 標籤會告知 Amazon Polly 如何轉譯文字：

- `<break time="1s"/>` 指示 Amazon Polly 在前兩個句子之間暫停 1 秒。
- `<sub alias="World Wide Web Consortium">W3C</sub>` 指示 Amazon Polly 將 World Wide Web Consortium 取代為縮寫 W3C。
- `<amazon:effect name="whispered">Nice job</amazon:effect>` 告知 Amazon Polly 低語說出「良好任務」的第二個執行個體。

Note

當您使用時 AWS CLI，請以引號括住輸入文字，以區分輸入文字與周圍的程式碼。Amazon Polly 主控台不會顯示程式碼，因此您在使用時不會以引號括住輸入文字。

5. 針對語言，選擇英文、美國，然後選擇語音。
6. 若要接聽語音，請選擇接聽。
7. 若要儲存語音檔案，請選擇下載。如果您想要以不同的格式儲存，請展開其他設定，開啟語音檔案格式設定，然後選擇所需的格式，然後選擇下載。

搭配 Synthesize-Speech 命令使用 SSML

此範例說明如何使用 `synthesize-speech` 命令搭配 SSML 字串。使用 `synthesize-speech` 命令時，通常會提供下列項目：

- 輸入文字 (必要)
- 開頭和結尾標籤 (必要)
- 輸出格式
- 語音

在這個範例中，您將在引號中指定一個簡單的文字字串，以及所需的開頭和結尾 `<speak></speak>` 標籤。

⚠ Important

雖然您不會在 Amazon Polly 主控台的輸入文字周圍使用引號，但您必須使用 中的 AWS CLI 引號。也請務必區分輸入文字周圍的引號和個別標籤所需的引號。

例如，您可以使用標準引號 (") 括住輸入文字，並使用單引號 (') 括住內部標籤，或者反過來。兩種選項都適用於 Unix、Linux 和 macOS。不過，在 Windows 中，您必須使用標準引號括住輸入文字，以及使用單引號括住標籤。

對於所有作業系統，您可以使用標準引號 (") 括住輸入文字，並使用單引號 (') 括住內部標籤。例如：

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

對於 Unix、Linux 和 macOS，您也可以採用相反方式，亦即使用單引號 (') 括住輸入文字、使用標準引號 (") 括住內部標籤：

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

下列 AWS CLI 範例已針對 Unix、Linux 和 macOS 格式化。對於 Windows，將每一行尾端的反斜線 (\) Unix 接續字元取代為插入符號 (^)，並在輸入文字周圍使用完整引號 (") 取代內部標籤的單引號 (')。

```
aws polly synthesize-speech \  
--text-type ssm1 \  
--text '<speak>Hello world</speak>' \  
--output-format mp3 \  
--voice-id Joanna \  
speech.mp3
```

若要聆聽合成語音，請使用任何音訊播放器來播放產生的 `speech.mp3` 檔案。

合成 SSML 增強文件

對於較長的輸入文字，比較輕鬆的方法是將 SSML 內容儲存到檔案，然後在 `synthesize-speech` 命令中指定檔案名稱。例如，您可以將以下內容儲存至名為 `example.xml` 的檔案：

```
<?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 屬性將 en-US (美式英文) 指定為輸入文字的語言。如需輸入文字的語言和所選聲音的語言如何影響 SynthesizeSpeech 操作的相關資訊，請參閱[為特定單字指定另一種語言](#)。

若要執行 SSML 增強的檔案

1. 將 SSML 儲存到檔案 (例如 example.xml)。
2. 從 XML 檔案的存放路徑，執行下列 synthesize-speech 指令，並將輸入的文字換成 file:\example.xml，來將 SSML 檔案指定為輸入項目。因為這個命令會指向檔案，而非包含實際的輸入文字，因此無使用需引號

Note

下列 AWS CLI 範例已針對 Unix、Linux 和 macOS 格式化。用於 Windows 時，請以插入號 (^) 取代每一行結尾處的 Unix 接續字元斜線 (\)。

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

3. 若要聆聽合成語音，請使用任何音訊播放器來播放產生的 speech.mp3 檔案。

支援的 SSML 標籤

標準語音 <amazon:domain name="news"> 支援 以外的所有標籤。下表提供其他語音的標籤可用性。

Amazon Polly 支援下列 SSML 標籤：

Action	SSML 標籤	神經語音可用性	長格式語音可用性	生成式語音可用性
新增暫停	<break>	完整可用性	完整可用性	完整可用性
強調單字	<emphasis>	不適用	不適用	不適用
為特定單字指定另一種語言	<lang>	完整可用性	完整可用性	完整可用性
在文字中放置自訂標籤	<mark>	完整可用性	完整可用性	部分可用性
在段落之間新增暫停	<p>	完整可用性	完整可用性	完整可用性
使用音標發音	<phoneme>	完整可用性	完整可用性	部分可用性
控制音量、說話速率和音調	<prosody>	部分可用性	部分可用性	部分可用性
設定合成語音的最長持續時間	<prosody amazon:max-duration>	不適用	不適用	不適用
在句子之間新增暫停	<s>	完整可用性	完整可用性	完整可用性
控制如何說出特殊類型的單字	<say-as>	部分可用性	完整可用性	完整可用性
識別 SSML 增強的文字	<speak>	完整可用性	完整可用性	完整可用性
發音縮寫和縮寫	<sub>	完整可用性	完整可用性	完整可用性
透過指定語音部分來改善發音	<w>	完整可用性	完整可用性	完整可用性
新增呼吸聲	<amazon:auto-breaths>	不適用	不適用	不適用
新聞播報員說話風格	<amazon:domain name="news">	僅選取神經語音	不適用	不適用

Action	SSML 標籤	神經語音可用性	長格式語音可用性	生成式語音可用性
新增動態範圍壓縮	<amazon:effect name="drc">	完整可用性	完整可用性	不適用
柔和地說話	<amazon:effect phonation="soft">	不適用	不適用	不適用
控制波紋	<amazon:effect vocal-tract-length>	不適用	不適用	不適用
輕聲低語	<amazon:effect name="whispered">	不適用	不適用	不適用

Note

如果您以標準、神經或長格式使用不支援的 SSML 標籤，您會收到錯誤。

識別 SSML 增強的文字

<speak>

生成式、長格式、神經和標準 TTS 格式支援此標籤。

<speak> 標籤是所有 Amazon Polly SSML 文字的根元素。所有 SSML 增強文字必須包含在一組 <speak> 標籤中。

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

新增暫停

<break>

生成式、長格式、神經和標準 TTS 格式支援此標籤。

若要在文字中加入停頓，請使用 <break> 標籤。您可以根據強度設定停頓 (等同於逗號、句子或段落後的停頓)，或者您可以將它設定為以秒或毫秒計算的特定時間長度。如果您未指定屬性來判斷暫停長

度，Amazon Polly 會使用預設值，也就是 `<break strength="medium"/>`，這會在逗號之後新增暫停長度。

strength 屬性值：

- none：無停頓。使用 none 可移除正常發生的停頓，例如在句點之後的停頓。
- x-weak：具有 none 相同的強度，無停頓。
- weak：設定和逗號後的停頓相同持續時間的停頓。
- medium：具有 weak 相同的強度。
- strong：設定和句子後的停頓相同持續時間的停頓。
- x-strong：設定和段落後的停頓相同持續時間的停頓。

time 屬性值：

- `[number]`s：停頓期間，以秒為單位。最大期間為 10s。
- `[number]`ms：停頓期間，以毫秒為單位。最大期間為 10000ms。

例如：

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

如果未搭配 break 標籤來使用屬性，結果會因文字內容而有所不同：

- 如果 break 標籤旁沒有其他標點符號，該標籤會建立 `<break strength="medium"/>` (逗號長度停頓)。
- 如果標籤旁邊有逗號，會將標籤升級為 `<break strength="strong"/>` (句子長度停頓)。
- 如果標籤旁邊有句點，會將標籤升級為 `<break strength="x-strong"/>` (段落長度停頓)。

強調單字

`<emphasis>`

只有標準 TTS 格式才支援此標籤。

若要強調文字，請使用 `<emphasis>` 標籤。強調文字會變更語音速度和音量。越強調，Amazon Polly 說出的文字越大越慢。強調程度越小，則會更安靜、更迅速說出。若要指定強調程度，請使用 `level` 屬性。

`level` 屬性值：

- **Strong**：提高音量並減慢說話的速度，讓語音更大聲、更緩慢。
- **Moderate**：提高音量並減慢說話的速度，但幅度比 **strong** 還小。**Moderate** 是預設值。
- **Reduced**：降低音量並加快說話的速度。語音更輕柔、更快速。

Note

聲音的正常講話速度和音量落在 **moderate** 和 **reduced** 等級之間。

例如：

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

為特定單字指定另一種語言

`<lang>`

生成式、長格式、神經和標準 TTS 格式支援此標籤。對於生成語音，`<lang>` 標籤只能圍繞完整句子使用。

使用 `<lang>` 標籤可指定特定文字、片語或句子的其他語言。外國語言文字和片語以一組 `<lang>` 標籤括住時，語音轉譯的表現一般都會比較好。若要指定語言，請使用 `xml:lang` 屬性。如需可用語言的完整清單，請參閱 [Amazon Polly 中的語言](#)。

除非套用 `<lang>` 標籤，否則輸入文字中的所有字詞，都會以 `voice-id` 指定的語音語言說出。如果您套用 `<lang>` 標籤，文字會以該語言說出。

例如，如果 `voice-id` 是 Joanna（說美式英文），Amazon Polly 會以不含法文重音的 Joanna 語音說出以下內容：

```
<speack>
```

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

如果您使用 Joanna 語音搭配 `<lang>` 標籤，Amazon Polly 會以美式法文的 Joanna 語音說出句子：

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

由於 Joanna 不是以法文為母語，發音會以其母語語言 (美式英文) 為主。例如，完美的法語發音在發出文字 français 的 /r/ 時會帶有小舌顫音，但 Joanna 的美式英文語音則會發音為對應的聲音 /r/。

如果您使用說義大利文 voice-id 的 Giorgio 的 搭配下列文字，Amazon Polly 會使用義大利文發音，以 Giorgio 的聲音說出句子：

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

如果您使用相同的語音搭配下列 `<lang>` 標籤，Amazon Polly 會以義大利英文發音 Bruce Springsteen：

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

在合成語音時，此標籤也可用來做為選用 [DefaultLangCode](#) 選項的替代項目。不過，這樣就需要使用 SSML 來設定文字的格式。

在文字中放置自訂標籤

`<mark>`

長格式、神經和標準 TTS 格式支援此標籤。此標籤不會對生成語音執行任何操作，因為語音標記不適用於生成語音。

若要在文字中放置自訂標籤，請使用 `<mark>` 標籤。Amazon Polly 不會對標籤採取任何動作，但會在 SSML 中繼資料中傳回標籤的位置。只要維持以下格式，此標籤可以是您想呼叫的任何項目：

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

例如，假設標籤名稱為「動物」，而輸入文字是：

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

Amazon Polly 可能會傳回下列 SSML 中繼資料：

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

在段落之間新增暫停

<p>

生成式、長格式、神經和標準 TTS 格式支援此標籤。

若要在文字的段落之間加入停頓，請使用 <p> 標籤。使用此標籤可提供比母語人士通常在逗號或句子結尾處所用停頓更長的停頓時間。使用 <p> 標籤來括住段落：

```
<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>
```

這相當於使用 <break strength="x-strong"/> 來指定停頓。

使用音標發音

<phoneme>

長格式、神經和標準 TTS 格式支援此標籤。

若要讓 Amazon Polly 針對特定文字使用音標發音，請使用 <phoneme> 標籤。

<phoneme> 標籤需要兩個屬性。這些屬性表示 Amazon Polly 使用的語音字母，以及修正發音的音標：

- alphabet
 - ipa— 指出要使用的國際音標 (IPA)
 - x-sampa— 指出要使用的拓展音標字母評估法 (X-SAMPA)。
- ph
 - 指定發音的音標。如需詳細資訊，請參閱[Amazon Polly 中的語言](#)

透過 <phoneme> 標籤，Amazon Polly 會使用 ph 屬性指定的發音，而不是預設與所選語音使用的語言相關聯的標準發音。

例如，文字「pecan」可用兩種方式發音。在下列範例中，每行都會指派不同的發音給「pecan」。Amazon Polly 會如 ph 屬性中所指定發音，而不是使用預設發音。

國際音標 (IPA)

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

拓展音標字母評估法 (X-SAMPA)

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

中文使用拼音來發音。

拼音

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

日文使用 Yomigana 和發音 Kana。

Yomigana

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

發音 Kana

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

控制音量、說話速率和音調

<prosody>

標準 TTS 語音完全支援 Prosody 標籤屬性。生成式、神經式和長格式語音支援 volume 和 rate 屬性，但不支援 pitch 屬性。對於生成語音，prosody 標籤只能圍繞完整句子使用。

若要控制所選語音的音量、速度或音調，請使用 prosody 標籤。

音量、說話速度和音調取決於選擇的特定語音。除了不同語言語音之間的差異，講相同語言的各個語音之間也存在差異。因此，雖然所有語言的屬性類似，但各語言間有明顯變化，因此沒有絕對值可用。

prosody 標籤有三個屬性，三個屬性都有數個可用值來設定屬性。每個屬性使用相同的語法：

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

- volume
 - default：重設音量為目前語音的預設音量。
 - silent、x-soft、soft、medium、loud、x-loud：將音量設定為目前語音的預先定義值。
 - +ndB、-ndB：根據目前音量來變更音量。+0dB 的值代表無變動，+6dB 表示大約目前音量的兩倍，-6dB 則是指大約目前音量的一半。

例如，您可以設定一個段落的音量，如下所示：

```
<speack>
  Sometimes it can be useful to <prosody volume="loud">increase the volume
```

```
    for a specific speech.</prosody>
</speak>
```

或者，您可以用下列方式來設定音量：

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

- **rate**

- x-slow、slow、medium、fast、x-fast。將音調設定為所選語音的預先定義值。
- n%：說話速度的非負值百分比變更。例如，100% 表示說話速度無變化，200% 表示說話速度是預設速度的兩倍，以及 50% 表示說話速度率是預設速度的一半。這個值的範圍為 20 到 200%。

例如，您可以設定一個段落的說話速度，如下所示：

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

或者，您可以用下列方式來設定音量：

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

- **pitch**

- default：重設音調為目前語音的預設層級。
- x-low、low、medium、high、x-high：設定音調為目前語音的預先定義值。
- +n% 或 -n%：利用相對百分比來調整音調。例如，+0% 的值代表基準音高未變，+5% 會將基準音高提高一些，而 -5% 則會使基準音高降低一些。

例如，您可以設定一個段落的音調，如下所示：

```
<speak>
```

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

或者，您可以用下列方式來設定音量：

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

<prosody> 標籤必須至少包含一個屬性，但在相同標籤中可以包含多個屬性。

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

也可以和巢狀標籤結合使用，如下所示：

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

Note

目前<prosody>部分可用於生成式語音。

設定合成語音的最長持續時間

<prosody amazon:max-duration>

目前只有標準 TTS 格式支援此標籤。

若要控制語音合成時的持續時間長度，請使用 <prosody> 標籤搭配 amazon:max-duration 屬性。

取決於您所選擇的語音，合成語音的持續時間長度會稍有不同。因此，如果要將合成的語音，對應到視覺效果或需要精準對時的其他活動，可能會有點困難。對翻譯應用程式而言，這項問題會更嚴重，因為不同語言說出特定詞組所需的時間，可能會有很大的差異。

<prosody amazon:max-duration> 標籤會將合成語音對應到您所要的持續時間長度。

此標籤使用以下語法：

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

使用 <prosody amazon:max-duration> 標籤，您可以指定持續時間長度 (秒或毫秒)：

- *ns*：持續時間長度上限 (以秒為單位)
- *nms*：持續時間長度上限 (以毫秒為單位)

例如，下列口說文字的時間長度上限為 2 秒：

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

放置在標籤內的文字，不會超過指定的持續時間長度。如果選擇的語音或語言通常需要超過該持續時間，Amazon Polly 會加速語音，使其符合指定的持續時間。

如果指定的持續時間超過以正常速率讀取文字所需的時間，Amazon Polly 會正常讀取語音。&POL; 不會減慢語音的速度或加入靜音，因此產生的音訊會比所請求的短。

Note

Amazon Polly 增加的速度不超過正常速率的 5 倍。如果用超過這個標準的速度來說出文字，通常會聽不清楚。如果即使加快到最高速度，也無法符合您指定的持續時間長度時，會將音訊的速度加快，但持續時間將會超過指定的長度。

您可以在 <prosody amazon:max-duration> 標籤中包含單一句子或多個句子，也可以在文字內容中使用多個 <prosody amazon:max-duration>。

例如：

```

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

```

當 Amazon Polly 傳回合成語音時，使用 `<prosody amazon:max-duration>` 標籤可能會增加延遲。延遲的程度取決於文字內容及其長度。我們建議所使用的文字內容，最好包含相對較短的文字段落。

限制

使用 `<prosody amazon:max-duration>` 標籤的方式和此標籤搭配其他 SSML 標籤運作的方式，皆受到限制：

- `<prosody amazon:max-duration>` 標籤內所包含的文字，其長度不能超過 1500 個字元。
- 您不能嵌套 `<prosody amazon:max-duration>` 標籤。如果您將一個 `<prosody amazon:max-duration>` 標籤放入另一個標籤，Amazon Polly 會忽略內部標籤。

例如，在下列的範例內容中，`<prosody amazon:max-duration="5s">` 標籤會被忽略：

```

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

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

```

```
</prosody>
```

```
    We naturally understand this information, which is why speech is ideal for  
    creating applications where a screen isn't practical or possible, or simply isn't  
    convenient.
```

```
</prosody>
```

```
</speak>
```

- 您不能在使用 `<prosody>` 標籤搭配 `rate` 標籤中的 `<prosody amazon:max-duration>` 屬性。因為這兩者都會影響到口語文字的速度。

在下列範例中，Amazon Polly 會忽略 `<prosody rate="2">` 標籤：

```
<speak>
```

```
  <prosody amazon:max-duration="7500ms">
```

```
    Human speech is a powerful way to communicate.
```

```
    <prosody rate="2">
```

```
      Even a simple 'Hello' can convey a lot of information depending on the  
      pitch, intonation, and tempo.
```

```
    </prosody>
```

```
  </prosody>
```

```
</speak>
```

停頓和 **max-duration**

使用 `max-duration` 標籤時，您仍然可以在文字內容中插入停頓。不過，Amazon Polly 會在計算語音持續時間上限時包含暫停的長度。此外，Amazon Polly 會保留短停頓，其中逗號和句點放置在段落中，並包含在最長持續時間中。

例如，在下列的段落中，8 秒的語音包含了 600 毫秒的停頓，以及逗號和句點所造成的停頓。

```
<speak>
```

```
  <prosody amazon:max-duration="8s">
```

```
    Human speech is a powerful way to communicate.
```

```
    <break time="600ms"/>
```

```
    Even a simple 'Hello' can convey a lot of information depending on the pitch,  
    intonation, and tempo.
```

```
  </prosody>
```

```
</speak>
```

在句子之間新增暫停

<s>

生成式、長格式、神經和標準 TTS 格式支援此標籤。

若要在您的文字行或句子之間加入停頓，請使用 <s> 標籤。使用此標籤與下列方式有相同的效果：

- 使用句號 (.) 結束句子
- 使用 <break strength="strong"/> 指定停頓

與 <break> 標籤不同的是，<s> 標籤必須括住句子。這可用於依行 (而非句子) 分組的合成語音，例如詩詞。

在下列範例中，<s> 標籤在第一個句子和第二個句子之後，建立短暫的停頓。最後一個句子沒有 <s> 標籤，但其後也有短暫的停頓，因為以句號結尾。

```
<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>
```

控制如何說出特殊類型的單字

<say-as>

<say-as> 標籤由生成式、長形、神經和標準 TTS 引擎支援。不過請注意，如果 Amazon Polly 使用神經語音，並在執行時間遇到具有 characters 選項的 <say-as> 標籤，則會使用相關的標準語音合成受影響的句子。不過，受影響的句子仍會像使用神經語音一樣計費。

使用 <say-as> 標籤搭配 interpret-as 屬性，告訴 Amazon Polly 如何說出特定字元、單字和數字。這可讓您提供額外的內容，以消除 Amazon Polly 應如何呈現文字的任何模稜兩可之處。

<say-as> 標籤使用一個屬性 interpret-as，它使用許多可用的值。每個值使用相同的語法：

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

以下值可用於 interpret-as：

- `characters` 或 `spell-out`：拼出文字的每個字母，如 a-b-c。

Note

神經語音目前不支援此選項。如果您使用的是神經語音，且 Amazon Polly 在執行時間遇到此 SSML 程式碼，則會使用相關的標準語音合成受影響的句子。不過請注意，這句話仍會像使用神經語音一樣計費。

- `cardinal` 或 `number`：解讀數值文字為基數，如 1,234。
- `ordinal`：解讀數值文字為序號，如第 1,234 個。
- `digits`：個別拼出每個位數，如 1-2-3-4。
- `fraction`：解讀數值文字為分數。這可以處理常見的分數，例如 3/20，以及混合分數，例如 2 ½。如需詳細資訊，請參閱下方。
- `unit`：將數值文字解譯為測量單位。值應該是一個數字或後面接著單位的分數，中間沒有空格，如 1/2inch；或只跟著一個單位，如 1meter。
- `date`：將文字解譯為日期。日期的格式必須以格式屬性指定。如需詳細資訊，請參閱下方。
- `time`：將數值文字解譯為以分鐘和秒為單位的持續時間，例如 1'21"。
- `address`：將文字解譯為街道地址的一部分。
- `expletive`：「嗶聲處理」標籤中所包含的內容。
- `telephone`：將數值文字解譯為 7 位數或 10 位數的電話號碼，如 2025551212。您也可以使用此值來處理電話分機，如 2025551212x345。如需詳細資訊，請參閱下方。

Note

目前 `telephone` 選項不是所有語言都適用。然而，它可用於語音英文變體 (en-AU、en-GB、en-IN、en-US 和 en-GB-WLS)、西班牙文變體 (es-ES、es-MX 和 es-US)、法文變體 (fr-FR 和 fr-CA) 和葡萄牙文變體 (pt-BR 和 pt-PT)，以及德文 (de-DE)、義大利文 (it-IT)、日文 (ja-JP) 和俄羅斯文 (ru-RU)。另請注意，在某些情況下，阿拉伯文 (arb) 等語言會自動處理設定為電話號碼的號碼，因此請勿實際實作 `telephone` SSML 標籤。

分數

Amazon Polly 會將具有 `interpret-as="fraction"` 屬性的 `say-as` 標籤內的值解譯為常見分數。以下是分數的語法：

- 分數

語法：`##/##`，如 2/9。

例如：`<say-as interpret-as="fraction">2/9</say-as>` 發音為「九分之二」。

- 非負數的混合數字

語法：`##+##/##`，例如 3+1/2。

例如，`<say-as interpret-as="fraction">3+1/2</say-as>` 發音為「三又二分之一」。

 Note

「3」和「1/2」+之間必須有一個空格。Amazon Polly 不支援沒有空格的混合數字+，例如 "3 1/2"。

日期

當 `interpret-as` 設定為 `date`，您也需要指出日期的格式：

此標籤使用以下語法：

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

例如：

```
<say-as interpret-as="date" format="mdy">12-31-1900</say-as>.  
</say-as>
```

下列格式可以搭配 `date` 屬性使用。

- `mdy`：月-日-年。
- `dmy`：日-月-年。
- `ymd`：年-月-日。
- `md`：月-日。
- `dm`：日-月。
- `ym`：年-月。
- `my`：月-年。

- d : 日。
- m: 月。
- y: 年。
- yyyyymmdd : 年-月-日。如果您使用此格式，您可以使用問號讓 Amazon Polly 略過部分日期。

例如，Amazon Polly 會將下列項目轉譯為「9 月 22 日」：

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

不需要 Format。

電話

即使沒有 `<say-as>` 標籤，Amazon Polly 也會根據文字的格式，嘗試正確解譯您提供的文字。例如，如果您的文字包含 "202-555-1212"，Amazon Polly 會將其解譯為 10 位數的電話號碼，並個別說出每個數字，每個破折號都會短暫暫停。在這種情況下，您不需要使用 `<say-as interpret-as="telephone">`。不過，如果您提供文字 "2025551212"，並希望 Amazon Polly 將其說出為電話號碼，您可以指定 `<say-as interpret-as="telephone">`。

解譯每個元素的邏輯具有語言特殊性。例如，美式英文和英式英文對於電話號碼的發音方式各不相同 (在英式英文中，同一個數字連續群組在一起時，例如「兩個五」或「三個四」)。若想了解差異，您可以使用美國語音與英國語音測試以下範例：

```
<say-as interpret-as="telephone">2122241555</say-as>
```

發音縮寫和縮寫

`<sub>`

生成式、長格式、神經和標準 TTS 格式支援此標籤。

使用 `<sub>` 標籤搭配 `alias` 屬性，可取代所選文字的不同字詞 (或讀音)，例如首字母縮寫或縮寫。

此屬性使用語法：

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

在下列範例中，名稱「Mercury」(水銀) 取代為元素的化學符號，讓音訊內容更清楚。

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

透過指定語音部分來改善發音

<w>

生成式、長格式、神經和標準 TTS 格式支援此標籤。

您可使用 <w> 標籤，透過指定單字的部分語音或替代含意來自訂文字發音。這可使用 role 屬性來完成。

此標籤使用以下語法：

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

下列值可以用於 role 屬性：

指定語音部分：

- amazon:VB：將文字解釋為動詞 (現在式)。
- amazon:VBD：將單字解釋為過去的張量動詞。
- amazon:DT：將單字解釋為判斷器。
- amazon:IN：將單字解釋為述詞。
- amazon:JJ：將單字解釋為形容詞。
- amazon:NN：將單字解釋為名詞。

例如，視語音部分而定，美式英文的「read」讀音可依照標籤而改變：

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

若要指定特定意義：

- `amazon:DEFAULT`：使用單字的預設意義。
- `amazon:SENSE_1`：顯示時使用非預設的字意。例如，名詞「bass」的讀音會因含意的不同而不同。預設的含意是指音域的最低部分。其替代含意是指一種淡水魚，也叫做「bass」，但讀音不同。使用 `<w role="amazon:SENSE_1">bass</w>` 轉譯音訊文字的非預設發音 (淡水魚)。

如果您合成以下內容，則可以聽到發音和意義的差異：

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

Note

有些語言可能會有不同的口語支援功能選擇。

新增呼吸聲

`<amazon:breath>` 和 `<amazon:auto-breaths>`

只有標準 TTS 格式才支援此標籤。

自然音調的語音包含正確發音的字詞以及呼吸聲。新增呼吸聲至合成語音中，可讓語音聽起來更加自然。`<amazon:breath>` 和 `<amazon:auto-breaths>` 標籤提供呼吸聲。您有下列選項：

- 手動模式：您設定文字中呼吸聲的出現的位置、長度以及音量
- 自動化模式：Amazon Polly 會自動將呼吸聲插入語音輸出
- 混合模式：您和 Amazon Polly 都會新增呼吸聲

手動模式

在手動模式中，您將 `<amazon:breath/>` 標籤放入輸入文字內想要加入呼吸聲的位置。您可以分別使用 `duration` 和 `volume` 屬性來自訂呼吸聲的長度和音量：

- **duration**：控制呼吸聲的長度。有效值為：default、x-short、short、medium、long、x-long。預設值為 medium。
- **volume**：控制呼吸聲的音量。有效值為：default、x-soft、soft、medium、loud、x-loud。預設值為 medium。

Note

每個屬性值的確切長度和音量取決於使用的特定 Amazon Polly 語音。

若要使用預設值來設定呼吸聲，請使用沒有屬性的 `<amazon:breath/>`。

例如，若要使用屬性來設定呼吸聲的長度與音量為中等，可將屬性設定如下：

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

若要使用預設值，您只需使用標籤：

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

您可以新增呼吸聲到段落中，如下所示：

```
<speack>
    <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>
</speack>
```

自動模式

在自動化模式下，您可以使用 `<amazon:auto-breaths>` 標籤，指示 Amazon Polly 以適當的間隔自動建立呼吸雜訊。您可以設定間隔的頻率、呼吸聲音量以及呼吸聲長度。將 `</amazon:auto-breaths>` 標籤置於您想要加入自動呼吸聲的文字開頭，然後在尾端加上結束標籤。

 Note

與手動模式標籤不同，`<amazon:breath/>`、`<amazon:auto-breaths>` 標籤需要結束標記 (`</amazon:auto-breaths>`)。

您可以使用以下含有 `<amazon:auto-breaths>` 標籤的可選屬性：

- `volume`：控制呼吸聲的音量。有效值為：`default`、`x-soft`、`soft`、`medium`、`loud`、`x-loud`。預設值為 `medium`。
- `frequency`：控制呼吸聲在文字中發生的頻率。有效值為：`default`、`x-low`、`low`、`medium`、`high`、`x-high`。預設值為 `medium`。
- `duration`：控制呼吸聲的長度。有效值為：`default`、`x-short`、`short`、`medium`、`long`、`x-long`。預設值為 `medium`。

在預設情況下，呼吸聲的頻率將取決於輸入文字。不過，呼吸聲經常出現在逗號和句點之後。

下列範例示範如何使用 `<amazon:auto-breaths>` 標籤：若要決定您的內容要使用哪些選項，請將適用的範例複製到 Amazon Polly 主控台，並聆聽差異。

- 使用不含選用參數的自動模式。

```
<speach>
  <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>
```

- 使用含有音量控制的自動模式。未指定參數 (duration 與 frequency) 皆設為預設值 (medium)。

```
<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>
```

- 使用含有頻率控制的自動模式。未指定參數 (duration 與 volume) 皆設為預設值 (medium)。

```
<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>
```

- 使用含有多個參數的自動模式。對於未指定的Duration參數，Amazon Polly 會使用預設值 (medium)。

```
<speak>
  <amazon:auto-breaths volume="x-loud" frequency="x-low">Amazon Polly is a service
that turns
```

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

新聞播報員說話風格

<amazon:domain name="news">

新聞播報員風格僅適用於 Matthew 或 Joanna 語音，僅提供美式英文 (en-US)、Lupe、美式西班牙文 (es-US) 和 Amy、英式英文 (en-GB)。只有使用 Neural 格式時才支援。

若要使用新聞播報員風格，請使用 SSML 標籤和下列語法：

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

例如，您可以將新聞播報員風格與 Amy 語音搭配使用，如下所示：

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

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

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

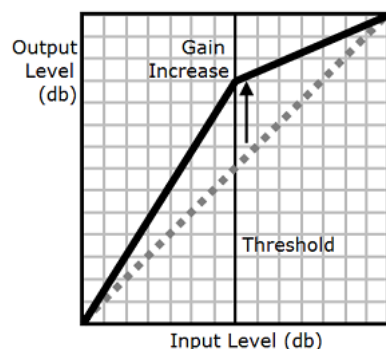
新增動態範圍壓縮

`<amazon:effect name="drc">`

長格式、神經和標準 TTS 格式支援此標籤。

根據音訊檔中所使用的文字、語言和語音，聲音範圍從柔和到大聲。環境音例如移動中車輛的聲音，通常可以遮罩為較柔和的聲音，使得音軌難以聽得清楚。若要增強音訊檔中特定聲音的音量，請使用動態範圍壓縮 (drc) 標記。

drc 標籤會為您的音訊設定中音的「響度」閾值，並提高該閾值附近聲音的音量 (增益)。最靠近閾值套用最大增益，遠離閾值則增益隨之減少。



這可讓中間範圍的聲音在吵雜環境中比較容易聽到，使得整個音訊檔更清晰。

drc 標記是一種布林值參數 (有或沒有)。它使用語法：`<amazon:effect name="drc">` 並以 `</amazon:effect>` 結束。

您可以使用 drc 標籤搭配 Amazon Polly 支援的任何語音或語言。您可以將其套用到錄音的整個部分或只幾個字。例如：

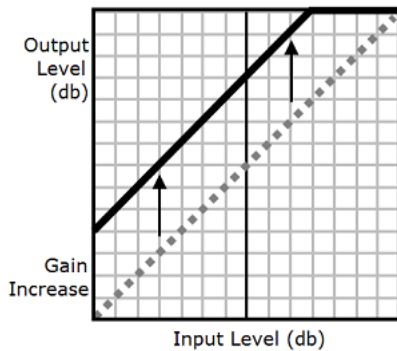
```
<speak>
  Some audio is difficult to hear in a moving vehicle, but <amazon:effect
name="drc"> this audio
  is less difficult to hear in a moving vehicle.</amazon:effect>
</speak>
```

Note

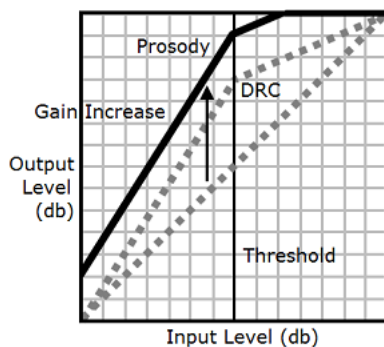
當您使用語法中的 drc 時，它會區分大小寫。amazon:effect

使用 **drc** 搭配 **prosody volume** 標籤

如下圖所示，prosody volume 標記從原始層級 (虛線) 到調整層級 (實線)，平均地提高整個音訊檔案的音量。若要進一步提高檔案特定部分的音量，請使用 drc 標記搭配 prosody volume 標記。結合使用標記不會影響 prosody volume 標記的設定。



當您同時使用 drc 和 prosody volume 標籤時，Amazon Polly 會先套用 drc 標籤，增加中間範圍聲音 (接近閾值的聲音)。接著套用 prosody volume 標記並進一步平均提高整個音軌的音量。



若要一起使用標記，請將其中一巢套到另一個內部。例如：

```
<speake>
  <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>
</speake>
```

在這個文字中，prosody volume 標記會提高整段的音量為「大聲」。drc 標記會在第二個句子增強中間範圍值的音量。

Note

一起使用 drc 和 prosody volume 標記時，對於巢狀標記請使用標準 XML 做法。

柔和地說話

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

目前只有標準 TTS 格式支援此標籤。

若要指定輸入文字的說出聲音應該比正常聲音更柔和，請使用 `<amazon:effect phonation="soft">` 標籤。

此屬性使用語法：

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

例如，您可以使用此標籤搭配 Matthew 語音，如下所示：

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

控制波紋

`<amazon:effect vocal-tract-length>`

目前只有標準 TTS 格式支援此標籤。

音色是語音的音質，可協助您分辨語音間的不同，甚至當其音調和大小聲相同時也能分辨出。決定語音音色的其中一個最重要生理特徵是聲道的長度，這是範圍從聲帶頂部到嘴唇邊緣的一個氣腔。

若要控制 Amazon Polly 中輸出語音的波紋，請使用 `vocal-tract-length` 標籤。此標記具有變更喇叭聲道長度的效果，聽到類似喇叭的大小有所變更。當您增加 `vocal-tract-length` 時，喇叭實際聽起來越大聲。當您縮短時，喇叭聲音會變小。您可以將此標籤與 Amazon Polly Text-to-Speech 產品組合中的任何語音搭配使用。

若要變更音色，請使用以下值：

- $+n\%$ 或 $-n\%$ ：按照目前語音的相對百分比變動來調整聲道長度。例如， $+4\%$ 或 -2% 。有效值範圍從 $+100\%$ 到 -50% 。這個範圍以外的值會剪除。例如， $+111\%$ 聽起來類似 $+100\%$ 而 -60% 聽起來像 -50% 。

- $n\%$ ：變更聲道長度為目前語音的聲道長度的絕對百分比。例如，110% 或 75%。絕對值 110% 等於相對值 +10%。絕對值 100% 與目前語音的預設值相同。

以下範例說明如何透過變更聲道長度來變更音色：

```
< 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>
```

結合多個標記

您可以結合 `vocal-tract-length` 標籤與 Amazon Polly 支援的任何其他 SSML 標籤。由於音色 (聲道長度) 和音調密切連結，您可以透過使用 `vocal-tract-length` 和 `<prosody pitch>` 標記獲得最佳結果。若要產生最逼真的語音，建議您針對兩個標籤使用不同的百分比變化。試驗各種組合來獲得您想要的結果。

下列範例顯示如何結合標記。

```
< 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>
```

輕聲低語

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

目前只有標準 TTS 格式支援此標籤。

此標籤表示輸入文字應以低聲而非一般語音說出。這可與 Amazon Polly Text-to-Speech 產品組合中的任何語音搭配使用。

此標籤使用以下語法：

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

例如：

```
<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>
```

在這種情況下，角色所說的合成語音是低語，但「她說」片語是在所選 Amazon Polly 語音的正常合成語音中說出的。

根據想要的效果，您可以透過將韻律率減慢最多 10%，來增強「低語」效果。

例如：

```
<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>
```

當產生低聲語音的語音標記時，音訊串流還必須包含低聲語音，以確保語音標記符合音訊串流。

管理語彙

發音語彙可讓您自訂字詞的發音。Amazon Polly 提供 API 操作，可讓您用來在 AWS 區域中存放語彙。之後這些語彙為該特定區域專屬。使用 SynthesizeSpeech 作業合成文字時，您可以使用來自該區域的一個或更多語彙。將會在合成開始時，將指定的語彙套用至輸入文字。如需詳細資訊，請參閱[SynthesizeSpeech](#)。

Note

這些語彙必須符合發音語彙規格 (PLS) W3C 建議。如需更多詳細資訊，請參閱 W3C 網站上的[發音語彙規格 \(PLS\) 版本 1.0](#)。

以下範例展示以語音合成引擎使用語彙的方式：

- 常見的單詞有時會以數字替代字母位置的方式格式化，如何「g3t sm4rt」(變得有智慧)。人類可以正確讀取這些單詞。不過，文字轉語音 (TTS) 引擎逐字地讀取文字，完全依照名稱拼法發音。您可以在此處使用 Amazon Polly 利用語彙來自訂合成語音。在此範例中，您可以為語彙中的「g3t sm4rt」指定別名 (變得有智慧)。
- 您的文字可能包含縮寫，例如 W3C。您可以使用語彙來定義單字 W3C 的別名，以完整展開的形式讀取 (全球資訊網協會)。

語彙可讓您進一步控制 Amazon Polly 如何向所選語言發出不常見的單字。例如，您可以使用音標指定發音。如需更多詳細資訊，請參閱 W3C 網站上的[發音語彙規格 \(PLS\) 版本 1.0](#)。

主題

- [使用多個語彙](#)
- [上傳語彙](#)
- [套用語彙 \(合成語音\)](#)
- [在主控台上篩選語彙清單](#)
- [在主控台下載語彙](#)
- [刪除語彙](#)

使用多個語彙

您最多可以指定五個語彙到您的文字。如果套用到您的文字的一個以上的語彙中都出現相同語素，那麼其套用順序會使產生的語音有所不同。例如，假設以下文字「您好，我的名字是 Bob」。而且不同語彙中的兩個詞元都使用語素 Bob。

LexA

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

LexB

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

如果語彙排列順序為 LexA 在先，LexB 在後，那麼合成語音將會是「Hello，我的名稱叫 Robert。」
如果語彙排列順序為 LexB 在先，LexA 在後，那麼合成語音為「Hello，我的名稱叫 Bobby。」

Example- 將 LexA 套用在 LexB 之前

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

語音輸出：「Hello，我的名稱叫 Robert。」

Example- 將 LexB 套用在 LexA 之前

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

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

語音輸出：「Hello，我的名稱叫 Bobby。」

如需使用 Amazon Polly 主控台套用語彙的資訊，請參閱 [套用語彙（合成語音）](#)。

上傳語彙

您使用的語彙必須符合發音語彙規格 (PLS) W3C 建議。如需更多詳細資訊，請參閱 W3C 網站上的 [發音語彙規格 \(PLS\) 版本 1.0](#)。

Console - Lexicons tab

若要使用發音語彙，您必須先上傳該發音語彙。在主控台有兩種位置可供上傳語彙：文字轉語音標籤和語彙標籤。

以下程序描述如何新增語彙，供您用於自訂所選語言稀有單字和片語的發音方式。

從語彙索引標籤新增語彙

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/polly/> 的 Amazon Polly 主控台。
2. 選擇 Lexicons (語彙) 索引標籤。
3. 選擇上傳語彙。
4. 提供語彙的名稱，然後使用選擇語彙檔案來尋找要上傳的語彙。您只能上傳副檔名為 .pls 或 .xml 的 PLS 檔案。
5. 選擇上傳語彙。如果同名語彙（無論是 .pls 或 .xml 檔案）已存在，上傳語彙會覆寫現有的語彙。

Console - TTS tab

從text-to-Speech索引標籤新增語彙

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/polly/> 的 Amazon Polly 主控台。
2. 選擇 Text-to-Speech (文字轉換語音) 索引標籤。
3. 展開其他設定，開啟自訂發音，然後選擇上傳語彙。

4. 提供語彙的名稱，然後使用選擇語彙檔案來尋找要上傳的語彙。您只能將 PLS 檔案與 .pls 或 .xml 副檔名搭配使用。
5. 選擇上傳語彙。如果已存在同名語彙（無論是 .pls 或 .xml 檔案），上傳語彙會覆寫現有的語彙。

AWS CLI - one lexeme

使用 Amazon Polly，您可以使用 [PutLexicon](#) 將發音語彙存放在您帳戶的特定 AWS 區域中。然後，您可以在您的 [SynthesizeSpeech](#) 請求中指定一個或多個儲存的語彙，以在服務開始合成這些文字之前套用。如需詳細資訊，請參閱[管理語彙](#)。

考慮以下 W3C PLS 相容的語彙。

```
<?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>
```

注意下列事項：

- 此兩個屬性指定於 <lexicon> 元素：
 - xml:lang 屬性會指定 en-US 語彙適用的語言代碼。如果您在 SynthesizeSpeech 通話中指定的語音具有相同的語言代碼 (en-US)，Amazon Polly 可以使用此範例語彙。

Note

您可以使用 DescribeVoices 操作來尋找和語音相關聯的語言代碼。

- 此 `alphabet` 屬性指定 IPA，其表示國際音標用於發音。IPA 是撰寫發音的字母之一。Amazon Polly 也支援拓展語音評估方法音標字母 (X-SAMPA)。
- `<lexeme>` 元素描述 `<grapheme>` (也就是單字的文字表示) 和 `<alias>` 之間的映射。

要測試此語彙，請依下列步驟執行：

1. 將語彙儲存為 `example.pls`。
2. 執行 `put-lexicon` AWS CLI 命令，將語彙（名稱為 `w3c`）存放在 `us-east-2` 區域中。

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

3. 執行 `synthesize-speech` 命令將範例文字合成到音訊串流 (`speech.mp3`)，然後指定選用的 `lexicon-name` 參數

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

4. 播放產生的 `speech.mp3`，並留意「全球資訊網協會」已取代 W3C 這個字。

前述範例語彙使用別名。未使用語彙中提及的 IPA 字母。以下語彙指定使用 `<phoneme>` 元素與 IPA 字母的語音發音。

```
<?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>

```

請遵循相同步驟來測試此語彙。請務必指定具有「pecan」一詞的輸入文字（例如，「Pecan pie 很可口」）。

如需 PutLexicon API 操作的其他程式碼範例，請參閱下列資源：

- Java 範例：[PutLexicon](#)
- Python (Boto3) 範例[PutLexicon](#)

AWS CLI - multiple lexemes

使用 Amazon Polly，您可以使用 [PutLexicon](#) 將發音語彙存放在您帳戶的特定 AWS 區域中。然後，您可以在您的 [SynthesizeSpeech](#) 請求中指定一個或多個儲存的語彙，以在服務開始合成這些文字之前套用。如需詳細資訊，請參閱[管理語彙](#)。

在這個範例中，您在語彙中指定的詞元僅適用於合成的輸入文字。請考量下列語彙：

```

<?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>

```

該語彙指定三個詞元，其中兩個為語素 W3C 定義別名，如下所示：

- 第一個 <lexeme> 元素定義別名 (全球資訊網協會)。
- 第二個 <lexeme> 定義另一種別名 (WWW 協會)

Amazon Polly 使用第一個取代語彙中任何指定圖形。

第三個 <lexeme> 定義協會這個字的取代 (社群)。

首先，我們來測試這個語彙。假設您要合成以下範例文字到音訊檔案 (speech.mp3)，而您將呼叫中的語彙指定為 SynthesizeSpeech。

```
The W3C is a Consortium
```

SynthesizeSpeech 先套用的語彙如下所示：

- 就第一個詞元，W3C 這個字修訂為「全球資訊網協會」。修訂文字如下所示：

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

- 第三個詞元中所定義的別名僅適用於「協會」這個字 (原始文字的一部分)，產生以下文字：

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

您可以使用 來測試此項目 AWS CLI，如下所示：

1. 將語彙儲存為 example.pls。
2. 執行 put-lexicon 命令，以名稱 w3c 將語彙儲存在 us-east-2 區域中。

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

3. 執行 list-lexicons 命令，驗證 w3c 語彙在傳回的語彙清單中。

```
aws polly list-lexicons
```

4. 執行 synthesize-speech 命令將範例文字合成到音訊檔案 (speech.mp3)，然後指定選用的 lexicon-name 參數

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

5. 播放產生的 speech.mp3 檔案，確認合成語音反映文字變更。

如需 PutLexicon API 操作的其他程式碼範例，請參閱下列資源：

- Java 範例：[PutLexicon](#)
- Python (Boto3) 範例[PutLexicon](#)

套用語彙（合成語音）

您使用的語彙必須符合發音語彙規格 (PLS) W3C 建議。如需更多詳細資訊，請參閱 W3C 網站上的[發音語彙規格 \(PLS\) 版本 1.0](#)。

Console

下列程序示範如何將語彙套用到您的輸入文字，方法是套用 W3c.pls 語彙，用「全球資訊網協會」取代「W3C」。如果您套用多個語彙到您的文字，會以從上到下順序套用，第一個相符優先於之後的相符。如果語彙中指定的語言和所選語言相同，語彙僅會套用到文字。

您可以套用語彙至純文字或 SSML 輸入。

Example- 套用 W3C.pls 語彙

如需建立此練習所需的語彙，請參閱[上傳語彙](#)。使用純文字編輯器來建立主題最上方顯示的 W3C.pls 語彙。請記住您儲存此檔案的地方。

將 W3C.pls 語彙套用到您的輸入

在此範例中，我們會引進語彙，以「全球資訊網協會」取代「W3C」。將此練習的結果與英文和其他語言的[在主控台上使用 SSML](#) 結果做比較。

1. 登入 AWS 管理主控台，並在 <https://console.aws.amazon.com/polly/> 開啟 Amazon Polly 主控台。

2. 執行以下任意一項：

- 關閉 SSML，然後在文字輸入方塊中輸入或貼上此文字。

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

- 開啟 SSML，然後在文字輸入方塊中輸入或貼上此文字。

```
<speack>He wasn't paying attention.<break time="1s"/>  
In the middle of the 10/3/2014 W3C meeting  
he shouted, "Score!" quite loudly.</speack>
```

- 從語言清單中，選擇英文、美國，然後選擇您要用於此文字的語音。
- 展開其他設定並開啟自訂發音。
- 從語彙清單中選擇 W3C (English, US)。

如果沒有列出 W3C (English, US) 語彙，請選擇 Upload lexicon (上傳語彙) 和上傳，然後從清單中選擇它。若要建立此語彙訊，請參閱 [上傳語彙](#)。

- 若要立即接聽語音，請選擇接聽。
- 若要將語音儲存至檔案，
 - 選擇 Download (下載)。
 - 若要變更為不同的檔案格式，請開啟語音檔案格式設定，選擇您想要的檔案格式，然後選擇下載。

重複之前的步驟，但選擇不同的語言，並留意輸出中的差異。

AWS CLI

在 SynthesizeSpeech 的呼叫中，您可以指定多個語彙。在此例中，第一個指定的語彙 (從左到右) 覆寫前面的語彙。

請考量下列兩個語彙。請注意，每個語彙描述相同語素 W3C 的不同別名。

- 語彙 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>
```

- 語彙 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>
```

假設您將這些語彙分別儲存為 w3c 和 w3cAlternate。如果您在 SynthesizeSpeech 呼叫中依順序 (w3c 在前，w3cAlternate 在後) 指定語彙，第一個語彙所定義的 W3C 別名優先於第二個。要測試語彙，請依下列步驟執行：

1. 將語彙儲存在本機檔案 w3c.pls 和 w3cAlternate.pls 中。
2. 使用 put-lexicon AWS CLI 命令上傳這些語彙。
 - 上傳 w3c.pls 語彙，並將它儲存為 w3c。

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

- 上傳服務上的 `w3cAlternate.pls` 語彙做為 `w3cAlternate`。

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

3. 執行 `synthesize-speech` 命令將範例文字合成到音訊串流 (`speech.mp3`)，然後使用 `lexicon-name` 參數指定兩種語彙

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

4. 測試產生的 `speech.mp3`。其讀取如下：

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

在主控台上篩選語彙清單

下列程序說明如何篩選語彙清單，僅顯示所選語言的語彙。

Console

依語言篩選列出的語彙

1. 登入 AWS 管理主控台，並在 <https://console.aws.amazon.com/polly/> 開啟 Amazon Polly 主控台。
2. 選擇 Lexicons (語彙) 索引標籤。
3. 選擇任何語言。
4. 從語言清單選擇您要篩選的語言。

清單只顯示所選語言的語彙。

AWS CLI

Amazon Polly 提供 [ListLexicons](#) API 操作，您可以用來取得您帳戶在特定 AWS 區域中的發音語彙清單。下列 AWS CLI 呼叫會列出您帳戶中 us-east-2 區域的語彙。

```
aws polly list-lexicons
```

以下是範例回應，顯示兩個名為 w3c 和 tomato 的語彙。對於每個語彙，回應傳回中繼資料如語彙套用的語言代碼、語彙中定義的詞元數、以位元組為單位的大小等等。語言代碼描述語彙套用中所定義詞元的語言和地區設定。

```
{
  "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"
    }
  ]
}
```

下列資源包含 ListLexicons 操作的其他資訊：

- Java 範例：[ListLexicons](#)

- Python (Boto3) 範例 [ListLexicon](#)

在主控台下載語彙

以下程序說明如何下載一或多個語彙。您可以新增、移除或修改檔案中的語彙項目，然後再次上傳，使您的語彙再次保持最新狀態。

Console

下載一或多個語彙

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/polly/> 的 Amazon Polly 主控台。
2. 選擇 Lexicons (語彙) 索引標籤。
3. 選擇您要下載的一個或多個語彙。
 - a. 若要下載單一語彙，請自清單中選擇其名稱。
 - b. 若要以單一壓縮封存檔案形式下載多個 (語彙)，選取清單中您要下載的每個項目。
4. 選擇 Download (下載)。
5. 開啟您想要下載語彙的資料夾。
6. 選擇儲存。

AWS CLI

Amazon Polly 提供 [GetLexicon](#) API 操作，以擷取您在帳戶中特定區域中存放的發音語彙內容。

下列 `get-lexicon` AWS CLI 命令會擷取 `example` 語彙的內容。

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

如果您帳戶中沒有儲存語彙，您可以使用 `PutLexicon` 操作儲存一個語彙。如需詳細資訊，請參閱 [上傳語彙](#)。

以下是範例回應。除了語彙內容外，回應中傳回中繼資料如語彙套用的語言代碼、語彙中定義的詞元數、Amazon Resource Name (ARN) 資源，以及以位元組為單位的語彙大小。LastModified 值是 Unix 時間戳記。

```
{
  "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
  }
}
```

下列資源包含 GetLexicon 操作的其他程式碼範例：

- Java 範例：[GetLexicon](#)
- Python (Boto3) 範例[GetLexicon](#)

刪除語彙

以下程序說明如何刪除語彙。刪除語彙之後，您必須將其加回來，才能在再次使用。您可以同時刪除一或多個語彙，方法是勾選個別語彙的核取方塊。

Console

刪除語彙

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/polly/> 的 Amazon Polly 主控台。
2. 選擇 Lexicons (語彙) 索引標籤。
3. 選擇您要從清單刪除的一或多個語彙。
4. 選擇 刪除。
5. 輸入確認文字，然後選擇刪除以從區域移除語彙，或選擇取消以保留語彙。

AWS CLI

Amazon Polly 提供 [DeleteLexicon](#) API 操作，可從您帳戶中的特定 AWS 區域刪除發音語彙。以下內容會 AWS CLI 刪除指定的語彙。

下列 AWS CLI 範例已針對 Unix、Linux 和 macOS 格式化。對於 Windows，將每一行結尾的反斜線 (\) Unix 接續字元取代為插入符號 (^)，並在輸入文字周圍使用完整引號 (") 取代內部標籤的單引號 (')。

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

下列資源包含 DeleteLexicon 操作的其他資訊：

- Java 範例：[DeleteLexicon](#)
- Python (Boto3) 範例[DeleteLexicon](#)

長音訊檔案

若要為大量文字段落建立 TTS 檔案，請使用 Amazon Polly 的非同步合成功能。這會用到三個 `SpeechSynthesisTask` API：

- `StartSpeechSynthesisTask`：開始執行新的合成作業。
- `GetSpeechSynthesisTask`：針對先前提交的合成作業傳回詳細資訊。
- `ListSpeechSynthesisTasks`：列出所有已提交的合成作業。

`SynthesizeSpeech` 作業會以近乎即時速度產生音訊，而且在大多數情況中的延遲相當少。為了實現此種效能，這項作業只能合成 3000 個字元。

Amazon Polly 的非同步合成功能透過變更文件的合成和傳回方式，克服處理較大文字文件的挑戰。使用提交輸入文字提出合成請求時 `StartSpeechSynthesisTask`，Amazon Polly 會將請求排入佇列，然後在系統資源可用時立即在背景中非同步處理這些請求。然後，Amazon Polly 會將產生的語音或語音標記串流直接上傳至（必要）Amazon Simple Storage Service (Amazon S3) 儲存貯體，並透過（選用）SNS 主題通知您已完成檔案的可用性。

利用這種方式，長度最多包含 100,000 個應計費字元 (或總計 200,000 個字元) 的文字，即可使用除了近乎即時處理以外的所有功能。

若要使用此方法合成文件，您必須擁有可寫入的 Amazon S3 儲存貯體，才能儲存音訊檔案。藉由提供選用的 SNS 主題識別符，您可以在合成音訊準備完成時收到通知。當合成任務完成時，Amazon Polly 將發佈該主題的訊息。此訊息也可能包含有用的錯誤資訊，這些資訊會在合成作業失敗時顯示。若要這麼做，請確定建立合成作業的使用者，也可以發佈到 SNS 主題。關於建立和訂閱 SNS 主題的方法，詳細資訊請參閱 [Amazon SNS 文件](#)。

加密

您可以根據需要，將加密的輸出檔案儲存於 S3 儲存貯體中。若要這麼做，您可以啟用 [Amazon S3 儲存貯體加密](#)，這項功能使用目前最強大的其中一種區塊加密法 (256 位元進階加密標準 (AES-256))。

主題

- [設定非同步合成的 IAM 政策](#)
- [建立長時間音訊檔案](#)

設定非同步合成的 IAM 政策

為了使用非同步合成功能，將需要 IAM 政策來允許下列動作：

- 使用新的 Amazon Polly 操作
- 寫入輸出的 S3 儲存貯體
- 發佈到狀態 SNS 主題 [選擇性]

下列的政策只授與進行非同步合成所需的必要權限，而且可以連結到 IAM 使用者。

建立長時間音訊檔案

您可以使用 Amazon Polly 主控台，透過與 搭配使用的相同功能，使用非同步合成來建立長語音 AWS CLI。如同其他所有的合成作業，這項作業也是在 Text-to-Speech (文字轉語音) 標籤完成。

Console

其他的非同步合成功能，也可透過主控台使用。S3 synthesis tasks (S3 合成作業) 標籤反映 ListSpeechSynthesisTasks 功能，會顯示儲存到 S3 儲存貯體的所有作業，並可讓您根據需要進行篩選。按一下特定的單一作業來顯示其詳細資訊 (代表 GetSpeechSynthesisTask 功能)。

使用 Amazon Polly 主控台合成大型文字

1. 登入 AWS 管理主控台 並開啟位於 <https://console.aws.amazon.com/polly/> 的 Amazon Polly 主控台。
2. 選擇 Text-to-Speech (文字轉換語音) 索引標籤。如果適當，請選取長表單做為引擎。
3. 在開啟或關閉 SSML 的情況下，將文字輸入或貼到輸入方塊中。
4. 為您的文字選擇語言、區域和語音。
5. 選擇儲存至 S3。

Note

如果文字長度超過即時SynthesizeSpeech操作的 3,000 個字元限制，則下載和接聽選項都會呈現灰色。

6. 主控台會開啟表單，讓您可以選擇要存放輸出檔案的位置。

- a. 填寫目的地 Amazon S3 儲存貯體的名稱。
- b. 或者，您可以填寫輸出的前綴金鑰。

 Note

輸出 S3 儲存貯體必須可以寫入。

- c. 如果您想要在合成任務完成時收到通知，請提供選用的 SNS 主題識別符。

 Note

SNS 必須開放給目前的主控台使用者，來發佈訊息，才能使用此選項。如需詳細資訊，請參閱 [Amazon Simple Notification Service \(SNS\)](#)

- d. 選擇儲存至 S3。

擷取語音合成作業的相關資訊

1. 在主控台中，選擇 S3 Synthesis Tasks (S3 合成作業) 標籤。
2. 作業會依日期順序顯示。若要依狀態篩選任務，請選擇所有狀態，然後選擇要使用的狀態。
3. 若要查看特定作業的詳細資訊，請選擇連結的 Task ID (作業 ID)。

AWS CLI

Amazon Polly 非同步合成功能使用三個 `SpeechSynthesisTask` APIs 來處理大量文字：

- `StartSpeechSynthesisTask`：開始執行新的合成作業。
- `GetSpeechSynthesisTask`：針對先前提交的合成作業傳回詳細資訊。
- `ListSpeechSynthesisTasks`：列出所有已提交的合成作業。

合成大量的文字 (`StartSpeechSynthesisTask`)

如果您想要建立一個音訊檔案，其大小超過您可以使用即時 `SynthesizeSpeech` 所建立的檔案時，請使用 `StartSpeechSynthesisTask` 作業。除了 `SynthesizeSpeech` 操作所需的引數之外，`StartSpeechSynthesisTask` 還需要 Amazon S3 儲存貯體的名稱。您也可使用其他兩種選用的引數：輸出檔案的金鑰前綴，和 SNS 主題的 ARN (如果想要收到關於作業狀態的通知)。

- `OutputS3BucketName`：應上傳合成的 Amazon S3 儲存貯體名稱。此儲存貯體應與 Amazon Polly 服務位於相同的區域。此外，用於進行呼叫的 IAM 使用者應有權存取儲存貯體。[必要]
- `OutputS3KeyPrefix`：輸出檔案的金鑰前綴。如果您想要將輸出語音檔案儲存在儲存貯體中的自訂目錄式索引鍵中，請使用此參數。[選用]
- `SnsTopicArn`：如果您想要接收任務狀態的通知，要使用的 SNS 主題 ARN。此 SNS 主題應與 Amazon Polly 服務位於相同的區域。此外，用於進行呼叫的 IAM 使用者應有權存取 主題。[選用]

例如，下列範例可用來在美國東部（俄亥俄）區域執行 `start-speech-synthesis-task` AWS CLI 命令：

下列 AWS CLI 範例已針對 Unix、Linux 和 macOS 格式化。對於 Windows，將每一行結尾的反斜線 (\) Unix 接續字元取代為插入符號 (^)，並在輸入文字周圍使用完整引號 (") 取代內部標籤的單引號 (')。

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

這將會產生類似於下列的回應：

```
"SynthesisTask":
{
  "OutputFormat": "mp3",
  "OutputUri": "https://s3.us-east-2.amazonaws.com/your-bucket-name/optional/
prefix/path/file.<task_id>.mp3",
  "TextType": "text",
  "CreationTime": [...],
  "RequestCharacters": [...],
  "TaskStatus": "scheduled",
  "TaskId": [task_id],
  "VoiceId": "Joanna"
}
```

`start-speech-synthesis-task` 作業會傳回幾個新的欄位：

- `OutputUri` : 輸出語音檔案的所在位置。
- `TaskId` : Amazon Polly 產生的語音合成任務的唯一識別符。
- `CreationTime` : 作業最初提交時的時間戳記。
- `RequestCharacters` : 作業中的應計費字元數目。
- `TaskStatus` : 針對已提交作業的狀態，提供相關的資訊。

當作業提交時，初始狀態會顯示為 `scheduled`。當 Amazon Polly 開始處理任務時，狀態會變更為 `inProgress`，稍後會變更為 `completed` 或 `failed`。在呼叫 `GetSpeechSynthesisTask` 或 `ListSpeechSynthesisTasks` 操作時，如果作業失敗，會傳回錯誤訊息。

任務完成後，語音檔案將會出現於 `OutputUri` 中所指定的位置，以提供使用。

擷取語音合成作業的相關資訊

您可以利用 `GetSpeechSynthesisTask` 作業，來取得作業的相關資訊，例如錯誤和狀態等。若要這麼做，您會需要由 `task-id` 所傳回的 `StartSpeechSynthesisTask`。

例如，下列範例可用來執行 `get-speech-synthesis-task` AWS CLI 命令：

```
aws polly get-speech-synthesis-task \  
--region us-east-2 \  
--endpoint-url "https:// polly.us-east-2.amazonaws.com/" \  
--task-id task identifier
```

您也可以利用 `ListSpeechSynthesisTasks` 操作，來列出在目前區域中已執行所有語音合成作業的清單。

例如，下列範例可用來執行 `list-speech-synthesis-tasks` AWS CLI 命令：

```
aws polly list-speech-synthesis-tasks \  
--region us-east-2 \  
--endpoint-url "https:// polly.us-east-2.amazonaws.com/"
```

Amazon Polly 中的配額

Amazon Polly 透過拒絕過多的請求，將配額套用至客戶流量。在單一區域中，標準語音SynthesizeSpeech請求的預設配額為每秒 80 筆交易 (tps)。AWS 帳戶如果限制沒有增加，而且如果您使用標準語音產生每秒 100 個SynthesizeSpeech請求，則每秒 80 個請求會成功，而 Amazon Polly 會調節每秒 20 個請求。這些請求會傳回 HTTP 狀態為 400 的回應，以及指出的回應標頭ThrottlingException。Amazon Polly 也會根據請求率調節所有操作的流量。

語音合成限制範例

- 合成英文字母前 24 個字母，一次一個字母。如果每個字母的合成花費不到 50 毫秒，且操作限制為 8 tps，則合成 24 個字母至少需要三秒。在此期間，您每秒最多可以合成八個字母。任何進一步的請求都會受到調節。隨著請求持續一小段時間，它們會以序列方式合成而不會重疊。
- 合成 16 個段落的文字。如果每個段落兩秒或更短時間內合成並完全在用戶端上接收，且操作限制為八個並行請求，則合成全部 16 篇文章至少需要四秒。在第一秒，您最多可以啟動八個請求。在並行請求期間，任何啟動新合成的嘗試都會因為並行限制而受到調節。您可以在第一批請求完成後，在前兩秒後合成剩餘的八個段落。

使用 Amazon Polly 時，請記住下列限制。

主題

- [支援的 區域](#)
- [配額和調節速率](#)
- [發音語彙](#)
- [SynthesizeSpeech API 操作](#)
- [SpeechSynthesisTask API 操作](#)
- [語音合成標記語言 \(SSML\)](#)

支援的 區域

如需可使用 Amazon Polly AWS 的區域清單，請參閱 [《》中的 Amazon Polly 端點和配額](#) Amazon Web Services 一般參考。

- 如需支援生成語音的區域，請參閱[生成語音](#)。
- 如需支援長格式語音的區域，請參閱[長格式語音](#)。

- 如需支援神經語音的區域，請參閱 [the section called “功能和區域相容性”](#) 以取得神經 TTS。

配額和調節速率

下表定義每個 Amazon Polly 操作的調節速率。您可以視需要使用 AWS 管理主控台 來請求增加可調整配額的配額。

作業	限制
語彙	
DeleteLexicon	結合以上操作的任何每秒 2 次交易 (tps)。 允許爆增上限 4 tps。
PutLexicon	
GetLexicon	
ListLexicons	
語音	
DescribeVoices	80 tps，爆增限制 100 tps
SynthesizeSpeech	生成式語音：8 tps 長形語音：8 tps，爆量限制為 10 tps 神經語音：8 tps，爆增限制 10 tps 標準語音：80 tps，爆增限制 100 tps
StartSpeechSynthesisTask	生成語音：1 tps 長形語音：1 tps 神經語音：10 tps 標準語音：10 tps，爆增限制 12 tps
StartSpeechSynthesisStream	生成式語音：8 tps

作業	限制
GetSynthesizeSpeechTask 和 ListSynthesizeSpeechTask	允許的爆增合併上限為 10 tps

並行請求

對於生成語音，Amazon Polly 最多支援 26 個並行請求。對於長格式語音，Amazon Polly 最多支援 26 個並行請求。對於神經語音，Amazon Polly 支援 8 tps，爆量限制為 10 tps，最多 18 個並行請求。Amazon Polly 也支援並行請求的限制。對於標準語音，Amazon Polly 支援最多 80 個並行請求的 80 tps。

對於 StartSpeechSynthesisStream，Amazon Polly 最多支援 8 個並行請求。

緩解限流的最佳實務

- 重試具有退避和抖動的調節，以便您可以在短時間內分散負載，並在不影響可用性的情況下處理非預期的用量峰值。AWS 程式碼範例目錄 已設定為在許多程式設計語言中執行此操作。造訪 [功能重試行為](#) 以查看詳細資訊。
- 使用 [Amazon Polly 指標](#)。Amazon Polly 會自動發佈至 CloudWatch，以分析您目前的用量並預測用量成長。

Note

在請求提高配額（如適用）之前，請遵循此頁面上的準則來計算您的 tps 需求。Amazon Polly 只會根據客戶需求保護必要的運算資源，以降低您的成本。

發音語彙

- 您可以為每個帳戶儲存最多 100 個語彙。
- 語彙名稱可以是英數字元字串，最長 20 個字元。
- 每個語彙的大小上限為 40,000 個字元。（注意，語彙的大小影響 SynthesizeSpeech 操作的延遲）。
- 您可以為語彙中的每個 <phoneme> 或 <alias> 替換指定多達 100 個字元。

如需使用語彙的資訊，請參閱[管理語彙](#)。

SynthesizeSpeech API 操作

估算 的用量時SynthesizeSpeech，請記住 Amazon Polly 產生的音訊，特別是互動式應用程式，播放通常至少需要幾秒鐘。這可降低對 的請求率SynthesizeSpeech，即使是大量並行消費者也是如此。此外，Amazon Polly 會依其合成的並行SynthesizeSpeech請求數量來調節請求。並行請求沒有單獨的設定。並行請求限制始終與允許的 tps 數量具有相同的值，並隨其擴展。

簡短案例範例應用程式。您可以使用 Amazon Polly 來建置播放一系列簡短故事的應用程式。使用這類應用程式時，第一個案例會開始播放，下一個案例會開始播放，以此類推，直到使用者退出應用程式為止。每個故事大約需要 0.5 秒才能合成，10 秒才能播放。在此案例中，您可能會預期客戶使用應用程式SynthesizeSpeech每 10 秒呼叫一次。這將轉換為同時使用該應用程式的每 10 位客戶每秒一次呼叫。如果您有 1000 位客戶同時使用應用程式，您可以預期的平均通話率為每秒SynthesizeSpeech只有 100 筆交易。

請注意，下列是和使用 SynthesizeSpeech API 作業相關的限制：

- 輸入文字的大小最多可達 3000 個收費字元 (總計 6000 個字元)。SSML 標籤不計為收費字元。
- 您可以指定最多五個語彙以套用到輸入文字。
- 輸出音訊串流 (合成) 限制為 10 分鐘。到達此限制時間之後，任何剩餘的語音都會被截斷。

如需詳細資訊，請參閱[SynthesizeSpeech](#)。

Note

SynthesizeSpeech API 作業的某些限制，可以利用 StartSynthesizeSpeechTask API 作業來略過。如需詳細資訊，請參閱[長音訊檔案](#)。

SpeechSynthesisTask API 操作

請注意，下列是和使用 StartSpeechSynthesisTask、GetSpeechSynthesisTask 及 ListSpeechSynthesisTasks API 作業相關的限制：

- 輸入文字的大小最多可達 100,000 個收費字元 (總計 200,000 個字元)。SSML 標籤不計為收費字元。

- 您可以指定最多五個語彙以套用到輸入文字。

語音合成標記語言 (SSML)

請注意，以下限制有關使用 SSML：

- 不支援 `<audio>`、`<lexicon>`、`<lookup>` 和 `<voice>` 標記。
- `<break>` 元素可以每一個指定最大持續時間 10 秒。
- `<prosody>` 標記不支援速率屬性低於 -80% 的值。

如需詳細資訊，請參閱[從 SSML 文件產生語音](#)。

Amazon Polly 的範例程式碼和應用程式

本節提供程式碼範例和範例應用程式，您可以用來探索 Amazon Polly。

範例程式碼主題包含依程式設計語言整理的程式碼片段，並分成不同 Amazon Polly 功能的範例。範例應用程式主題包含依程式設計語言組織的應用程式，可獨立用於探索 Amazon Polly。

開始使用這些範例之前，我們建議您先閱讀 [Amazon Polly 的運作方式](#) 並依照 [Amazon Polly 入門](#) 所述的步驟進行。

主題

- [Java 範例](#)
- [Python 範例](#)
- [Java 範例](#)
- [Python 範例 \(HTML5 用戶端和 Python 伺服器 \)](#)
- [iOS 範例](#)
- [Android 範例](#)

Java 範例

下列程式碼範例示範如何使用 Java 型應用程式，透過 Amazon Polly 完成各種任務。這些範例並非完整範例，但可以包含在使用的大型 Java 應用程式中[適用於 Java 的 AWS SDK](#)。

程式碼片段

- [DeleteLexicon](#)
- [DescribeVoices](#)
- [GetLexicon](#)
- [ListLexicons](#)
- [PutLexicon](#)
- [StartSpeechSynthesisTask](#)
- [語音標記](#)
- [SynthesizeSpeech](#)

DeleteLexicon

下列 Java 程式碼範例示範如何使用 Java 型應用程式來刪除存放在 AWS 區域中的特定語彙。已刪除的語彙不可用於語音合成，也不可使用 GetLexicon 或 ListLexicon API 擷取。

如需此操作的詳細資訊，請參閱 [DeleteLexicon](#) API 的參考文章。

SDK v2

```
/*
   Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
   SPDX-License-Identifier: Apache-2.0
 */

package com.example.polly;

import software.amazon.awssdk.auth.credentials.ProfileCredentialsProvider;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.polly.PollyClient;
import software.amazon.awssdk.services.polly.model.DeleteLexiconRequest;
import software.amazon.awssdk.services.polly.model.DeleteLexiconResponse;
import software.amazon.awssdk.services.polly.model.PollyException ;

/**
 * Before running this Java V2 code example, set up your development environment,
 * including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class DeleteLexiconSample {

    public static void main(String args[]) {

        PollyClient polly = PollyClient.builder()
            .region(Region.US_WEST_2)
            .credentialsProvider(ProfileCredentialsProvider.create())
            .build();

        deleteLexicon(polly) ;
        polly.close();
    }
}
```

```
private static String LEXICON_NAME = "SampleLexicon";
public static void deleteLexicon(PollyClient client) {

    try {
        DeleteLexiconRequest deleteLexiconRequest = DeleteLexiconRequest.builder()
            .name(LEXICON_NAME).build();

        DeleteLexiconResponse deleteLexiconResult =
client.deleteLexicon(deleteLexiconRequest);

    } catch (PollyException e) {
        System.err.println("Exception caught: " + e);
        System.exit(1);
    }
}
```

SDK v1

```
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

以下 Java 程式碼範例展示如何使用以 Java 為基礎的應用程式，來產生聲音清單。這些聲音可在要求語音合成時使用。您可以選擇指定語言代碼來篩選可用的聲音。例如，如果您指定 en-US，此次操作就會傳回所有可用美式英文聲音的清單。

如需此操作的詳細資訊，請參閱 [DescribeVoices](#) API 的參考文章。

```
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

下列 Java 程式碼範例示範如何使用 Java 型應用程式來產生存放在 AWS 區域中的特定發音語彙內容。

如需此操作的詳細資訊，請參閱 [GetLexicon](#) API 的參考文章。

SDK v2

```

/*
   Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
   SPDX-License-Identifier: Apache-2.0
 */

package com.example.polly;

import software.amazon.awssdk.auth.credentials.ProfileCredentialsProvider;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.polly.PollyClient;
import software.amazon.awssdk.services.polly.model.GetLexiconRequest;
import software.amazon.awssdk.services.polly.model.GetLexiconResponse;
import software.amazon.awssdk.services.polly.model.PollyException ;

/**
 * Before running this Java V2 code example, set up your development environment,
 * including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class GetLexiconSample {

    public static void main(String args[]) {

        PollyClient polly = PollyClient.builder()
            .region(Region.US_WEST_2)
            .credentialsProvider(ProfileCredentialsProvider.create())

```

```
        .build();

        getLexicon(polly) ;
        polly.close();
    }

    private static String LEXICON_NAME = "SampleLexicon";
    public static void getLexicon(PollyClient client) {

        try {
            GetLexiconRequest getLexiconRequest = GetLexiconRequest.builder()
                .name(LEXICON_NAME).build();

            GetLexiconResponse getLexiconResult = client.getLexicon(getLexiconRequest);
            System.out.println("The name of the Lexicon is " +
getLexiconResult.lexicon().name());

        } catch (PollyException e) {
            System.err.println("Exception caught: " + e);
            System.exit(1);
        }
    }
}
```

SDK v1

```
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

下列 Java 程式碼範例示範如何使用 Java 型應用程式來產生儲存在 AWS 區域中的發音語彙清單。

如需此操作的詳細資訊，請參閱 [ListLexicons](#) API 的參考文章。

```
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())
                }
            } while (nextToken != null);
        } catch (Exception e) {
            System.err.println("Exception caught: " + e);
        }
    }
}
```

```

        + ", 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

下列 Java 程式碼範例示範如何使用 Java 型應用程式在 AWS 區域中存放發音語彙。

如需此操作的詳細資訊，請參閱 [PutLexicon](#) API 的參考文章。

SDK v2

```

/*
   Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
   SPDX-License-Identifier: Apache-2.0
 */

package com.example.polly;

import software.amazon.awssdk.auth.credentials.ProfileCredentialsProvider;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.polly.PollyClient;
import software.amazon.awssdk.services.polly.model.PutLexiconRequest;
import software.amazon.awssdk.services.polly.model.PutLexiconResponse;
import software.amazon.awssdk.services.polly.model.PollyException ;

/**
 * Before running this Java V2 code example, set up your development environment,
 * including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */

```

```

public class PutLexiconSample {

    public static void main(String args[]) {

        PollyClient polly = PollyClient.builder()
            .region(Region.US_WEST_2)
            .credentialsProvider(ProfileCredentialsProvider.create())
            .build();

        putLexicon(polly) ;
        polly.close();
    }

    private static String LEXICON_CONTENT = "<?xml version=\"1.0\" encoding=\"UTF-8\"?>\" +
        "<lexicon version=\"1.0\" xmlns=\"http://www.w3.org/2005/01/pronunciation-lexicon\" xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\" \" \" +
        \"xsi:schemaLocation=\"http://www.w3.org/2005/01/pronunciation-lexicon http://www.w3.org/TR/2007/CR-pronunciation-lexicon-20071212/pls.xsd\" \" \" +
        \"alphabet=\"ipa\" xml:lang=\"en-US\">\" +
        "<lexeme><grapheme>test1</grapheme><alias>test2</alias></lexeme>\" +
        "</lexicon>";
    private static String LEXICON_NAME = "SampleLexicon";
    public static void putLexicon(PollyClient client) {

        try {
            PutLexiconRequest putLexiconRequest = PutLexiconRequest.builder()
                .name(LEXICON_NAME).content(LEXICON_CONTENT).build();

            PutLexiconResponse putLexiconResult = client.putLexicon(putLexiconRequest);

        } catch (PollyException e) {
            System.err.println("Exception caught: " + e);
            System.exit(1);
        }
    }
}

```

SDK v1

```
package com.amazonaws.polly.samples;
```

```

import com.amazonaws.services.polly.AmazonPolly;
import com.amazonaws.services.polly.AmazonPollyClientBuilder;
import com.amazonaws.services.polly.model.PutLexiconRequest;

public class PutLexiconSample {
    AmazonPolly client = AmazonPollyClientBuilder.defaultClient();

    private String LEXICON_CONTENT = "<?xml version=\"1.0\" encoding=\"UTF-8\"?>" +
        "<lexicon version=\"1.0\" xmlns=\"http://www.w3.org/2005/01/pronunciation-lexicon\" xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\" " +
        "xsi:schemaLocation=\"http://www.w3.org/2005/01/pronunciation-lexicon http://www.w3.org/TR/2007/CR-pronunciation-lexicon-20071212/pls.xsd\" " +
        "alphabet=\"ipa\" xml:lang=\"en-US\">" +
        "<lexeme><grapheme>test1</grapheme><alias>test2</alias></lexeme>" +
        "</lexicon>";
    private String LEXICON_NAME = "SampleLexicon";

    public void putLexicon() {
        PutLexiconRequest putLexiconRequest = new PutLexiconRequest()
            .withContent(LEXICON_CONTENT)
            .withName(LEXICON_NAME);

        try {
            client.putLexicon(putLexiconRequest);
        } catch (Exception e) {
            System.err.println("Exception caught: " + e);
        }
    }
}

```

StartSpeechSynthesisTask

下列 Java 程式碼範例示範如何使用 Java 應用程式合成語音（最多 100,000 個計費字元），並將其直接存放在 Amazon S3 儲存貯體中。

如需詳細資訊，請參閱 [StartSpeechSynthesisTask](#) API 的參考文章。

SDK v2

```

/*
    Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
    SPDX-License-Identifier: Apache-2.0
*/

```

```
package com.example.polly;

import software.amazon.awssdk.auth.credentials.ProfileCredentialsProvider;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.polly.PollyClient;
import software.amazon.awssdk.services.polly.model.*;

import java.time.Duration;
import org.awaitility.Durations;
import org.awaitility.Awaitility;

/**
 * Before running this Java V2 code example, set up your development environment,
 * including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class StartSpeechSynthesisTaskSample {

    public static void main(String args[]) {

        PollyClient polly = PollyClient.builder()
            .region(Region.US_WEST_2)
            .credentialsProvider(ProfileCredentialsProvider.create())
            .build();

        startSpeechSynthesisTask(polly) ;
        polly.close();
    }

    private static final String PLAIN_TEXT = "This is a sample text to be
synthesized.";
    private static final String OUTPUT_FORMAT_MP3 = OutputFormat.MP3.toString();
    private static final String OUTPUT_BUCKET = "synth-books-buckets";
    private static final String SNS_TOPIC_ARN = "arn:aws:sns:eu-
west-2:123456789012:synthesize-finish-topic";
    private static final Duration SYNTHESIS_TASK_POLL_INTERVAL =
Durations.FIVE_SECONDS;
    private static final Duration SYNTHESIS_TASK_POLL_DELAY = Durations.TEN_SECONDS;
    private static final Duration SYNTHESIS_TASK_TIMEOUT = Durations.FIVE_MINUTES;
    public static void startSpeechSynthesisTask(PollyClient client) {
```

```

        try {
            StartSpeechSynthesisTaskRequest startSpeechSynthesisTaskRequest =
StartSpeechSynthesisTaskRequest.builder()

.outputFormat(OUTPUT_FORMAT_MP3).text(PLAIN_TEXT).textType(TextType.TEXT)

.voiceId(VoiceId.AMY).outputS3BucketName(OUTPUT_BUCKET).snsTopicArn(SNS_TOPIC_ARN)
                .engine("neural").build();

            StartSpeechSynthesisTaskResponse startSpeechSynthesisTaskResponse =
                client.startSpeechSynthesisTask(startSpeechSynthesisTaskRequest);
            String taskId = startSpeechSynthesisTaskResponse.synthesisTask().taskId();

            Awaitility.await().with()
                .pollInterval(SYNTHESIS_TASK_POLL_INTERVAL)
                .pollDelay(SYNTHESIS_TASK_POLL_DELAY)
                .atMost(SYNTHESIS_TASK_TIMEOUT)
                .until(
                    () -> getSynthesisTaskStatus(client,
taskId).equals(TaskStatus.COMPLETED.toString())
                );

        } catch (PollyException e) {
            System.err.println("Exception caught: " + e);
            System.exit(1);
        }
    }

    private static String getSynthesisTaskStatus(PollyClient client, String taskId) {
        GetSpeechSynthesisTaskRequest getSpeechSynthesisTaskRequest =
GetSpeechSynthesisTaskRequest.builder()
                .taskId(taskId).build();

        GetSpeechSynthesisTaskResponse result =
client.getSpeechSynthesisTask(getSpeechSynthesisTaskRequest);
        return result.synthesisTask().taskStatusAsString();
    }
}

```

SDK v1

```
package com.amazonaws.parrot.service.tests.speech.task;
```

```
import com.amazonaws.parrot.service.tests.AbstractParrotServiceTest;
import com.amazonaws.services.polly.AmazonPolly;
import com.amazonaws.services.polly.model.*;
import org.awaitility.Duration;

import java.util.concurrent.TimeUnit;

import static org.awaitility.Awaitility.await;

public class StartSpeechSynthesisTaskSample {

    private static final int SYNTHESIS_TASK_TIMEOUT_SECONDS = 300;
    private static final AmazonPolly AMAZON_POLLY_CLIENT =
AmazonPollyClientBuilder.defaultClient();
    private static final String PLAIN_TEXT = "This is a sample text to be
synthesized.";
    private static final String OUTPUT_FORMAT_MP3 = OutputFormat.Mp3.toString();
    private static final String OUTPUT_BUCKET = "synth-books-buckets";
    private static final String SNS_TOPIC_ARN = "arn:aws:sns:eu-
west-2:123456789012:synthesize-finish-topic";
    private static final Duration SYNTHESIS_TASK_POLL_INTERVAL = Duration.FIVE_SECONDS;
    private static final Duration SYNTHESIS_TASK_POLL_DELAY = Duration.TEN_SECONDS;

    public static void main(String... args) {
        StartSpeechSynthesisTaskRequest request = new StartSpeechSynthesisTaskRequest()
            .withOutputFormat(OUTPUT_FORMAT_MP3)
            .withText(PLAIN_TEXT)
            .withTextType(TextType.Text)
            .withVoiceId(VoiceId.Amy)
            .withOutputS3BucketName(OUTPUT_BUCKET)
            .withSnsTopicArn(SNS_TOPIC_ARN)
            .withEngine("neural");

        StartSpeechSynthesisTaskResult result =
AMAZON_POLLY_CLIENT.startSpeechSynthesisTask(request);
        String taskId = result.getSynthesisTask().getTaskId();

        await().with()
            .pollInterval(SYNTHESIS_TASK_POLL_INTERVAL)
            .pollDelay(SYNTHESIS_TASK_POLL_DELAY)
            .atMost(SYNTHESIS_TASK_TIMEOUT_SECONDS, TimeUnit.SECONDS)
            .until(
```

```

        () ->
getSynthesisTaskStatus(taskId).equals(TaskStatus.Completed.toString())
    );
}

private static SynthesisTask getSynthesisTask(String taskId) {
    GetSpeechSynthesisTaskRequest getSpeechSynthesisTaskRequest = new
GetSpeechSynthesisTaskRequest()
        .withTaskId(taskId);
    GetSpeechSynthesisTaskResult result
=AMAZON_POLLY_CLIENT.getSpeechSynthesisTask(getSpeechSynthesisTaskRequest);
    return result.getSynthesisTask();
}

private static String getSynthesisTaskStatus(String taskId) {
    GetSpeechSynthesisTaskRequest getSpeechSynthesisTaskRequest = new
GetSpeechSynthesisTaskRequest()
        .withTaskId(taskId);
    GetSpeechSynthesisTaskResult result
=AMAZON_POLLY_CLIENT.getSpeechSynthesisTask(getSpeechSynthesisTaskRequest);
    return result.getSynthesisTask().getTaskStatus();
}
}

```

語音標記

以下 Java 程式碼範例展示如何使用以 Java 為基礎的應用程式，來合成輸入文字的語音標記。此功能使用 SynthesizeSpeech API。

如需該功能的詳細資訊，請參閱[語音標記](#)。

如需 API 詳細資訊，請參閱 [SynthesizeSpeech](#) API 的參考文章。

SDK v2

```

/*
    Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
    SPDX-License-Identifier: Apache-2.0
*/

package com.example.polly;

```

```
import software.amazon.awssdk.auth.credentials.ProfileCredentialsProvider;
import software.amazon.awssdk.core.ResponseInputStream;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.polly.PollyClient;
import software.amazon.awssdk.services.polly.model.*;

import java.io.File;
import java.io.FileOutputStream;
import java.io.InputStream;

/**
 * Before running this Java V2 code example, set up your development environment,
 * including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class SpeechMarksSample {

    public static void main(String args[]) {

        PollyClient polly = PollyClient.builder()
            .region(Region.US_WEST_2)
            .credentialsProvider(ProfileCredentialsProvider.create())
            .build();

        speechMarksSample(polly) ;
        polly.close();
    }

    private static final String OUTPUT_FILE = "./speechMarks.json";
    public static void speechMarksSample(PollyClient client) {

        try {
            SynthesizeSpeechRequest speechMarksSampleRequest =
            SynthesizeSpeechRequest.builder()
                .outputFormat(OutputFormat.JSON)
                .speechMarkTypes(SpeechMarkType.VISEME, SpeechMarkType.WORD)
                .voiceId(VoiceId.JOANNA)
                .text("This is a sample text to be synthesized")
                .build();

            try (FileOutputStream outputStream = new FileOutputStream(new
            File(OUTPUT_FILE))) {
```

```

        ResponseInputStream<SynthesizeSpeechResponse> synthesizeSpeechResponse
= client

        .synthesizeSpeech(speechMarksSampleRequest);
        byte[] buffer = new byte[2 * 1024];
        int readBytes;

        try (InputStream in = synthesizeSpeechResponse){
            while ((readBytes = in.read(buffer)) > 0) {
                outputStream.write(buffer, 0, readBytes);
            }
        }
        } catch (Exception e) {
            System.err.println("Exception caught: " + e);
        }

        } catch (PollyException e) {
            System.err.println("Exception caught: " + e);
            System.exit(1);
        }
    }
}

```

SDK v1

```

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

下列的 Java 程式碼範例示範如何使用以 Java 撰寫的應用程式，來合成較短篇的語音，實現接近即時的處理速度。

如需詳細資訊，請參閱 [SynthesizeSpeech](#) API 的參考文章。

```

package com.amazonaws.polly.samples;

import com.amazonaws.services.polly.AmazonPolly;
import com.amazonaws.services.polly.AmazonPollyClientBuilder;
import com.amazonaws.services.polly.model.OutputFormat;
import com.amazonaws.services.polly.model.SynthesizeSpeechRequest;
import com.amazonaws.services.polly.model.SynthesizeSpeechResult;
import com.amazonaws.services.polly.model.VoiceId;

import java.io.File;

```

```
import java.io.FileOutputStream;
import java.io.InputStream;

public class SynthesizeSpeechSample {
    AmazonPolly client = AmazonPollyClientBuilder.defaultClient();

    public void synthesizeSpeech() {
        String outputFileName = "/tmp/speech.mp3";

        SynthesizeSpeechRequest synthesizeSpeechRequest = new SynthesizeSpeechRequest()
            .withOutputFormat(OutputFormat.Mp3)
            .withVoiceId(VoiceId.Joanna)
            .withText("This is a sample text to be synthesized.")
            .withEngine("neural");

        try (FileOutputStream outputStream = new FileOutputStream(new
File(outputFileName))) {
            SynthesizeSpeechResult synthesizeSpeechResult =
client.synthesizeSpeech(synthesizeSpeechRequest);
            byte[] buffer = new byte[2 * 1024];
            int readBytes;

            try (InputStream in = synthesizeSpeechResult.getAudioStream()){
                while ((readBytes = in.read(buffer)) > 0) {
                    outputStream.write(buffer, 0, readBytes);
                }
            }
        } catch (Exception e) {
            System.err.println("Exception caught: " + e);
        }
    }
}
```

Python 範例

下列程式碼範例示範如何使用 Python (boto3) 型應用程式，透過 Amazon Polly 完成各種任務。這些範例並非預定做為完整的範例，但可包括在使用 [適用於 Python \(Boto\) 的 AWS SDK](#) 的更大型 Python 應用程式中。

程式碼片段

- [DeleteLexicon](#)

- [GetLexicon](#)
- [ListLexicon](#)
- [PutLexicon](#)
- [StartSpeechSynthesisTask](#)
- [SynthesizeSpeech](#)

DeleteLexicon

下列 Python 程式碼範例使用 在本機 AWS 組態中指定的區域中 適用於 Python (Boto) 的 AWS SDK 刪除語彙。此範例僅刪除指定的語彙。此範例會在真正刪除語彙前詢問您，以確認您要繼續進行。

下列程式碼範例使用存放在 AWS SDK 組態檔案中的預設登入資料。如需建立組態檔的資訊，請參閱「[設定 AWS CLI](#)」。

如需此操作的詳細資訊，請參閱 [DeleteLexicon](#) API 的參考文章。

```
from argparse import ArgumentParser
from sys import version_info

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

# Define and parse the command line arguments
cli = ArgumentParser(description="DeleteLexicon example")
cli.add_argument("name", type=str, metavar="LEXICON_NAME")
arguments = cli.parse_args()

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

# Request confirmation
prompt = input if version_info >= (3, 0) else raw_input
proceed = prompt((u"This will delete the \"{0}\" lexicon,"
                  " do you want to proceed? [y,n]: ").format(arguments.name))

if proceed in ("y", "Y"):
    print(u"Deleting {0}...".format(arguments.name))
```

```
try:
    # Request deletion of a lexicon by name
    response = polly.delete_lexicon(Name=arguments.name)
except (BotoCoreError, ClientError) as error:
    # The service returned an error, exit gracefully
    cli.error(error)

print("Done.")
else:
    print("Cancelled.")
```

GetLexicon

下列 Python 程式碼使用適用於 Python (Boto) 的 AWS SDK 來擷取存放在 AWS 區域的所有語彙。此範例接受作為命令行參數的語彙名稱，並僅擷取該語彙，列印本機儲存的 tmp 路徑。

下列程式碼範例使用存放在 AWS SDK 組態檔案中的預設登入資料。如需建立組態檔的資訊，請參閱「[設定 AWS CLI](#)」。

如需此操作的詳細資訊，請參閱 [GetLexicon](#) API 的參考文章。

```
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

下列 Python 程式碼範例使用 在本機 AWS 組態中指定的區域中 適用於 Python (Boto) 的 AWS SDK ，列出您帳戶中的語彙。如需建立組態檔的資訊，請參閱「[設定 AWS CLI](#)」。

如需此操作的詳細資訊，請參閱 [ListLexicons](#) API 的參考文章。

```

import sys

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

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

try:

```

```
# Request the list of available lexicons
response = polly.list_lexicons()
except (BotoCoreError, ClientError) as error:
    # The service returned an error, exit gracefully
    print(error)
    sys.exit(-1)

# Get the list of lexicons in the response
lexicons = response.get("Lexicons", [])
print("{0} lexicon(s) found".format(len(lexicons)))

# Output a formatted list of lexicons with some of the attributes
for lexicon in lexicons:
    print((u" - {Name} ({Attributes[LanguageCode]}), "
          "{Attributes[LexemesCount]} lexeme(s)").format(**lexicon))
```

PutLexicon

下列程式碼範例示範如何使用 Python (boto3) 型應用程式在 AWS 區域中存放發音語彙。

如需此操作的詳細資訊，請參閱 [PutLexicon](#) API 的參考文章。

注意下列事項：

- 您需要提供本機語彙檔名和儲存的語彙名稱，以更新程式碼。
- 此範例假設您在 pls 子目錄中建立語彙檔案。您需要適時更新路徑。

下列程式碼範例使用存放在 AWS SDK 組態檔案中的預設登入資料。如需建立組態檔的資訊，請參閱「[設定 AWS CLI](#)」。

如需此操作的詳細資訊，請參閱 [PutLexicon](#) API 的參考文章。

```
from argparse import ArgumentParser

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

# Define and parse the command line arguments
cli = ArgumentParser(description="PutLexicon example")
cli.add_argument("path", type=str, metavar="FILE_PATH")
cli.add_argument("-n", "--name", type=str, required=True,
```

```

        metavar="LEXICON_NAME", dest="name")
arguments = cli.parse_args()

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

# Open the PLS lexicon file for reading
try:
    with open(arguments.path, "r") as lexicon_file:
        # Read the pls file contents
        lexicon_data = lexicon_file.read()

        # Store the PLS lexicon on the service.
        # If a lexicon with that name already exists,
        # its contents will be updated
        response = polly.put_lexicon(Name=arguments.name,
                                     Content=lexicon_data)
except (IOError, BotoCoreError, ClientError) as error:
    # Could not open/read the file or the service returned an error,
    # exit gracefully
    cli.error(error)

print(u"The \"{0}\" lexicon is now available for use.".format(arguments.name))

```

StartSpeechSynthesisTask

下列 Python 程式碼範例使用 在本機 AWS 組態中指定的區域中 適用於 Python (Boto) 的 AWS SDK ，列出您帳戶中的語彙。如需建立組態檔的資訊，請參閱「[設定 AWS CLI](#)」。

如需詳細資訊，請參閱 [StartSpeechSynthesisTask](#) API 的參考文章。

```

import boto3
import time

polly_client = boto3.Session(
    aws_access_key_id='',
    aws_secret_access_key='',
    region_name='eu-west-2').client('polly')

response = polly_client.start_speech_synthesis_task(VoiceId='Joanna',

```

```
        OutputS3BucketName='synth-books-buckets',
        OutputS3KeyPrefix='key',
        OutputFormat='mp3',
        Text='This is a sample text to be synthesized.',
        Engine='neural')

taskId = response['SynthesisTask']['TaskId']

print( "Task id is {} ".format(taskId))

task_status = polly_client.get_speech_synthesis_task(TaskId = taskId)

print(task_status)
```

SynthesizeSpeech

下列 Python 程式碼範例使用具有較短文字的 適用於 Python (Boto) 的 AWS SDK 合成語音，以進行近乎即時的處理。如需詳細資訊，請參閱 [SynthesizeSpeech](#) 操作的參考。

此範例使用短字串的純文字。您可以使用 SSML 文字來進一步控制輸出。如需詳細資訊，請參閱[從 SSML 文件產生語音](#)。

```
import boto3

polly_client = boto3.Session(
    aws_access_key_id=,
    aws_secret_access_key=,
    region_name='us-west-2').client('polly')

response = polly_client.synthesize_speech(VoiceId='Joanna',
    OutputFormat='mp3',
    Text = 'This is a sample text to be synthesized.',
    Engine = 'neural')

file = open('speech.mp3', 'wb')
file.write(response['AudioStream'].read())
file.close()
```

Java 範例

此範例示範如何使用 Amazon Polly 從以 Java 為基礎的應用程式串流語音。此範例使用 [適用於 Java 的 AWS SDK](#)，使用從清單中選取的語音來讀取指定的文字。

顯示的程式碼包括主要工作，但僅進行最基本的錯誤檢查。如果 Amazon Polly 發生錯誤，應用程式會終止。

若要執行此範例應用程式，您需要以下項目：

- Java 8 Java Development Kit (JDK)
- [適用於 Java 的 AWS SDK](#)
- [Apache Maven](#)

若要測試應用程式

1. 確認 JAVA_HOME 環境變數已針對 JDK 設定。

例如，如果您在 Windows 中將 JDK 1.8.0_121 安裝於 C:\Program Files\Java\jdk1.8.0_121，您可在命令提示輸入以下命令：

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

如果您在 Linux 中將 JDK 1.8.0_121 安裝於 /usr/lib/jvm/java8-openjdk-amd64，您可在命令提示輸入以下命令：

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

2. 設定 Maven 環境變數，從命令列執行 Maven。

例如，如果您在 Windows 中將 Maven 3.3.9 安裝於 C:\Program Files\apache-maven-3.3.9，您可以輸入以下命令：

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

如果您在 Linux 中將 Maven 3.3.9 安裝於 /home/ec2-user/opt/apache-maven-3.3.9，您可以輸入以下命令：

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

3. 建立名為 `polly-java-demo` 的新目錄。
4. 在 `polly-java-demo` 目錄中，建立名為 `pom.xml` 的新檔案，並將以下程式碼貼入其中：

```
<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>
```

```
</plugin>
</plugins>
</build>
</project>
```

5. 在 `src/main/java/com/amazonaws/demos` 建立名為 `polly` 的新目錄。
6. 在 `polly` 目錄中，建立名為 `PollyDemo.java` 的新 Java 來源檔案，並在以下程式碼貼中上：

```
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 javax.swing.plaf.basic.AdvancedPlayer;
import javax.swing.plaf.basic.PlaybackEvent;
import javax.swing.plaf.basic.PlaybackListener;

public class PollyDemo {

    private final AmazonPollyClient polly;
    private final Voice voice;
    private static final String SAMPLE = "Congratulations. You have successfully built this working demo of Amazon Polly in Java. Have fun building voice enabled apps with Amazon Polly (that's me!), and always look at the AWS website for tips and tricks on using Amazon Polly and other great services from AWS";

    public PollyDemo(Region region) {
        // create an Amazon Polly client in a specific region
```

```
polly = new AmazonPollyClient(new DefaultAWSCredentialsProviderChain(),
    new ClientConfiguration());
polly.setRegion(region);
// Create describe voices request.
DescribeVoicesRequest describeVoicesRequest = new DescribeVoicesRequest();

// Synchronously ask Amazon Polly to describe available TTS voices.
DescribeVoicesResult describeVoicesResult =
polly.describeVoices(describeVoicesRequest);
voice = describeVoicesResult.getVoices().get(0);
}

public InputStream synthesize(String text, OutputFormat format) throws IOException
{
    SynthesizeSpeechRequest synthReq =
    new SynthesizeSpeechRequest().withText(text).withVoiceId(voice.getId())
        .withOutputFormat(format).withEngine("neural");
    SynthesizeSpeechResult synthRes = polly.synthesizeSpeech(synthReq);

    return synthRes.getAudioStream();
}

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

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

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

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

```
// play it!  
player.play();  
  
}  
}
```

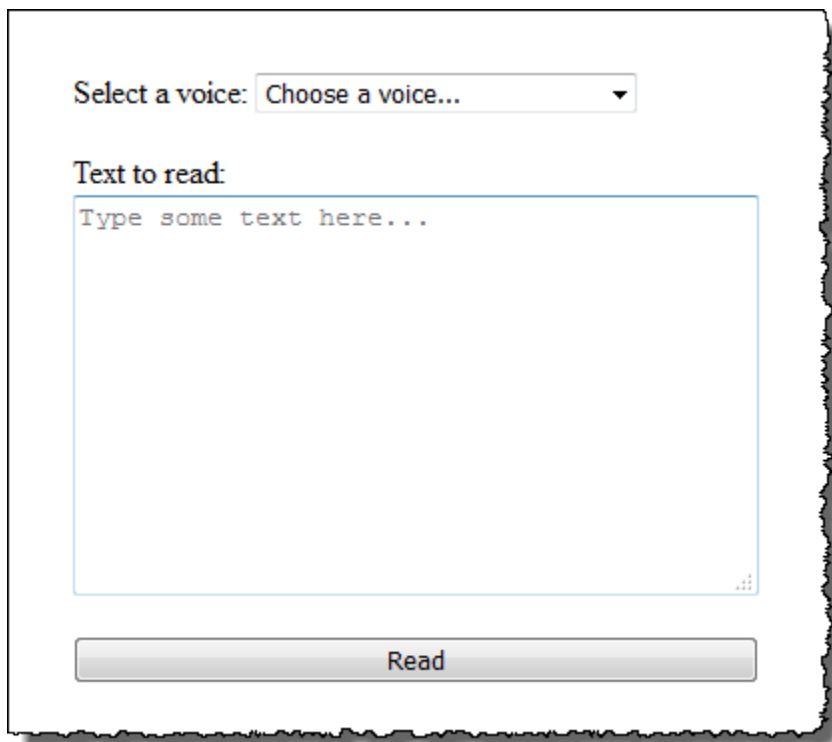
7. 回到 polly-java-demo 目錄清理、編譯和執行示範：

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

Python 範例 (HTML5 用戶端和 Python 伺服器)

此範例應用程式包含以下項目：

- 使用 HTTP 區塊傳輸編碼的 HTTP 1.1 伺服器 (請參閱 [區塊傳輸編碼](#))
- 與 HTTP 1.1 伺服器互動的簡易 HTML5 使用者介面 (如下所示)：



此範例的目標是示範如何使用 Amazon Polly 從瀏覽器型 HTML5 應用程式串流語音。在合成文字時使用 Amazon Polly 產生的音訊串流，是回應能力是重要因素（例如，對話方塊系統、螢幕閱讀器等）的使用案例建議方法。

若要執行此範例應用程式，您需要以下項目：

- 符合 HTML5 和 EcmaScript5 標準的網頁瀏覽器 (如 Chrome 23.0 或以上、Firefox 21.0 或以上、Internet Explorer 9.0 或以上)
- Python 3.0 以上的版本

若要測試應用程式

1. 將伺服器程式碼另存為 `server.py`。如需程式碼，請參閱 [Python 範例：Python 伺服器程式碼 \(server.py : //\)](#)。
2. 將 HTML5 用戶端程式碼另存為 `index.html`。如需程式碼，請參閱 [Python 範例：HTML5 使用者介面 \(index.html\)](#)。
3. 從您儲存 `server.py` 的路徑執行以下命令，以啟動應用程式 (在某些系統中，您可能需要在執行命令時使用 `python3`，而非 `python`)。

```
$ python server.py
```

應用程式啟動後，URL 在終端機上出現。

4. 在網頁瀏覽器中開啟在終端機中顯示的 URL。

您可以將應用程式伺服器的位址和連接埠 (以作為參數使用) 傳遞至 `server.py`。如需詳細資訊，請執行 `python server.py -h`。

5. 若要聆聽語音，請從清單選擇聲音、輸入一些文字，然後選擇 [Read]。一旦 Amazon Polly 傳輸第一個可用的音訊資料區塊，語音就會開始播放。
6. 若要在完成測試應用程式時停止 Python 伺服器，請在伺服器正在執行的終端機中按下 Ctrl+C。

Note

伺服器使用適用於 Python (Boto) 的 AWS SDK 建立 Boto3 用戶端。用戶端會使用存放在您電腦上 AWS 組態檔案中的登入資料來簽署和驗證對 Amazon Polly 的請求。如需如何建立 AWS

組態檔案和存放登入資料的詳細資訊，請參閱AWS Command Line Interface 《使用者指南》中的[設定 AWS Command Line Interface](#)。

Python 範例：HTML5 使用者介面 (index.html)

本節提供 [Python 範例 \(HTML5 用戶端和 Python 伺服器\)](#) 所述的 HTML5 用戶端程式碼。

```
<html>  
  
<head>  
  <title>Text-to-Speech Example Application</title>  
  <script>  
    /*  
     * This sample code requires a web browser with support for both the  
     * HTML5 and ECMAScript 5 standards; the following is a non-comprehensive  
     * list of compliant browsers and their minimum version:  
     *  
     * - Chrome 23.0+  
     * - Firefox 21.0+  
     * - Internet Explorer 9.0+  
     * - Edge 12.0+  
     * - Opera 15.0+  
     * - Safari 6.1+  
     * - Android (stock web browser) 4.4+  
     * - Chrome for Android 51.0+  
     * - Firefox for Android 48.0+  
     * - Opera Mobile 37.0+  
     * - iOS (Safari Mobile and Chrome) 3.2+  
     * - Internet Explorer Mobile 10.0+  
     * - Blackberry Browser 10.0+  
     */  
  
    // Mapping of the OutputFormat parameter of the SynthesizeSpeech API  
    // and the audio format strings understood by the browser  
    var AUDIO_FORMATS = {  
      'ogg_vorbis': 'audio/ogg',  
      'mp3': 'audio/mpeg',  
      'pcm': 'audio/wave; codecs=1',  
      'mulaw': 'audio/mulaw',  
      'alaw': 'audio/alaw'  
    };
```

```
/**
 * Handles fetching JSON over HTTP
 */
function fetchJSON(method, url, onSuccess, onError) {
    var request = new XMLHttpRequest();
    request.open(method, url, true);
    request.onload = function () {
        // If loading is complete
        if (request.readyState === 4) {
            // if the request was successful
            if (request.status === 200) {
                var data;

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

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

    request.send();
}

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

// Initialize the application when the DOM is loaded and ready to be
// manipulated
document.addEventListener("DOMContentLoaded", function () {
```

```
var input = document.getElementById('input'),
    voiceMenu = document.getElementById('voice'),
    text = document.getElementById('text'),
    player = document.getElementById('player'),
    submit = document.getElementById('submit'),
    supportedFormats = getSupportedAudioFormats(player);

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

```
}

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

```
    } else {
        var selectedVoice = voiceMenu
            .options[voiceMenu.selectedIndex]
            .value;

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

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

// Load the list of available voices and display them in a menu
fetchJSON('GET', '/voices',
    // If the request succeeds
```

```
function (voices) {
    var container = document.createDocumentFragment();

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

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

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

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

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

```

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

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

</html>

```

Python 範例：Python 伺服器程式碼 (server.py：//)

本節提供 [Python 範例 \(HTML5 用戶端和 Python 伺服器\)](#) 所述的 Python 伺服器程式碼。

```

"""
Example Python 2.7+/3.3+ Application

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

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

```

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

To test the application, run 'python server.py' and then open the URL displayed in the terminal in a web browser (see index.html for a list of supported browsers). The address and port for the server can be passed as parameters to server.py. For more information, run: 'python server.py -h'

```

"""
from argparse import ArgumentParser
from collections import namedtuple
from contextlib import closing
from io import BytesIO
from json import dumps as json_encode
import os
import sys

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

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

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

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

# Mapping the output format used in the client to the content type for the
# response
AUDIO_FORMATS = {"ogg_vorbis": "audio/ogg",
                  "mp3": "audio/mpeg",
                  "pcm": "audio/wave; codecs=1",
                  "mulaw": "audio/mulaw",
                  "alaw": "audio/alaw"}

CHUNK_SIZE = 1024
HTTP_STATUS = {"OK": ResponseStatus(code=200, message="OK"),

```

```

        "BAD_REQUEST": ResponseStatus(code=400, message="Bad request"),
        "NOT_FOUND": ResponseStatus(code=404, message="Not found"),
        "INTERNAL_SERVER_ERROR": ResponseStatus(code=500, message="Internal
server error")}]
PROTOCOL = "http"
ROUTE_INDEX = "/index.html"
ROUTE_VOICES = "/voices"
ROUTE_READ = "/read"

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

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

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

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

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

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

```

```

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

    # Extract values from the query string
    path, _, query_string = self.path.partition('?')
    query = parse_qs(query_string)

    response = None

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

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

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

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

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

    print("[END]")

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

def route_index(self, path, query):
    """Handles routing for the application's entry point"""

```

```

try:
    return ResponseData(status=HTTP_STATUS["OK"], content_type="text_html",
                        # Open a binary stream for reading the index
                        # HTML file
                        data_stream=open(os.path.join(sys.path[0],
                                                    path[1:]), "rb"))

except IOError as err:
    # Couldn't open the stream
    raise HTTPStatusError(HTTP_STATUS["INTERNAL_SERVER_ERROR"],
                          str(err))

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

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

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

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

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

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

```

```

        data_stream=BytesIO(bytes_data))

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

    # Validate the parameters, set error flag in case of unexpected
    # values
    if len(text) == 0 or len(voiceId) == 0 or \
        outputFormat not in AUDIO_FORMATS:
        raise HTTPStatusError(HTTP_STATUS["BAD_REQUEST"],
                               "Wrong parameters")
    else:
        try:
            # Request speech synthesis
            response = polly.synthesize_speech(Text=text,
                                                VoiceId=voiceId,
                                                OutputFormat=outputFormat,
                                                Engine="neural")

        except (BotoCoreError, ClientError) as err:
            # The service returned an error
            raise HTTPStatusError(HTTP_STATUS["INTERNAL_SERVER_ERROR"],
                                   str(err))

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

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

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

```

```

if stream:
    # Note: Closing the stream is important as the service throttles on
    # the number of parallel connections. Here we are using
    # contextlib.closing to ensure the close method of the stream object
    # will be called automatically at the end of the with statement's
    # scope.
    with closing(stream) as managed_stream:
        # Push out the stream's content in chunks
        while True:
            data = managed_stream.read(CHUNK_SIZE)
            self.wfile.write(b"%X\r\n%s\r\n" % (len(data), data))

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

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

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

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

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

```

```
ROUTE_INDEX))
```

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

iOS 範例

下列範例使用適用於 Amazon Polly 的 iOS 開發套件，使用從語音清單中選擇的語音來讀取指定的文字。

在此顯示的程式碼涵蓋主要工作，但不會處理錯誤。如需完整的程式碼，請參閱 [AWS Mobile SDK for iOS Amazon Polly 示範](#)。

初始化

```
// 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()
```

取得可用聲音的清單

```
// 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
})
```

合成語音

```
// 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 範例

下列範例使用適用於 Amazon Polly 的 Android 開發套件，使用從語音清單中選擇的語音來讀取指定的文字。

在此顯示的程式碼涵蓋主要工作，但不會處理錯誤。如需完整的程式碼，請參閱 [適用於 Android 的 AWS Mobile SDK Amazon Polly 示範](#)。

初始化

```
// 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);
```

取得可用聲音的清單

```
// 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();
```

取得音訊串流的 URL

```
// 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);
```

播放合成的語音

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

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

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

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

// Set the callback to start the MediaPlayer when it's prepared.
mediaPlayer.setOnPreparedListener(new MediaPlayer.OnPreparedListener() {
    @Override
```

```
        public void onPrepared(MediaPlayer mp) {
            mp.start();
        }
    });

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

Amazon Polly AWS SDKs的程式碼範例

下列程式碼範例示範如何使用 Amazon Polly 搭配 AWS 軟體開發套件 (SDK)。

Actions 是大型程式的程式碼摘錄，必須在內容中執行。雖然動作會告訴您如何呼叫個別服務函數，但您可以在其相關情境中查看內容中的動作。

案例是向您展示如何呼叫服務中的多個函數或與其他 AWS 服務組合來完成特定任務的程式碼範例。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [搭配 SDK 使用 Amazon Polly AWS](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

程式碼範例

- [使用 AWS SDKs Amazon Polly 基本範例](#)
 - [Amazon Polly AWS SDKs的動作](#)
 - [DeleteLexicon 搭配 AWS SDK 或 CLI 使用](#)
 - [DescribeVoices 搭配 AWS SDK 使用](#)
 - [GetLexicon 搭配 AWS SDK 或 CLI 使用](#)
 - [GetSpeechSynthesisTask 搭配 AWS SDK 或 CLI 使用](#)
 - [ListLexicons 搭配 AWS SDK 或 CLI 使用](#)
 - [ListSpeechSynthesisTasks 搭配 AWS SDK 或 CLI 使用](#)
 - [PutLexicon 搭配 AWS SDK 或 CLI 使用](#)
 - [StartSpeechSynthesisTask 搭配 AWS SDK 或 CLI 使用](#)
 - [SynthesizeSpeech 搭配 AWS SDK 使用](#)
 - [Amazon Polly AWS SDKs案例](#)
 - [使用 AWS SDK 將文字轉換為語音並返回文字](#)
 - [使用 SDK 與 Amazon Polly 建立唇同步應用程式 AWS](#)
 - [建立可分析客戶意見回饋並合成音訊的應用程式](#)
 - [text-to-speech合成入門](#)

使用 AWS SDKs Amazon Polly 基本範例

下列程式碼範例示範如何搭配 AWS SDK 使用 Amazon Polly 的基本功能。

範例

- [Amazon Polly AWS SDKs的動作](#)
 - [DeleteLexicon 搭配 AWS SDK 或 CLI 使用](#)
 - [DescribeVoices 搭配 AWS SDK 使用](#)
 - [GetLexicon 搭配 AWS SDK 或 CLI 使用](#)
 - [GetSpeechSynthesisTask 搭配 AWS SDK 或 CLI 使用](#)
 - [ListLexicons 搭配 AWS SDK 或 CLI 使用](#)
 - [ListSpeechSynthesisTasks 搭配 AWS SDK 或 CLI 使用](#)
 - [PutLexicon 搭配 AWS SDK 或 CLI 使用](#)
 - [StartSpeechSynthesisTask 搭配 AWS SDK 或 CLI 使用](#)
 - [SynthesizeSpeech 搭配 AWS SDK 使用](#)

Amazon Polly AWS SDKs的動作

下列程式碼範例示範如何使用 AWS SDKs 執行個別 Amazon Polly 動作。每個範例均包含 GitHub 的連結，您可以在連結中找到設定和執程式碼的相關說明。

這些摘錄會呼叫 Amazon Polly API，是必須在內容中執行之大型程式的程式碼摘錄。您可以在 [Amazon Polly AWS SDKs案例](#) 中查看內容中的動作。

下列範例僅包含最常使用的動作。如需完整清單，請參閱《[Amazon Polly API 參考](#)》。

範例

- [DeleteLexicon 搭配 AWS SDK 或 CLI 使用](#)
- [DescribeVoices 搭配 AWS SDK 使用](#)
- [GetLexicon 搭配 AWS SDK 或 CLI 使用](#)
- [GetSpeechSynthesisTask 搭配 AWS SDK 或 CLI 使用](#)
- [ListLexicons 搭配 AWS SDK 或 CLI 使用](#)
- [ListSpeechSynthesisTasks 搭配 AWS SDK 或 CLI 使用](#)
- [PutLexicon 搭配 AWS SDK 或 CLI 使用](#)
- [StartSpeechSynthesisTask 搭配 AWS SDK 或 CLI 使用](#)
- [SynthesizeSpeech 搭配 AWS SDK 使用](#)

DeleteLexicon 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 DeleteLexicon。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [text-to-speech合成入門](#)

.NET

適用於 .NET 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
using System;
using System.Threading.Tasks;
using Amazon.Polly;
using Amazon.Polly.Model;

/// <summary>
/// Deletes an existing Amazon Polly lexicon using the AWS SDK for .NET.
/// </summary>
public class DeleteLexicon
{
    public static async Task Main()
    {
        string lexiconName = "SampleLexicon";

        var client = new AmazonPollyClient();

        var success = await DeletePollyLexiconAsync(client, lexiconName);

        if (success)
        {
            Console.WriteLine($"Successfully deleted {lexiconName}.");
        }
    }
}
```

```

        else
        {
            Console.WriteLine($"Could not delete {lexiconName}.");
        }
    }

    /// <summary>
    /// Deletes the named Amazon Polly lexicon.
    /// </summary>
    /// <param name="client">The initialized Amazon Polly client object.</
param>
    /// <param name="lexiconName">The name of the Amazon Polly lexicon to
    /// delete.</param>
    /// <returns>A Boolean value indicating the success of the operation.</
returns>
    public static async Task<bool> DeletePollyLexiconAsync(
        AmazonPollyClient client,
        string lexiconName)
    {
        var deleteLexiconRequest = new DeleteLexiconRequest()
        {
            Name = lexiconName,
        };

        var response = await client.DeleteLexiconAsync(deleteLexiconRequest);

        return response.HttpStatusCode == System.Net.HttpStatusCode.OK;
    }
}

```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [DeleteLexicon](#)。

CLI

AWS CLI

刪除語彙

下列 delete-lexicon 範例會刪除指定的語彙。

```
aws polly delete-lexicon \
```

```
--name w3c
```

此命令不會產生輸出。

如需詳細資訊，請參閱《Amazon Polly 開發人員指南》中的[使用 DeleteLexicon 操作](#)。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的[DeleteLexicon](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在[AWS 程式碼範例儲存庫](#)中設定和執行。

```
TRY.  
    lo_ply->delelexicon( iv_name ).  
    MESSAGE 'Lexicon deleted successfully.' TYPE 'I'.  
CATCH /aws1/cx_plylexiconnotfoundex.  
    MESSAGE 'Lexicon not found.' TYPE 'E'.  
CATCH /aws1/cx_plyservicefailureex.  
    MESSAGE 'Service failure occurred.' TYPE 'E'.  
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的[DeleteLexicon](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱[搭配 SDK 使用 Amazon Polly AWS](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

DescribeVoices 搭配 AWS SDK 使用

下列程式碼範例示範如何使用 DescribeVoices。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [text-to-speech合成入門](#)

.NET

適用於 .NET 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
using System;
using System.Threading.Tasks;
using Amazon.Polly;
using Amazon.Polly.Model;

public class DescribeVoices
{
    public static async Task Main()
    {
        var client = new AmazonPollyClient();

        var allVoicesRequest = new DescribeVoicesRequest();
        var enUsVoicesRequest = new DescribeVoicesRequest()
        {
            LanguageCode = "en-US",
        };

        try
        {
            string nextToken;
            do
            {
                var allVoicesResponse = await
client.DescribeVoicesAsync(allVoicesRequest);
                nextToken = allVoicesResponse.NextToken;
                allVoicesRequest.NextToken = nextToken;

                Console.WriteLine("\nAll voices: ");
                allVoicesResponse.Voices.ForEach(voice =>
                {
```

```

        DisplayVoiceInfo(voice);
    });
}
while (nextToken is not null);

do
{
    var enUsVoicesResponse = await
client.DescribeVoicesAsync(enUsVoicesRequest);
    nextToken = enUsVoicesResponse.NextToken;
    enUsVoicesRequest.NextToken = nextToken;

    Console.WriteLine("\nen-US voices: ");
    enUsVoicesResponse.Voices.ForEach(voice =>
    {
        DisplayVoiceInfo(voice);
    });
}
while (nextToken is not null);
}
catch (Exception ex)
{
    Console.WriteLine("Exception caught: " + ex.Message);
}
}

public static void DisplayVoiceInfo(Voice voice)
{
    Console.WriteLine($" Name: {voice.Name}\tGender:
{voice.Gender}\tLanguageName: {voice.LanguageName}");
}
}

```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [DescribeVoices](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.polly.PollyClient;
import software.amazon.awssdk.services.polly.model.DescribeVoicesRequest;
import software.amazon.awssdk.services.polly.model.DescribeVoicesResponse;
import software.amazon.awssdk.services.polly.model.PollyException;
import software.amazon.awssdk.services.polly.model.Voice;
import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
 * started.html
 */
public class DescribeVoicesSample {
    public static void main(String args[]) {
        PollyClient polly = PollyClient.builder()
            .region(Region.US_WEST_2)
            .build();

        describeVoice(polly);
        polly.close();
    }

    public static void describeVoice(PollyClient polly) {
        try {
            DescribeVoicesRequest voicesRequest = DescribeVoicesRequest.builder()
                .languageCode("en-US")
                .build();
```

```

        DescribeVoicesResponse enUsVoicesResult =
polly.describeVoices(voicesRequest);
        List<Voice> voices = enUsVoicesResult.voices();
        for (Voice myVoice : voices) {
            System.out.println("The ID of the voice is " + myVoice.id());
            System.out.println("The gender of the voice is " +
myVoice.gender());
        }

    } catch (PollyException e) {
        System.err.println("Exception caught: " + e);
        System.exit(1);
    }
}
}

```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [DescribeVoices](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

class PollyWrapper:
    """Encapsulates Amazon Polly functions."""

    def __init__(self, polly_client, s3_resource):
        """
        :param polly_client: A Boto3 Amazon Polly client.
        :param s3_resource: A Boto3 Amazon Simple Storage Service (Amazon S3)
resource.
        """
        self.polly_client = polly_client
        self.s3_resource = s3_resource
        self.voice_metadata = None

```

```
def describe_voices(self):
    """
    Gets metadata about available voices.

    :return: The list of voice metadata.
    """
    try:
        response = self.polly_client.describe_voices()
        self.voice_metadata = response["Voices"]
        logger.info("Got metadata about %s voices.",
len(self.voice_metadata))
    except ClientError:
        logger.exception("Couldn't get voice metadata.")
        raise
    else:
        return self.voice_metadata
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 [DescribeVoices](#)。

Ruby

SDK for Ruby

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
require 'aws-sdk-polly' # In v2: require 'aws-sdk'

begin
  # Create an Amazon Polly client using
  # credentials from the shared credentials file ~/.aws/credentials
```

```
# and the configuration (region) from the shared configuration file ~/.aws/
config
polly = Aws::Polly::Client.new

# Get US English voices
resp = polly.describe_voices(language_code: 'en-US')

resp.voices.each do |v|
  puts v.name
  puts "  #{v.gender}"
  puts
end
rescue StandardError => e
  puts 'Could not get voices'
  puts 'Error message:'
  puts e.message
end
```

- 如需 API 詳細資訊，請參閱《適用於 Ruby 的 AWS SDK API 參考》中的 [DescribeVoices](#)。

Rust

適用於 Rust 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
async fn list_voices(client: &Client) -> Result<(), Error> {
  let resp = client.describe_voices().send().await?;

  println!("Voices:");

  let voices = resp.voices();
  for voice in voices {
    println!("  Name:      {}", voice.name().unwrap_or("No name!"));
    println!(
      "    Language: {}",
      voice.language_name().unwrap_or("No language!")
    );
  }
}
```

```

    );

    println!();
}

println!("Found {} voices", voices.len());

Ok(())
}

```

- 如需 API 詳細資訊，請參閱《AWS SDK for Rust API 參考》中的 [DescribeVoices](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

TRY.
    " Only pass optional parameters if they have values
    IF iv_engine IS NOT INITIAL AND iv_language IS NOT INITIAL.
        oo_result = lo_ply->describevoices(
            iv_engine = iv_engine
            iv_languagecode = iv_language ).
    ELSEIF iv_engine IS NOT INITIAL.
        oo_result = lo_ply->describevoices(
            iv_engine = iv_engine ).
    ELSEIF iv_language IS NOT INITIAL.
        oo_result = lo_ply->describevoices(
            iv_languagecode = iv_language ).
    ELSE.
        oo_result = lo_ply->describevoices( ).
    ENDIF.
    MESSAGE 'Retrieved voice metadata.' TYPE 'I'.
CATCH /aws1/cx_plyinvalidnexttokenex.
    MESSAGE 'The NextToken is invalid.' TYPE 'E'.
CATCH /aws1/cx_plyservicefailureex.

```

```
MESSAGE 'Service failure occurred.' TYPE 'E'.  
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [DescribeVoices](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [搭配 SDK 使用 Amazon Polly AWS](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

GetLexicon 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 GetLexicon。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [text-to-speech合成入門](#)

.NET

適用於 .NET 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
using System;  
using System.Threading.Tasks;  
using Amazon.Polly;  
using Amazon.Polly.Model;  
  
/// <summary>  
/// Retrieves information about a specific Amazon Polly lexicon.  
/// </summary>  
public class GetLexicon  
{  
    public static async Task Main(string[] args)
```

```
{
    string lexiconName = "SampleLexicon";

    var client = new AmazonPollyClient();

    await GetPollyLexiconAsync(client, lexiconName);
}

public static async Task GetPollyLexiconAsync(AmazonPollyClient client,
string lexiconName)
{
    var getLexiconRequest = new GetLexiconRequest()
    {
        Name = lexiconName,
    };

    try
    {
        var response = await client.GetLexiconAsync(getLexiconRequest);
        Console.WriteLine($"Lexicon:\n Name: {response.Lexicon.Name}");
        Console.WriteLine($"Content: {response.Lexicon.Content}");
    }
    catch (Exception ex)
    {
        Console.WriteLine("Error: " + ex.Message);
    }
}
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [GetLexicon](#)。

CLI

AWS CLI

擷取語彙的內容

下列 get-lexicon 範例會擷取指定發音語彙的內容。

```
aws polly get-lexicon \
```

```
--name w3c
```

輸出：

```
{
  "Lexicon": {
    "Content": "<?xml version=\"1.0\" encoding=\"UTF-8\"?>\n<lexicon version=
\"1.0\" \n      xmlns=      \"http://www.w3.org/2005/01/pronunciation-lexicon
\" \n      xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\" \n
xsi:schemaLocation=\"http://www.w3.org/2005/01/pronunciation-lexicon \n
http://www.w3.org/TR/2007/CR-pronunciation-lexicon-20071212/pls.xsd\" \n
alphabet=\"ipa\" \n      xml:lang=\"en-US\">\n  <lexeme>\n    <grapheme>W3C</
grapheme>\n      <alias>World Wide Web Consortium</alias>\n    </lexeme>\n</
lexicon>\n",
    "Name": "w3c"
  },
  "LexiconAttributes": {
    "Alphabet": "ipa",
    "LanguageCode": "en-US",
    "LastModified": 1603908910.99,
    "LexiconArn": "arn:aws:polly:us-west-2:880185128111:lexicon/w3c",
    "LexemesCount": 1,
    "Size": 492
  }
}
```

如需詳細資訊，請參閱《Amazon Polly 開發人員指南》中的[使用 GetLexicon 操作](#)。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的[GetLexicon](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
class PollyWrapper:
```

```
"""Encapsulates Amazon Polly functions."""

def __init__(self, polly_client, s3_resource):
    """
    :param polly_client: A Boto3 Amazon Polly client.
    :param s3_resource: A Boto3 Amazon Simple Storage Service (Amazon S3)
resource.
    """
    self.polly_client = polly_client
    self.s3_resource = s3_resource
    self.voice_metadata = None

def get_lexicon(self, name):
    """
    Gets metadata and contents of an existing lexicon.

    :param name: The name of the lexicon to retrieve.
    :return: The retrieved lexicon.
    """
    try:
        response = self.polly_client.get_lexicon(Name=name)
        logger.info("Got lexicon %s.", name)
    except ClientError:
        logger.exception("Couldn't get lexicon %s.", name)
        raise
    else:
        return response
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 [GetLexicon](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

TRY.
  oo_result = lo_ply->getlexicon( iv_name ).
  DATA(lo_lexicon) = oo_result->get_lexicon( ).
  IF lo_lexicon IS BOUND.
    DATA(lv_lex_name) = lo_lexicon->get_name( ).
    MESSAGE |Retrieved lexicon: { lv_lex_name }| TYPE 'I'.
  ENDIF.
CATCH /aws1/cx_plylexiconnotfoundex.
  MESSAGE 'Lexicon not found.' TYPE 'E'.
CATCH /aws1/cx_plyservicefailureex.
  MESSAGE 'Service failure occurred.' TYPE 'E'.
ENDTRY.

```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [GetLexicon](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [搭配 SDK 使用 Amazon Polly AWS](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

GetSpeechSynthesisTask 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 GetSpeechSynthesisTask。

CLI

AWS CLI

取得語音合成任務的相關資訊

下列 get-speech-synthesis-task 範例會擷取指定語音合成任務的相關資訊。

```

aws polly get-speech-synthesis-task \
  --task-id 70b61c0f-57ce-4715-a247-cae8729dcce9

```

輸出：

```

{
  "SynthesisTask": {
    "TaskId": "70b61c0f-57ce-4715-a247-cae8729dcce9",
    "TaskStatus": "completed",

```

```

        "OutputUri": "https://s3.us-west-2.amazonaws.com/amzn-s3-demo-
bucket/70b61c0f-57ce-4715-a247-cae8729dcce9.mp3",
        "CreationTime": 1603911042.689,
        "RequestCharacters": 1311,
        "OutputFormat": "mp3",
        "TextType": "text",
        "VoiceId": "Joanna"
    }
}

```

如需詳細資訊，請參閱《Amazon Polly 開發人員指南》中的[建立長時間音訊檔案](#)。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的[GetSpeechSynthesisTask](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在[AWS 程式碼範例儲存庫](#)中設定和執行。

```

class PollyWrapper:
    """Encapsulates Amazon Polly functions."""

    def __init__(self, polly_client, s3_resource):
        """
        :param polly_client: A Boto3 Amazon Polly client.
        :param s3_resource: A Boto3 Amazon Simple Storage Service (Amazon S3)
        resource.
        """
        self.polly_client = polly_client
        self.s3_resource = s3_resource
        self.voice_metadata = None

    def get_speech_synthesis_task(self, task_id):
        """
        Gets metadata about an asynchronous speech synthesis task, such as its
        status.

```

```

:param task_id: The ID of the task to retrieve.
:return: Metadata about the task.
"""
try:
    response =
self.polly_client.get_speech_synthesis_task(TaskId=task_id)
    task = response["SynthesisTask"]
    logger.info("Got synthesis task. Status is %s.", task["TaskStatus"])
except ClientError:
    logger.exception("Couldn't get synthesis task %s.", task_id)
    raise
else:
    return task

```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 [GetSpeechSynthesisTask](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

TRY.
    oo_result = lo_ply->getspeechsynthesistask( iv_task_id ).
    DATA(lo_task) = oo_result->get_synthesistask( ).
    IF lo_task IS BOUND.
        DATA(lv_status) = lo_task->get_taskstatus( ).
        MESSAGE |Task status: { lv_status }| TYPE 'I'.
    ENDIF.
CATCH /aws1/cx_plyinvalidtaskidex.
    MESSAGE 'Invalid task ID.' TYPE 'E'.
CATCH /aws1/cx_plyservicefailureex.
    MESSAGE 'Service failure occurred.' TYPE 'E'.

```

```
CATCH /aws1/cx_plysynthesistsknotf00.  
    MESSAGE 'Synthesis task not found.' TYPE 'E'.  
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [GetSpeechSynthesisTask](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [搭配 SDK 使用 Amazon Polly AWS](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

ListLexicons 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 ListLexicons。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [text-to-speech合成入門](#)

.NET

適用於 .NET 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
using System;  
using System.Threading.Tasks;  
using Amazon.Polly;  
using Amazon.Polly.Model;  
  
/// <summary>  
/// Lists the Amazon Polly lexicons that have been defined. By default,  
/// lists the lexicons that are defined in the same AWS Region as the default  
/// user. To view Amazon Polly lexicons that are defined in a different AWS  
/// Region, supply it as a parameter to the Amazon Polly constructor.
```

```
/// </summary>
public class ListLexicons
{
    public static async Task Main()
    {
        var client = new AmazonPollyClient();
        var request = new ListLexiconsRequest();

        try
        {
            Console.WriteLine("All voices: ");

            do
            {
                var response = await client.ListLexiconsAsync(request);
                request.NextToken = response.NextToken;

                response.Lexicons.ForEach(lexicon =>
                {
                    var attributes = lexicon.Attributes;
                    Console.WriteLine($"Name: {lexicon.Name}");
                    Console.WriteLine($"\\tAlphabet: {attributes.Alphabet}");
                    Console.WriteLine($"\\tLanguageCode:
{attributes.LanguageCode}");
                    Console.WriteLine($"\\tLastModified:
{attributes.LastModified}");
                    Console.WriteLine($"\\tLexemesCount:
{attributes.LexemesCount}");
                    Console.WriteLine($"\\tLexiconArn:
{attributes.LexiconArn}");
                    Console.WriteLine($"\\tSize: {attributes.Size}");
                });
            }
            while (request.NextToken is not null);
        }
        catch (Exception ex)
        {
            Console.WriteLine($"Error: {ex.Message}");
        }
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [ListLexicons](#)。

CLI

AWS CLI

列出語彙

下列 `list-lexicons` 範例會列出您的發音語彙。

```
aws polly list-lexicons
```

輸出：

```
{
  "Lexicons": [
    {
      "Name": "w3c",
      "Attributes": {
        "Alphabet": "ipa",
        "LanguageCode": "en-US",
        "LastModified": 1603908910.99,
        "LexiconArn": "arn:aws:polly:us-east-2:123456789012:lexicon/w3c",
        "LexemesCount": 1,
        "Size": 492
      }
    }
  ]
}
```

如需詳細資訊，請參閱《Amazon Polly 開發人員指南》中的 [使用 ListLexicons 操作](#)。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [ListLexicons](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.polly.PollyClient;
import software.amazon.awssdk.services.polly.model.ListLexiconsResponse;
import software.amazon.awssdk.services.polly.model.ListLexiconsRequest;
import software.amazon.awssdk.services.polly.model.LexiconDescription;
import software.amazon.awssdk.services.polly.model.PollyException;
import java.util.List;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
 * started.html
 */
public class ListLexicons {
    public static void main(String args[]) {
        PollyClient polly = PollyClient.builder()
            .region(Region.US_WEST_2)
            .build();

        listLexicons(polly);
        polly.close();
    }

    public static void listLexicons(PollyClient client) {
        try {
            ListLexiconsRequest listLexiconsRequest =
                ListLexiconsRequest.builder()
                    .build();
```

```

        ListLexiconsResponse listLexiconsResult =
client.listLexicons(listLexiconsRequest);
        List<LexiconDescription> lexiconDescription =
listLexiconsResult.lexicons();
        for (LexiconDescription lexDescription : lexiconDescription) {
            System.out.println("The name of the Lexicon is " +
lexDescription.name());
        }

    } catch (PollyException e) {
        System.err.println("Exception caught: " + e);
        System.exit(1);
    }
}
}

```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [ListLexicons](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

class PollyWrapper:
    """Encapsulates Amazon Polly functions."""

    def __init__(self, polly_client, s3_resource):
        """
        :param polly_client: A Boto3 Amazon Polly client.
        :param s3_resource: A Boto3 Amazon Simple Storage Service (Amazon S3)
resource.
        """
        self.polly_client = polly_client
        self.s3_resource = s3_resource
        self.voice_metadata = None

```

```
def list_lexicons(self):
    """
    Lists lexicons in the current account.

    :return: The list of lexicons.
    """
    try:
        response = self.polly_client.list_lexicons()
        lexicons = response["Lexicons"]
        logger.info("Got %s lexicons.", len(lexicons))
    except ClientError:
        logger.exception(
            "Couldn't get %s.",
        )
        raise
    else:
        return lexicons
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 [ListLexicons](#)。

Ruby

SDK for Ruby

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
require 'aws-sdk-polly' # In v2: require 'aws-sdk'

begin
  # Create an Amazon Polly client using
  # credentials from the shared credentials file ~/.aws/credentials
  # and the configuration (region) from the shared configuration file ~/.aws/
  config
```

```

polly = Aws::Polly::Client.new

resp = polly.list_lexicons

resp.lexicons.each do |l|
  puts l.name
  puts "  Alphabet:#{l.attributes.alphabet}"
  puts "  Language:#{l.attributes.language}"
  puts
end
rescue StandardError => e
  puts 'Could not get lexicons'
  puts 'Error message:'
  puts e.message
end

```

- 如需 API 詳細資訊，請參閱《適用於 Ruby 的 AWS SDK API 參考》中的 [ListLexicons](#)。

Rust

適用於 Rust 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

async fn show_lexicons(client: &Client) -> Result<(), Error> {
  let resp = client.list_lexicons().send().await?;

  println!("Lexicons:");

  let lexicons = resp.lexicons();

  for lexicon in lexicons {
    println!("  Name:      {}", lexicon.name().unwrap_or_default());
    println!(
      "    Language: {:?}\n",
      lexicon
        .attributes()

```

```

        .as_ref()
        .map(|attrib| attrib
            .language_code
            .as_ref()
            .expect("languages must have language codes"))
        .expect("languages must have attributes")
    );
}

println!();
println!("Found {} lexicons.", lexicons.len());
println!();

Ok(())
}

```

- 如需 API 詳細資訊，請參閱《AWS SDK for Rust API 參考》中的 [ListLexicons](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

TRY.
    oo_result = lo_ply->listlexicons( ).
    DATA(lt_lexicons) = oo_result->get_lexicons( ).
    DATA(lv_count) = lines( lt_lexicons ).
    MESSAGE |Found { lv_count } lexicons| TYPE 'I'.
CATCH /aws1/cx_plyinvalidnexttokenex.
    MESSAGE 'Invalid NextToken.' TYPE 'E'.
CATCH /aws1/cx_plyservicefailureex.
    MESSAGE 'Service failure occurred.' TYPE 'E'.
ENDTRY.

```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [ListLexicons](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [搭配 SDK 使用 Amazon Polly AWS](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

ListSpeechSynthesisTasks 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 ListSpeechSynthesisTasks。

CLI

AWS CLI

列出您的語音合成任務

下列 list-speech-synthesis-tasks 範例會列出您的語音合成任務。

```
aws polly list-speech-synthesis-tasks
```

輸出：

```
{
  "SynthesisTasks": [
    {
      "TaskId": "70b61c0f-57ce-4715-a247-cae8729dcce9",
      "TaskStatus": "completed",
      "OutputUri": "https://s3.us-west-2.amazonaws.com/amzn-s3-demo-bucket/70b61c0f-57ce-4715-a247-cae8729dcce9.mp3",
      "CreationTime": 1603911042.689,
      "RequestCharacters": 1311,
      "OutputFormat": "mp3",
      "TextType": "text",
      "VoiceId": "Joanna"
    }
  ]
}
```

如需詳細資訊，請參閱《Amazon Polly 開發人員指南》中的 [建立長時間音訊檔案](#)。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [ListSpeechSynthesisTasks](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
TRY.  
    " Only pass optional parameters if they have values  
    IF iv_max_results IS NOT INITIAL AND iv_status IS NOT INITIAL.  
        oo_result = lo_ply->listspeechsynthesistasks(  
            iv_maxresults = iv_max_results  
            iv_status = iv_status ).  
    ELSEIF iv_max_results IS NOT INITIAL.  
        oo_result = lo_ply->listspeechsynthesistasks(  
            iv_maxresults = iv_max_results ).  
    ELSEIF iv_status IS NOT INITIAL.  
        oo_result = lo_ply->listspeechsynthesistasks(  
            iv_status = iv_status ).  
    ELSE.  
        oo_result = lo_ply->listspeechsynthesistasks( ).  
    ENDIF.  
    DATA(lt_tasks) = oo_result->get_synthesistasks( ).  
    DATA(lv_count) = lines( lt_tasks ).  
    MESSAGE |Found { lv_count } synthesis tasks| TYPE 'I'.  
    CATCH /aws1/cx_plyinvalidnexttokenex.  
        MESSAGE 'Invalid NextToken.' TYPE 'E'.  
    CATCH /aws1/cx_plyservicefailureex.  
        MESSAGE 'Service failure occurred.' TYPE 'E'.  
ENDTRY.
```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [ListSpeechSynthesisTasks](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [搭配 SDK 使用 Amazon Polly AWS](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

PutLexicon 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 PutLexicon。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [text-to-speech合成入門](#)

.NET

適用於 .NET 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
using System;
using System.Threading.Tasks;
using Amazon.Polly;
using Amazon.Polly.Model;

/// <summary>
/// Creates a new Amazon Polly lexicon using the AWS SDK for .NET.
/// </summary>
public class PutLexicon
{
    public static async Task Main()
    {
        string lexiconContent = "<?xml version=\"1.0\" encoding=\"UTF-8\"?>"
+
        "<lexicon version=\"1.0\" xmlns=\"http://www.w3.org/2005/01/"
pronunciation-lexicon\" xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\" "
+
        "xsi:schemaLocation=\"http://www.w3.org/2005/01/pronunciation-
lexicon http://www.w3.org/TR/2007/CR-pronunciation-lexicon-20071212/pls.xsd\" " +
        "alphabet=\"ipa\" xml:lang=\"en-US\">" +
        "<lexeme><grapheme>test1</grapheme><alias>test2</alias></lexeme>"
+
    }
```

```
        "</lexicon>";
        string lexiconName = "SampleLexicon";

        var client = new AmazonPollyClient();
        var putLexiconRequest = new PutLexiconRequest()
        {
            Name = lexiconName,
            Content = lexiconContent,
        };

        try
        {
            var response = await client.PutLexiconAsync(putLexiconRequest);
            if (response.HttpStatusCode == System.Net.HttpStatusCode.OK)
            {
                Console.WriteLine($"Successfully created Lexicon:
{lexiconName}.");
            }
            else
            {
                Console.WriteLine($"Could not create Lexicon:
{lexiconName}.");
            }
        }
        catch (Exception ex)
        {
            Console.WriteLine("Exception caught: " + ex.Message);
        }
    }
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [PutLexicon](#)。

CLI

AWS CLI

存放語彙

下列 put-lexicon 範例存放指定的發音語彙。example.pls 檔案會指定 W3C PLS 相容語彙。

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

example.pls 的內容

```
{  
  <?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>  
}
```

此命令不會產生輸出。

如需詳細資訊，請參閱《Amazon Polly 開發人員指南》中的[使用 PutLexicon 操作](#)。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的[PutLexicon](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在[AWS 程式碼範例儲存庫](#)中設定和執行。

```
class PollyWrapper:  
    """Encapsulates Amazon Polly functions."""
```

```

def __init__(self, polly_client, s3_resource):
    """
    :param polly_client: A Boto3 Amazon Polly client.
    :param s3_resource: A Boto3 Amazon Simple Storage Service (Amazon S3)
resource.
    """
    self.polly_client = polly_client
    self.s3_resource = s3_resource
    self.voice_metadata = None

def create_lexicon(self, name, content):
    """
    Creates a lexicon with the specified content. A lexicon contains custom
pronunciations.

    :param name: The name of the lexicon.
    :param content: The content of the lexicon.
    """
    try:
        self.polly_client.put_lexicon(Name=name, Content=content)
        logger.info("Created lexicon %s.", name)
    except ClientError:
        logger.exception("Couldn't create lexicon %s.")
        raise

```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 [PutLexicon](#)。

Rust

適用於 Rust 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

async fn make_lexicon(client: &Client, name: &str, from: &str, to: &str) ->
Result<(), Error> {

```

```

let content = format!("<?xml version=\"1.0\" encoding=\"UTF-8\"?>
<lexicon version=\"1.0\" xmlns=\"http://www.w3.org/2005/01/pronunciation-
lexicon\" xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\"
xsi:schemaLocation=\"http://www.w3.org/2005/01/pronunciation-lexicon http://
www.w3.org/TR/2007/CR-pronunciation-lexicon-20071212/pls.xsd\"
alphabet=\"ipa\" xml:lang=\"en-US\">
<lexeme><grapheme>{}</grapheme><alias>{}</alias></lexeme>
</lexicon>", from, to);

client
    .put_lexicon()
    .name(name)
    .content(content)
    .send()
    .await?;

println!("Added lexicon");

Ok(())
}

```

- 如需 API 詳細資訊，請參閱《AWS SDK for Rust API 參考》中的 [PutLexicon](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

TRY.
    lo_ply->putlexicon(
        iv_name = iv_name
        iv_content = iv_content ).
    MESSAGE 'Lexicon created successfully.' TYPE 'I'.
CATCH /aws1/cx_plyinvalidlexiconex.
    MESSAGE 'Invalid lexicon.' TYPE 'E'.
CATCH /aws1/cx_plylexiconsizeexcdex.

```

```

    MESSAGE 'Lexicon size exceeded.' TYPE 'E'.
  CATCH /aws1/cx_plymaxlexemelengthe00.
    MESSAGE 'Maximum lexeme length exceeded.' TYPE 'E'.
  CATCH /aws1/cx_plymaxlexiconsnoexc00.
    MESSAGE 'Maximum number of lexicons exceeded.' TYPE 'E'.
  CATCH /aws1/cx_plyservicefailureex.
    MESSAGE 'Service failure occurred.' TYPE 'E'.
  CATCH /aws1/cx_plyunsuppdpalsalpha00.
    MESSAGE 'Unsupported PLS alphabet.' TYPE 'E'.
  CATCH /aws1/cx_plyunsuppdpyslangu00.
    MESSAGE 'Unsupported PLS language.' TYPE 'E'.
ENDTRY.

```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [PutLexicon](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [搭配 SDK 使用 Amazon Polly AWS](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

StartSpeechSynthesisTask 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 StartSpeechSynthesisTask。

CLI

AWS CLI

合成文字

下列 start-speech-synthesis-task 範例會在 text_file.txt 中合成文字，並將產生的 MP3 檔案存放在指定的儲存貯體中。

```

aws polly start-speech-synthesis-task \
  --output-format mp3 \
  --output-s3-bucket-name amzn-s3-demo-bucket \
  --text file://text_file.txt \
  --voice-id Joanna

```

輸出：

```

{
  "SynthesisTask": {

```

```

        "TaskId": "70b61c0f-57ce-4715-a247-cae8729dcce9",
        "TaskStatus": "scheduled",
        "OutputUri": "https://s3.us-east-2.amazonaws.com/amzn-s3-demo-
bucket/70b61c0f-57ce-4715-a247-cae8729dcce9.mp3",
        "CreationTime": 1603911042.689,
        "RequestCharacters": 1311,
        "OutputFormat": "mp3",
        "TextType": "text",
        "VoiceId": "Joanna"
    }
}

```

如需詳細資訊，請參閱《Amazon Polly 開發人員指南》中的[建立長時間音訊檔案](#)。

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的 [StartSpeechSynthesisTask](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

class PollyWrapper:
    """Encapsulates Amazon Polly functions."""

    def __init__(self, polly_client, s3_resource):
        """
        :param polly_client: A Boto3 Amazon Polly client.
        :param s3_resource: A Boto3 Amazon Simple Storage Service (Amazon S3)
        resource.
        """
        self.polly_client = polly_client
        self.s3_resource = s3_resource
        self.voice_metadata = None

    def do_synthesis_task(
        self,

```

```

    text,
    engine,
    voice,
    audio_format,
    s3_bucket,
    lang_code=None,
    include_visemes=False,
    wait_callback=None,
):
    """
    Start an asynchronous task to synthesize speech or speech marks, wait for
    the task to complete, retrieve the output from Amazon S3, and return the
    data.

    An asynchronous task is required when the text is too long for near-real
    time
    synthesis.

    :param text: The text to synthesize.
    :param engine: The kind of engine used. Can be standard or neural.
    :param voice: The ID of the voice to use.
    :param audio_format: The audio format to return for synthesized speech.
    When
        speech marks are synthesized, the output format is
    JSON.
    :param s3_bucket: The name of an existing Amazon S3 bucket that you have
        write access to. Synthesis output is written to this
    bucket.
    :param lang_code: The language code of the voice to use. This has an
    effect
        only when a bilingual voice is selected.
    :param include_visemes: When True, a second request is made to Amazon
    Polly
        to synthesize a list of visemes, using the
    specified
        text and voice. A viseme represents the visual
    position
        of the face and mouth when saying part of a word.
    :param wait_callback: A callback function that is called periodically
    during
        task processing, to give the caller an opportunity
    to
        take action, such as to display status.
    :return: The audio stream that contains the synthesized speech and a list

```

```

        of visemes that are associated with the speech audio.
    """
    try:
        kwargs = {
            "Engine": engine,
            "OutputFormat": audio_format,
            "OutputS3BucketName": s3_bucket,
            "Text": text,
            "VoiceId": voice,
        }
        if lang_code is not None:
            kwargs["LanguageCode"] = lang_code
        response = self.polly_client.start_speech_synthesis_task(**kwargs)
        speech_task = response["SynthesisTask"]
        logger.info("Started speech synthesis task %s.",
speech_task["TaskId"])

        viseme_task = None
        if include_visemes:
            kwargs["OutputFormat"] = "json"
            kwargs["SpeechMarkTypes"] = ["viseme"]
            response =
self.polly_client.start_speech_synthesis_task(**kwargs)
            viseme_task = response["SynthesisTask"]
            logger.info("Started viseme synthesis task %s.",
viseme_task["TaskId"])
        except ClientError:
            logger.exception("Couldn't start synthesis task.")
            raise
        else:
            bucket = self.s3_resource.Bucket(s3_bucket)
            audio_stream = self._wait_for_task(
                10, speech_task["TaskId"], "speech", wait_callback, bucket
            )

            visemes = None
            if include_visemes:
                viseme_data = self._wait_for_task(
                    10, viseme_task["TaskId"], "viseme", wait_callback, bucket
                )
                visemes = [
                    json.loads(v) for v in viseme_data.read().decode().split() if
v
                ]

```

```
return audio_stream, visemes
```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 [StartSpeechSynthesisTask](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
TRY.
  " Only pass optional parameters if they have values
  IF iv_lang_code IS NOT INITIAL AND iv_s3_key_prefix IS NOT INITIAL.
    oo_result = lo_ply->startspeechsynthesistask(
      iv_engine = iv_engine
      iv_outputformat = iv_audio_format
      iv_outputs3bucketname = iv_s3_bucket
      iv_outputs3keyprefix = iv_s3_key_prefix
      iv_text = iv_text
      iv_voiceid = iv_voice_id
      iv_languagecode = iv_lang_code ).
  ELSEIF iv_lang_code IS NOT INITIAL.
    oo_result = lo_ply->startspeechsynthesistask(
      iv_engine = iv_engine
      iv_outputformat = iv_audio_format
      iv_outputs3bucketname = iv_s3_bucket
      iv_text = iv_text
      iv_voiceid = iv_voice_id
      iv_languagecode = iv_lang_code ).
  ELSEIF iv_s3_key_prefix IS NOT INITIAL.
    oo_result = lo_ply->startspeechsynthesistask(
      iv_engine = iv_engine
      iv_outputformat = iv_audio_format
```

```

        iv_outputs3bucketname = iv_s3_bucket
        iv_outputs3keyprefix = iv_s3_key_prefix
        iv_text = iv_text
        iv_voiceid = iv_voice_id ).
    ELSE.
        oo_result = lo_ply->startspeechsynthesistask(
            iv_engine = iv_engine
            iv_outputformat = iv_audio_format
            iv_outputs3bucketname = iv_s3_bucket
            iv_text = iv_text
            iv_voiceid = iv_voice_id ).
    ENDIF.
    MESSAGE 'Speech synthesis task started.' TYPE 'I'.
CATCH /aws1/cx_plyinvalids3bucketex.
    MESSAGE 'Invalid S3 bucket.' TYPE 'E'.
CATCH /aws1/cx_plyinvalidssmlex.
    MESSAGE 'Invalid SSML.' TYPE 'E'.
CATCH /aws1/cx_plylexiconnotfoundex.
    MESSAGE 'Lexicon not found.' TYPE 'E'.
CATCH /aws1/cx_plyservicefailureex.
    MESSAGE 'Service failure occurred.' TYPE 'E'.
CATCH /aws1/cx_plytextlengthexcdex.
    MESSAGE 'Text length exceeded maximum.' TYPE 'E'.
ENDTRY.

```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [StartSpeechSynthesisTask](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [搭配 SDK 使用 Amazon Polly AWS](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

SynthesizeSpeech 搭配 AWS SDK 使用

下列程式碼範例示範如何使用 SynthesizeSpeech。

動作範例是大型程式的程式碼摘錄，必須在內容中執行。您可以在下列程式碼範例的內容中看到此動作：

- [text-to-speech合成入門](#)

.NET

適用於 .NET 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
using System;
using System.IO;
using System.Threading.Tasks;
using Amazon.Polly;
using Amazon.Polly.Model;

public class SynthesizeSpeech
{
    public static async Task Main()
    {
        string outputFileName = "speech.mp3";
        string text = "Twas brillig, and the slithy toves did gyre and gimbol
in the wabe";

        var client = new AmazonPollyClient();
        var response = await PollySynthesizeSpeech(client, text);

        WriteSpeechToStream(response.AudioStream, outputFileName);
    }

    /// <summary>
    /// Calls the Amazon Polly SynthesizeSpeechAsync method to convert text
    /// to speech.
    /// </summary>
    /// <param name="client">The Amazon Polly client object used to connect
    /// to the Amazon Polly service.</param>
    /// <param name="text">The text to convert to speech.</param>
    /// <returns>A SynthesizeSpeechResponse object that includes an
    AudioStream
    /// object with the converted text.</returns>
    private static async Task<SynthesizeSpeechResponse>
    PollySynthesizeSpeech(IAmazonPolly client, string text)
```

```

    {
        var synthesizeSpeechRequest = new SynthesizeSpeechRequest()
        {
            OutputFormat = OutputFormat.Mp3,
            VoiceId = VoiceId.Joanna,
            Text = text,
        };

        var synthesizeSpeechResponse =
            await client.SynthesizeSpeechAsync(synthesizeSpeechRequest);

        return synthesizeSpeechResponse;
    }

    /// <summary>
    /// Writes the AudioStream returned from the call to
    /// SynthesizeSpeechAsync to a file in MP3 format.
    /// </summary>
    /// <param name="audioStream">The AudioStream returned from the
    /// call to the SynthesizeSpeechAsync method.</param>
    /// <param name="outputFileName">The full path to the file in which to
    /// save the audio stream.</param>
    private static void WriteSpeechToStream(Stream audioStream, string
outputFileName)
    {
        var outputStream = new FileStream(
            outputFileName,
            FileMode.Create,
            FileAccess.Write);
        byte[] buffer = new byte[2 * 1024];
        int readBytes;

        while ((readBytes = audioStream.Read(buffer, 0, 2 * 1024)) > 0)
        {
            outputStream.Write(buffer, 0, readBytes);
        }

        // Flushes the buffer to avoid losing the last second or so of
        // the synthesized text.
        outputStream.Flush();
        Console.WriteLine($"Saved {outputFileName} to disk.");
    }
}

```

使用 SDK 搭配 Amazon Polly 使用語音標記從文字合成語音 AWS 。

```
using System;
using System.Collections.Generic;
using System.IO;
using System.Threading.Tasks;
using Amazon.Polly;
using Amazon.Polly.Model;

public class SynthesizeSpeechMarks
{
    public static async Task Main()
    {
        var client = new AmazonPollyClient();
        string outputFileName = "speechMarks.json";

        var synthesizeSpeechRequest = new SynthesizeSpeechRequest()
        {
            OutputFormat = OutputFormat.Json,
            SpeechMarkTypes = new List<string>
            {
                SpeechMarkType.Viseme,
                SpeechMarkType.Word,
            },
            VoiceId = VoiceId.Joanna,
            Text = "This is a sample text to be synthesized.",
        };

        try
        {
            using (var outputStream = new FileStream(outputFileName,
                FileMode.Create, FileAccess.Write))
            {
                var synthesizeSpeechResponse = await
                    client.SynthesizeSpeechAsync(synthesizeSpeechRequest);
                var buffer = new byte[2 * 1024];
                int readBytes;

                var inputStream = synthesizeSpeechResponse.AudioStream;
                while ((readBytes = inputStream.Read(buffer, 0, 2 * 1024)) >
                    0)
```

```
        {
            outputStream.Write(buffer, 0, readBytes);
        }
    }
}
catch (Exception ex)
{
    Console.WriteLine($"Error: {ex.Message}");
}
}
```

- 如需 API 詳細資訊，請參閱《適用於 .NET 的 AWS SDK API 參考》中的 [SynthesizeSpeech](#)。

Java

SDK for Java 2.x

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
import javazoom.jl.decoder.JavaLayerException;
import software.amazon.awssdk.core.ResponseInputStream;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.polly.PollyClient;
import software.amazon.awssdk.services.polly.model.DescribeVoicesRequest;
import software.amazon.awssdk.services.polly.model.Voice;
import software.amazon.awssdk.services.polly.model.DescribeVoicesResponse;
import software.amazon.awssdk.services.polly.model.OutputFormat;
import software.amazon.awssdk.services.polly.model.PollyException;
import software.amazon.awssdk.services.polly.model.SynthesizeSpeechRequest;
import software.amazon.awssdk.services.polly.model.SynthesizeSpeechResponse;
import java.io.IOException;
import java.io.InputStream;
import javazoom.jl.player.advanced.AdvancedPlayer;
```

```
import javazoom.jl.player.advanced.PlaybackEvent;
import javazoom.jl.player.advanced.PlaybackListener;

/**
 * Before running this Java V2 code example, set up your development
 * environment, including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-started.html
 */
public class PollyDemo {
    private static final String SAMPLE = "Congratulations. You have successfully
        built this working demo " +
        " of Amazon Polly in Java Version 2. Have fun building voice enabled
        apps with Amazon Polly (that's me!), and always "
        +
        " look at the AWS website for tips and tricks on using Amazon Polly
        and other great services from AWS";

    public static void main(String args[]) {
        PollyClient polly = PollyClient.builder()
            .region(Region.US_WEST_2)
            .build();

        talkPolly(polly);
        polly.close();
    }

    public static void talkPolly(PollyClient polly) {
        try {
            DescribeVoicesRequest describeVoiceRequest =
                DescribeVoicesRequest.builder()
                    .engine("standard")
                    .build();

            DescribeVoicesResponse describeVoicesResult =
                polly.describeVoices(describeVoiceRequest);
            Voice voice = describeVoicesResult.voices().stream()
                .filter(v -> v.name().equals("Joanna"))
                .findFirst()
                .orElseThrow(() -> new RuntimeException("Voice not found"));
        }
    }
}
```

```

        InputStream stream = synthesize(polly, SAMPLE, voice,
        OutputFormat.MP3);
        AdvancedPlayer player = new AdvancedPlayer(stream,

javazoom.jl.player.FactoryRegistry.systemRegistry().createAudioDevice());
        player.setPlaybackListener(new PlaybackListener() {
            public void playbackStarted(PlaybackEvent evt) {
                System.out.println("Playback started");
                System.out.println(SAMPLE);
            }

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

        // play it!
        player.play();

    } catch (PollyException | JavaLayerException | IOException e) {
        System.err.println(e.getMessage());
        System.exit(1);
    }
}

public static InputStream synthesize(PollyClient polly, String text, Voice
voice, OutputFormat format)
    throws IOException {
    SynthesizeSpeechRequest synthReq = SynthesizeSpeechRequest.builder()
        .text(text)
        .voiceId(voice.id())
        .outputFormat(format)
        .build();

    ResponseInputStream<SynthesizeSpeechResponse> synthRes =
polly.synthesizeSpeech(synthReq);
    return synthRes;
}
}

```

- 如需 API 詳細資訊，請參閱《AWS SDK for Java 2.x API 參考》中的 [SynthesizeSpeech](#)。

Python

適用於 Python 的 SDK (Boto3)

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
class PollyWrapper:
    """Encapsulates Amazon Polly functions."""

    def __init__(self, polly_client, s3_resource):
        """
        :param polly_client: A Boto3 Amazon Polly client.
        :param s3_resource: A Boto3 Amazon Simple Storage Service (Amazon S3)
        resource.
        """
        self.polly_client = polly_client
        self.s3_resource = s3_resource
        self.voice_metadata = None

    def synthesize(
        self, text, engine, voice, audio_format, lang_code=None,
        include_visemes=False
    ):
        """
        Synthesizes speech or speech marks from text, using the specified voice.

        :param text: The text to synthesize.
        :param engine: The kind of engine used. Can be standard or neural.
        :param voice: The ID of the voice to use.
        :param audio_format: The audio format to return for synthesized speech.
        When
                               speech marks are synthesized, the output format is
        JSON.
        :param lang_code: The language code of the voice to use. This has an
        effect
                               only when a bilingual voice is selected.
```

```

        :param include_visemes: When True, a second request is made to Amazon
Polly
                                to synthesize a list of visemes, using the
specified
                                text and voice. A viseme represents the visual
position
                                of the face and mouth when saying part of a word.
:return: The audio stream that contains the synthesized speech and a list
        of visemes that are associated with the speech audio.
"""
try:
    kwargs = {
        "Engine": engine,
        "OutputFormat": audio_format,
        "Text": text,
        "VoiceId": voice,
    }
    if lang_code is not None:
        kwargs["LanguageCode"] = lang_code
    response = self.polly_client.synthesize_speech(**kwargs)
    audio_stream = response["AudioStream"]
    logger.info("Got audio stream spoken by %s.", voice)
    visemes = None
    if include_visemes:
        kwargs["OutputFormat"] = "json"
        kwargs["SpeechMarkTypes"] = ["viseme"]
        response = self.polly_client.synthesize_speech(**kwargs)
        visemes = [
            json.loads(v)
            for v in response["AudioStream"].read().decode().split()
            if v
        ]
        logger.info("Got %s visemes.", len(visemes))
except ClientError:
    logger.exception("Couldn't get audio stream.")
    raise
else:
    return audio_stream, visemes

```

- 如需 API 詳細資訊，請參閱《AWS SDK for Python (Boto3) API 參考》中的 [SynthesizeSpeech](#)。

Ruby

SDK for Ruby

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```
require 'aws-sdk-polly' # In v2: require 'aws-sdk'

begin
  # Get the filename from the command line
  if ARGV.empty?
    puts 'You must supply a filename'
    exit 1
  end

  filename = ARGV[0]

  # Open file and get the contents as a string
  if File.exist?(filename)
    contents = IO.read(filename)
  else
    puts "No such file: #{filename}"
    exit 1
  end

  # Create an Amazon Polly client using
  # credentials from the shared credentials file ~/.aws/credentials
  # and the configuration (region) from the shared configuration file ~/.aws/
  config
  polly = Aws::Polly::Client.new

  resp = polly.synthesize_speech({
    output_format: 'mp3',
    text: contents,
    voice_id: 'Joanna'
  })
```

```

# Save output
# Get just the file name
# abc/xyz.txt -> xyz.txt
name = File.basename(filename)

# Split up name so we get just the xyz part
parts = name.split('.')
first_part = parts[0]
mp3_file = "#{first_part}.mp3"

IO.copy_stream(resp.audio_stream, mp3_file)

puts "Wrote MP3 content to: #{mp3_file}"
rescue StandardError => e
  puts 'Got error:'
  puts 'Error message:'
  puts e.message
end

```

- 如需 API 詳細資訊，請參閱《適用於 Ruby 的 AWS SDK API 參考》中的 [SynthesizeSpeech](#)。

Rust

適用於 Rust 的 SDK

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

```

async fn synthesize(client: &Client, filename: &str) -> Result<(), Error> {
  let content = fs::read_to_string(filename);

  let resp = client
    .synthesize_speech()
    .output_format(OutputFormat::Mp3)
    .text(content.unwrap())
    .voice_id(VoiceId::Joanna)

```

```

        .send()
        .await?;

// Get MP3 data from response and save it
let mut blob = resp
    .audio_stream
    .collect()
    .await
    .expect("failed to read data");

let parts: Vec<&str> = filename.split('.').collect();
let out_file = format!("{}", String::from(parts[0]), ".mp3");

let mut file = tokio::fs::File::create(out_file)
    .await
    .expect("failed to create file");

file.write_all_buf(&mut blob)
    .await
    .expect("failed to write to file");

Ok(())
}

```

- 如需 API 詳細資訊，請參閱《AWS SDK for Rust API 參考》中的 [SynthesizeSpeech](#)。

SAP ABAP

適用於 SAP ABAP 的開發套件

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在 [AWS 程式碼範例儲存庫](#) 中設定和執行。

TRY.

```

" Only pass optional language code if it has a value
IF iv_lang_code IS NOT INITIAL.
    oo_result = lo_ply->synthesizespeech(
        iv_engine = iv_engine

```

```

        iv_outputformat = iv_output_fmt
        iv_text = iv_text
        iv_voiceid = iv_voice_id
        iv_languagecode = iv_lang_code ).
    ELSE.
        oo_result = lo_ply->synthesizespeech(
            iv_engine = iv_engine
            iv_outputformat = iv_output_fmt
            iv_text = iv_text
            iv_voiceid = iv_voice_id ).
    ENDIF.
    MESSAGE 'Speech synthesized successfully.' TYPE 'I'.
CATCH /aws1/cx_plyinvalidssmlex.
    MESSAGE 'Invalid SSML.' TYPE 'E'.
CATCH /aws1/cx_plylexiconnotfoundex.
    MESSAGE 'Lexicon not found.' TYPE 'E'.
CATCH /aws1/cx_plyservicefailureex.
    MESSAGE 'Service failure occurred.' TYPE 'E'.
CATCH /aws1/cx_plytextlengthexcdex.
    MESSAGE 'Text length exceeded maximum.' TYPE 'E'.
ENDTRY.

```

- 如需 API 詳細資訊，請參閱《適用於 AWS SAP ABAP 的 SDK API 參考》中的 [SynthesizeSpeech](#)。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [搭配 SDK 使用 Amazon Polly AWS](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

Amazon Polly AWS SDKs 案例

下列程式碼範例示範如何在 Amazon Polly AWS SDKs 中實作常見案例。這些案例示範如何呼叫 Amazon Polly 中的多個函數，或與其他 AWS 服務結合，藉以完成特定任務。每個案例均包含完整原始碼的連結，您可在連結中找到如何設定和執程式碼的相關指示。

案例的目標是獲得中等水平的經驗，協助您了解內容中的服務動作。

範例

- [使用 AWS SDK 將文字轉換為語音並返回文字](#)
- [使用 SDK 與 Amazon Polly 建立唇同步應用程式 AWS](#)

- [建立可分析客戶意見回饋並合成音訊的應用程式](#)
- [text-to-speech合成入門](#)

使用 AWS SDK 將文字轉換為語音並返回文字

以下程式碼範例顯示做法：

- 使用 Amazon Polly 將純文字 (UTF-8) 輸入檔案合成至音訊檔案中。
- 將音訊檔案上傳至 Amazon S3 儲存貯體。
- 使用 Amazon Transcribe 將音訊檔案轉換為文字。
- 顯示文字。

Rust

適用於 Rust 的 SDK

使用 Amazon Polly 將純文字 (UTF-8) 輸入檔案合成至音訊檔案中，將音訊檔案上傳至 Amazon S3 儲存貯體，使用 Amazon Transcribe 將該音訊檔案轉換為文字，然後顯示文字。

如需完整的原始碼和如何設定及執行的指示，請參閱 [GitHub](#) 上的完整範例。

此範例中使用的服務

- Amazon Polly
- Amazon S3
- Amazon Transcribe

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [搭配 SDK 使用 Amazon Polly AWS](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

使用 SDK 與 Amazon Polly 建立唇同步應用程式 AWS

下列程式碼範例示範如何搭配使用 lip-sync 應用程式和 Amazon Polly。

Python

適用於 Python 的 SDK (Boto3)

示範如何使用 Amazon Polly 和 Tkinter 建立 lip-sync 應用程式，在該應用程式中會顯示動畫人臉隨著 Amazon Polly 合成的語音說話。透過向 Amazon Polly 請求與合成語音相符的單字清單，即可完成同步對嘴。

- 從 Amazon Polly 取得語音中繼資料，並將其顯示在 Tkinter 應用程式中。
- 從 Amazon Polly 取得合成語音音訊和相符的視素語音標記。
- 隨著動畫人臉的嘴部動作同步播放音訊。
- 提交長文字的非同步合成任務，並從 Amazon Simple Storage Service (Amazon S3) 儲存貯體擷取輸出。

如需完整的原始碼和如何設定及執行的指示，請參閱 [GitHub](#) 上的完整範例。

此範例中使用的服務

- Amazon Polly

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [搭配 SDK 使用 Amazon Polly AWS](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

建立可分析客戶意見回饋並合成音訊的應用程式

下列程式碼範例示範如何建立應用程式，以分析客戶評論卡、從其原始語言進行翻譯、判斷對方情緒，以及透過翻譯後的文字產生音訊檔案。

.NET

適用於 .NET 的 SDK

此範例應用程式會分析和存儲客戶的意見回饋卡。具體來說，它滿足了紐約市一家虛構飯店的需求。飯店以實體評論卡的形式收到賓客以各種語言撰寫的意見回饋。這些意見回饋透過 Web 用戶端上傳至應用程式。評論卡的影像上傳後，系統會執行下列步驟：

- 文字內容是使用 Amazon Textract 從影像中擷取。
- Amazon Comprehend 會決定擷取文字及其用語的情感。
- 擷取的文字內容會使用 Amazon Translate 翻譯成英文。

- Amazon Polly 會使用擷取的文字內容合成音訊檔案。

完整的應用程式可透過 AWS CDK 部署。如需原始程式碼和部署的說明，請參閱 [GitHub](#) 中的專案。

此範例中使用的服務

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

Java

適用於 Java 2.x 的 SDK

此範例應用程式會分析和存儲客戶的意見回饋卡。具體來說，它滿足了紐約市一家虛構飯店的需求。飯店以實體評論卡的形式收到賓客以各種語言撰寫的意見回饋。這些意見回饋透過 Web 用戶端上傳至應用程式。評論卡的影像上傳後，系統會執行下列步驟：

- 文字內容是使用 Amazon Textract 從影像中擷取。
- Amazon Comprehend 會決定擷取文字及其用語的情感。
- 擷取的文字內容會使用 Amazon Translate 翻譯成英文。
- Amazon Polly 會使用擷取的文字內容合成音訊檔案。

完整的應用程式可透過 AWS CDK 部署。如需原始程式碼和部署的說明，請參閱 [GitHub](#) 中的專案。

此範例中使用的服務

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

JavaScript

適用於 JavaScript (v3) 的 SDK

此範例應用程式會分析和存儲客戶的意見回饋卡。具體來說，它滿足了紐約市一家虛構飯店的需求。飯店以實體評論卡的形式收到賓客以各種語言撰寫的意見回饋。這些意見回饋透過 Web 用戶端上傳至應用程式。評論卡的影像上傳後，系統會執行下列步驟：

- 文字內容是使用 Amazon Textract 從影像中擷取。
- Amazon Comprehend 會決定擷取文字及其用語的情感。
- 擷取的文字內容會使用 Amazon Translate 翻譯成英文。
- Amazon Polly 會使用擷取的文字內容合成音訊檔案。

完整的應用程式可透過 AWS CDK 部署。如需原始程式碼和部署的說明，請參閱 [GitHub](#) 中的專案。以下摘錄顯示如何在 Lambda 函數內適用於 JavaScript 的 AWS SDK 使用。

```
import {
  ComprehendClient,
  DetectDominantLanguageCommand,
  DetectSentimentCommand,
} from "@aws-sdk/client-comprehend";

/**
 * Determine the language and sentiment of the extracted text.
 *
 * @param {{ source_text: string }} extractTextOutput
 */
export const handler = async (extractTextOutput) => {
  const comprehendClient = new ComprehendClient({});

  const detectDominantLanguageCommand = new DetectDominantLanguageCommand({
    Text: extractTextOutput.source_text,
  });

  // The source language is required for sentiment analysis and
  // translation in the next step.
  const { Languages } = await comprehendClient.send(
    detectDominantLanguageCommand,
  );

  const languageCode = Languages[0].LanguageCode;
```

```

const detectSentimentCommand = new DetectSentimentCommand({
  Text: extractTextOutput.source_text,
  LanguageCode: languageCode,
});

const { Sentiment } = await comprehendClient.send(detectSentimentCommand);

return {
  sentiment: Sentiment,
  language_code: languageCode,
};
};

```

```

import {
  DetectDocumentTextCommand,
  TextractClient,
} from "@aws-sdk/client-textract";

/**
 * Fetch the S3 object from the event and analyze it using Amazon Textract.
 *
 * @param {import("@types/aws-lambda").EventBridgeEvent<"Object Created">}
eventBridgeS3Event
 */
export const handler = async (eventBridgeS3Event) => {
  const textractClient = new TextractClient();

  const detectDocumentTextCommand = new DetectDocumentTextCommand({
    Document: {
      S3Object: {
        Bucket: eventBridgeS3Event.bucket,
        Name: eventBridgeS3Event.object,
      },
    },
  });

  // Textract returns a list of blocks. A block can be a line, a page, word, etc.
  // Each block also contains geometry of the detected text.
  // For more information on the Block type, see https://docs.aws.amazon.com/textract/latest/dg/API\_Block.html.
  const { Blocks } = await textractClient.send(detectDocumentTextCommand);

  // For the purpose of this example, we are only interested in words.

```

```
const extractedWords = Blocks.filter((b) => b.BlockType === "WORD").map(
  (b) => b.Text,
);

return extractedWords.join(" ");
};
```

```
import { PollyClient, SynthesizeSpeechCommand } from "@aws-sdk/client-polly";
import { S3Client } from "@aws-sdk/client-s3";
import { Upload } from "@aws-sdk/lib-storage";

/**
 * Synthesize an audio file from text.
 *
 * @param {{ bucket: string, translated_text: string, object: string }}
 * sourceDestinationConfig
 */
export const handler = async (sourceDestinationConfig) => {
  const pollyClient = new PollyClient({});

  const synthesizeSpeechCommand = new SynthesizeSpeechCommand({
    Engine: "neural",
    Text: sourceDestinationConfig.translated_text,
    VoiceId: "Ruth",
    OutputFormat: "mp3",
  });

  const { AudioStream } = await pollyClient.send(synthesizeSpeechCommand);

  const audioKey = `${sourceDestinationConfig.object}.mp3`;

  // Store the audio file in S3.
  const s3Client = new S3Client();
  const upload = new Upload({
    client: s3Client,
    params: {
      Bucket: sourceDestinationConfig.bucket,
      Key: audioKey,
      Body: AudioStream,
      ContentType: "audio/mp3",
    },
  });
};
```

```
    await upload.done();  
    return audioKey;  
};
```

```
import {  
    TranslateClient,  
    TranslateTextCommand,  
} from "@aws-sdk/client-translate";  
  
/**  
 * Translate the extracted text to English.  
 *  
 * @param {{ extracted_text: string, source_language_code: string }}  
 textAndSourceLanguage  
 */  
export const handler = async (textAndSourceLanguage) => {  
    const translateClient = new TranslateClient({});  
  
    const translateCommand = new TranslateTextCommand({  
        SourceLanguageCode: textAndSourceLanguage.source_language_code,  
        TargetLanguageCode: "en",  
        Text: textAndSourceLanguage.extracted_text,  
    });  
  
    const { TranslatedText } = await translateClient.send(translateCommand);  
  
    return { translated_text: TranslatedText };  
};
```

此範例中使用的服務

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

Ruby

SDK for Ruby

此範例應用程式會分析和存儲客戶的意見回饋卡。具體來說，它滿足了紐約市一家虛構飯店的需求。飯店以實體評論卡的形式收到賓客以各種語言撰寫的意見回饋。這些意見回饋透過 Web 用戶端上傳至應用程式。評論卡的影像上傳後，系統會執行下列步驟：

- 文字內容是使用 Amazon Textract 從影像中擷取。
- Amazon Comprehend 會決定擷取文字及其用語的情感。
- 擷取的文字內容會使用 Amazon Translate 翻譯成英文。
- Amazon Polly 會使用擷取的文字內容合成音訊檔案。

完整的應用程式可透過 AWS CDK 部署。如需原始程式碼和部署的說明，請參閱 [GitHub](#) 中的專案。

此範例中使用的服務

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

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [搭配 SDK 使用 Amazon Polly AWS](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

text-to-speech合成入門

以下程式碼範例顯示做法：

- 清除資源

Bash

AWS CLI 使用 Bash 指令碼

Note

GitHub 上提供更多範例。尋找完整範例，並了解如何在[範例開發人員教學課程](#)儲存庫中設定和執行。

```
#!/bin/bash

# Amazon Polly Getting Started Script
# This script demonstrates how to use Amazon Polly with the AWS CLI

set -euo pipefail

# Set up logging
LOG_FILE="polly-tutorial.log"
SCRIPT_DIR="$(cd "$(dirname "${BASH_SOURCE[0]}")" && pwd)"
WORK_DIR=$(mktemp -d)
trap 'cleanup_temp' EXIT

cleanup_temp() {
    rm -rf "$WORK_DIR"
}

echo "Starting Amazon Polly tutorial at $(date)" > "$LOG_FILE"

# Function to log commands and their output
log_cmd() {
    echo "Running: $1" | tee -a "$LOG_FILE"
    # Use bash array to safely handle arguments
    bash -c "$1" 2>&1 | tee -a "$LOG_FILE" || return $?
}

# Function to check for errors
check_error() {
    if echo "$1" | grep -iq "error"; then
        echo "ERROR detected in output. Exiting script." | tee -a "$LOG_FILE"
        echo "$1" | tee -a "$LOG_FILE"
        return 1
    fi
}
```

```

    fi
    return 0
}

# Function to handle errors and cleanup
handle_error() {
    local line_number=$1
    echo "Error occurred at line $line_number. Attempting cleanup..." | tee -a
"$LOG_FILE"
    cleanup
    exit 1
}

# Function to clean up resources
cleanup() {
    echo "" | tee -a "$LOG_FILE"
    echo "===== " | tee -a
"$LOG_FILE"
    echo "CLEANUP PROCESS" | tee -a "$LOG_FILE"
    echo "===== " | tee -a
"$LOG_FILE"

    # Delete lexicon if it exists
    if [[ -n "${LEXICON_NAME:-}" ]]; then
        echo "Deleting lexicon: $LEXICON_NAME" | tee -a "$LOG_FILE"
        if aws polly delete-lexicon --name "$LEXICON_NAME" 2>&1 | tee -a
"$LOG_FILE"; then
            echo "Lexicon deleted successfully." | tee -a "$LOG_FILE"
        else
            echo "Warning: Failed to delete lexicon." | tee -a "$LOG_FILE"
        fi
    fi

    # Remove audio files
    for file in output.mp3 ssm1-output.mp3 lexicon-output.mp3 example.pls; do
        if [[ -f "$file" ]]; then
            rm -f "$file"
            echo "Removed $file" | tee -a "$LOG_FILE"
        fi
    done

    echo "Cleanup complete." | tee -a "$LOG_FILE"
}

```

```
# Trap errors with line number
trap 'handle_error ${LINENO}' ERR

# Verify AWS CLI is available
if ! command -v aws &> /dev/null; then
    echo "AWS CLI is not installed. Please install it first." | tee -a
    "$LOG_FILE"
    exit 1
fi

# Verify AWS credentials are configured
if ! aws sts get-caller-identity &> /dev/null; then
    echo "AWS credentials are not configured. Please configure them first." | tee
    -a "$LOG_FILE"
    exit 1
fi

# Step 1: Verify Amazon Polly is available
echo "Step 1: Verifying Amazon Polly availability" | tee -a "$LOG_FILE"
if aws polly describe-voices --query 'Voices[0].Name' --output text &> /dev/null;
then
    echo "Amazon Polly is available. Proceeding with tutorial." | tee -a
    "$LOG_FILE"
else
    echo "Amazon Polly is not available in your AWS CLI installation or region."
    | tee -a "$LOG_FILE"
    echo "Please update your AWS CLI to the latest version or check your region."
    | tee -a "$LOG_FILE"
    exit 1
fi

# Step 2: List available voices
echo "" | tee -a "$LOG_FILE"
echo "Step 2: Listing available voices" | tee -a "$LOG_FILE"
log_cmd "aws polly describe-voices --language-code en-US --output text --query
'Voices[0:3].[Id, LanguageCode, Gender]'" || true

# Step 3: Basic text-to-speech conversion
echo "" | tee -a "$LOG_FILE"
echo "Step 3: Converting text to speech" | tee -a "$LOG_FILE"
OUTPUT_FILE="${WORK_DIR}/output.mp3"
POLLY_TEXT="Hello, welcome to Amazon Polly. This is a sample text to speech
conversion."
```

```
log_cmd "aws polly synthesize-speech --output-format mp3 --voice-id Joanna --text
'$POLLY_TEXT' '$OUTPUT_FILE'" || true

if [[ -f "$OUTPUT_FILE" ]]; then
    echo "Successfully created output.mp3 file." | tee -a "$LOG_FILE"
    echo "You can play this file with your preferred audio player." | tee -a
"$LOG_FILE"
    cp "$OUTPUT_FILE" output.mp3
else
    echo "Failed to create output.mp3 file." | tee -a "$LOG_FILE"
    exit 1
fi

# Step 4: Using SSML for enhanced speech
echo "" | tee -a "$LOG_FILE"
echo "Step 4: Using SSML for enhanced speech" | tee -a "$LOG_FILE"
SSML_OUTPUT="${WORK_DIR}/ssml-output.mp3"
SSML_TEXT='<speaK>Hello! <break time="1s"/> This is a sample of <emphasis>SSML
enhanced speech</emphasis>.</speaK>'
log_cmd "aws polly synthesize-speech --output-format mp3 --voice-id Matthew --
text-type ssml --text '$SSML_TEXT' '$SSML_OUTPUT'" || true

if [[ -f "$SSML_OUTPUT" ]]; then
    echo "Successfully created ssml-output.mp3 file." | tee -a "$LOG_FILE"
    echo "You can play this file with your preferred audio player." | tee -a
"$LOG_FILE"
    cp "$SSML_OUTPUT" ssml-output.mp3
else
    echo "Failed to create ssml-output.mp3 file." | tee -a "$LOG_FILE"
    exit 1
fi

# Step 5: Working with lexicons
echo "" | tee -a "$LOG_FILE"
echo "Step 5: Working with lexicons" | tee -a "$LOG_FILE"

# Generate a random identifier for the lexicon (max 20 chars, alphanumeric only)
LEXICON_NAME="example$(openssl rand -hex 6 | cut -c 1-10)"
echo "Using lexicon name: $LEXICON_NAME" | tee -a "$LOG_FILE"

# Create a lexicon file
echo "Creating lexicon file..." | tee -a "$LOG_FILE"
LEXICON_FILE="${WORK_DIR}/example.pls"
cat > "$LEXICON_FILE" << 'EOF'
```

```

<?xml version="1.0" encoding="UTF-8"?>
<lexicon version="1.0"
  xmlns="http://www.w3.org/2005/01/pronunciation-lexicon"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.w3.org/2005/01/pronunciation-lexicon
    http://www.w3.org/TR/2007/CR-pronunciation-lexicon-20071212/pls.xsd"
  alphabet="ipa"
  xml:lang="en-US">
  <lexeme>
    <grapheme>AWS</grapheme>
    <alias>Amazon Web Services</alias>
  </lexeme>
</lexicon>
EOF

# Upload the lexicon
echo "Uploading lexicon..." | tee -a "$LOG_FILE"
LEXICON_ARN=$(aws polly put-lexicon --name "$LEXICON_NAME" --content
  file://"${LEXICON_FILE}" --query 'LexiconArn' --output text 2>&1) || true
if [[ -n "$LEXICON_ARN" && "$LEXICON_ARN" != "" && ! "$LEXICON_ARN" =~ error ]];
then
  echo "Lexicon uploaded with ARN: $LEXICON_ARN" | tee -a "$LOG_FILE"
  aws polly tag-resource --resource-arn "$LEXICON_ARN" --tags
    Key=project,Value=doc-smith Key=tutorial,Value=amazon-polly-gs 2>&1 | tee -a
    "$LOG_FILE" || true
else
  echo "Lexicon uploaded." | tee -a "$LOG_FILE"
fi

# List available lexicons
echo "Listing available lexicons..." | tee -a "$LOG_FILE"
log_cmd "aws polly list-lexicons --output text --query 'Lexicons[*].[Name]'" ||
  true

# Get details about the lexicon
echo "Getting details about the lexicon..." | tee -a "$LOG_FILE"
log_cmd "aws polly get-lexicon --name '$LEXICON_NAME' --output text --query
  'Lexicon.Name'" || true

# Use the lexicon when synthesizing speech
echo "Using the lexicon for speech synthesis..." | tee -a "$LOG_FILE"
LEXICON_OUTPUT="${WORK_DIR}/lexicon-output.mp3"
LEXICON_TEXT="I work with AWS every day."

```

```

log_cmd "aws polly synthesize-speech --output-format mp3 --voice-id Joanna --
lexicon-names '$LEXICON_NAME' --text '$LEXICON_TEXT' '$LEXICON_OUTPUT'" || true

if [[ -f "$LEXICON_OUTPUT" ]]; then
    echo "Successfully created lexicon-output.mp3 file." | tee -a "$LOG_FILE"
    echo "You can play this file with your preferred audio player." | tee -a
"$LOG_FILE"
    cp "$LEXICON_OUTPUT" lexicon-output.mp3
else
    echo "Failed to create lexicon-output.mp3 file." | tee -a "$LOG_FILE"
    exit 1
fi

# Summary of created resources
echo "" | tee -a "$LOG_FILE"
echo "===== " | tee -a
"$LOG_FILE"
echo "TUTORIAL SUMMARY" | tee -a "$LOG_FILE"
echo "===== " | tee -a
"$LOG_FILE"
echo "Created resources:" | tee -a "$LOG_FILE"
echo "1. Lexicon: $LEXICON_NAME" | tee -a "$LOG_FILE"
echo "2. Audio files:" | tee -a "$LOG_FILE"
echo "   - output.mp3" | tee -a "$LOG_FILE"
echo "   - ssm1-output.mp3" | tee -a "$LOG_FILE"
echo "   - lexicon-output.mp3" | tee -a "$LOG_FILE"
echo "" | tee -a "$LOG_FILE"

# Cleanup with auto-confirmation
echo "" | tee -a "$LOG_FILE"
echo "===== " | tee -a
"$LOG_FILE"
echo "CLEANUP CONFIRMATION" | tee -a "$LOG_FILE"
echo "===== " | tee -a
"$LOG_FILE"
echo "Cleaning up all created resources..." | tee -a "$LOG_FILE"
cleanup

echo "" | tee -a "$LOG_FILE"
echo "Tutorial completed successfully!" | tee -a "$LOG_FILE"
echo "Log file: $LOG_FILE" | tee -a "$LOG_FILE"

```

- 如需 API 詳細資訊，請參閱《AWS CLI 命令參考》中的下列主題。

- [DeleteLexicon](#)
- [DescribeVoices](#)
- [GetLexicon](#)
- [Help \(說明\)](#)
- [ListLexicons](#)
- [PutLexicon](#)
- [SynthesizeSpeech](#)

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 [搭配 SDK 使用 Amazon Polly AWS](#)。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

Amazon Polly 的安全性

的雲端安全性 AWS 是最高優先順序。身為 AWS 客戶，您可以受益於資料中心和網路架構，這些架構是為了滿足最安全敏感組織的需求而建置。

安全性是 AWS 與您之間共同責任。[共同責任模式](#)將其描述為雲端的安全性，和雲端中的安全性：

- 雲端的安全性 – AWS 負責保護在 AWS Cloud 中執行 AWS 服務的基礎設施。AWS 也為您提供可安全使用的服務。在[AWS 合規計畫](#)中，第三方稽核人員會定期測試和驗證我們安全的有效性。若要了解適用於 Amazon Polly 的合規計畫，請參閱[AWS 合規計畫的服務範圍](#)。
- 雲端的安全性 – 您的責任取決於您使用 AWS 的服務。您也必須對其他因素負責，包括資料的機密性、您公司的要求和適用法律和法規。

本文件可協助您了解如何在使用 Amazon Polly 時套用共同責任模型。下列主題說明如何設定 Amazon Polly 以符合您的安全與合規目標。您也會了解如何使用其他 AWS 服務來協助您監控和保護 Amazon Polly 資源。

主題

- [Amazon Polly 中的資料保護](#)
- [Amazon Polly 中的 Identity and Access Management](#)
- [在 Amazon Polly 中記錄和監控](#)
- [Amazon Polly 的合規驗證](#)
- [Amazon Polly 中的彈性](#)
- [Amazon Polly 中的基礎設施安全](#)
- [Amazon Polly 的安全最佳實務](#)
- [搭配界面 VPC 端點使用 Amazon Polly](#)

Amazon Polly 中的資料保護

Amazon Polly 遵循 AWS [共同責任模型](#)，其中包含資料保護的法規和指導方針。AWS 負責保護執行所有 AWS 服務的全球基礎設施。AWS 會維持對此基礎設施上託管資料的控制，包括用於處理客戶內容和個人資料的安全組態控制。身為資料控制者或資料處理者的 AWS 客戶和 APN 合作夥伴，會對其放置在 AWS 雲端中的任何個人資料負責。

基於資料保護目的，建議您保護 AWS 帳戶登入資料，並使用 AWS Identity and Access Management (IAM) 設定個別使用者，以便每個使用者只獲得完成其任務所需的許可。我們也建議您採用下列方式保護資料：

- 每個帳戶均要使用多重要素驗證 (MFA)。
- 使用 SSL/TLS 與 AWS 資源通訊。
- 使用設定 API 和使用使用者活動記錄 AWS CloudTrail。
- 使用 AWS 加密解決方案，以及服務中的所有 AWS 預設安全控制。

我們強烈建議您絕對不要將客戶帳戶號碼等敏感的識別資訊，放在自由格式的欄位中，例如 Name (名稱) 欄位。這包括當您使用 Amazon Polly 或使用主控台 AWS CLI、API 或 AWS SDKs 的其他 AWS 服務時。您在 Amazon Polly 或其他服務中輸入的任何資料都可能被選入診斷日誌中。當您提供外部伺服器的 URL 時，請勿在驗證您對該伺服器請求的 URL 中包含憑證資訊。

如需關於資料保護的詳細資訊，請參閱 AWS 安全部落格上的 [AWS 共同責任模型和歐盟《一般資料保護規範》\(GDPR\)](#) 部落格文章。

靜態加密

Amazon Polly 語音合成的輸出可以儲存在您自己的系統上。您也可以呼叫 Amazon Polly，然後使用您選擇的任何加密金鑰來加密檔案，並將其存放在 Amazon Simple Storage Service (Amazon S3) 或其他安全儲存中。Amazon Polly [the section called “SynthesizeSpeech”](#) 操作是無狀態操作，且與客戶身分沒有關聯。您無法稍後從 Amazon Polly 擷取它。

傳輸中加密

傳輸時，所有文字提交都會受到 TLS 的保護。Amazon Polly 不會保留文字提交的內容。

網際網路流量隱私權

透過 AWS 主控台、CLI 或 SDKs 存取 Amazon Polly。通訊會針對驗證和完整性的機密性和 [數位簽章](#)，利用傳輸層安全性 (TLS) 工作階段加密。

Amazon Polly 中的 Identity and Access Management

AWS Identity and Access Management (IAM) 是 AWS 服務，可協助管理員安全地控制對 AWS 資源的存取。IAM 管理員可控制誰可以進行身分驗證（登入）和授權（具有許可），以使用 Amazon Polly 資源。IAM 是您可以免費使用 AWS 服務的。

主題

- [目標對象](#)
- [使用身分驗證](#)
- [使用政策管理存取權](#)
- [Amazon Polly 如何與 IAM 搭配使用](#)
- [Amazon Polly 的身分型政策範例](#)
- [Amazon Polly API 許可：動作、許可和資源參考](#)
- [對 Amazon Polly 身分和存取進行故障診斷](#)

目標對象

使用方式 AWS Identity and Access Management (IAM) 會根據您的角色而有所不同：

- 服務使用者 — 若無法存取某些功能，請向管理員申請所需許可 (請參閱 [對 Amazon Polly 身分和存取進行故障診斷](#))
- 服務管理員 — 負責設定使用者存取權並提交相關許可請求 (請參閱 [Amazon Polly 如何與 IAM 搭配使用](#))
- IAM 管理員 — 撰寫政策以管理存取控制 (請參閱 [Amazon Polly 的身分型政策範例](#))

使用身分驗證

身分驗證是您 AWS 使用身分憑證登入的方式。您必須以 AWS 帳戶根使用者、IAM 使用者或擔任 IAM 角色身分進行身分驗證。

您可以使用身分來源的登入資料，例如 AWS IAM Identity Center (IAM Identity Center)、單一登入身分驗證或 Google/Facebook 登入資料，以聯合身分的形式登入。如需有關登入的詳細資訊，請參閱《AWS 登入 使用者指南》中的[如何登入您的 AWS 帳戶](#)。

對於程式設計存取，AWS 提供 SDK 和 CLI 以密碼編譯方式簽署請求。如需詳細資訊，請參閱《IAM 使用者指南》中的[API 請求的AWS 第 4 版簽署程序](#)。

AWS 帳戶 根使用者

當您建立時 AWS 帳戶，您會從一個名為 AWS 帳戶 theroot 使用者的登入身分開始，該身分可完整存取所有 AWS 服務和資源。強烈建議不要使用根使用者來執行日常任務。有關需要根使用者憑證的任務，請參閱《IAM 使用者指南》中的[需要根使用者憑證的任務](#)。

聯合身分

最佳實務是要求人類使用者使用聯合身分提供者，以 AWS 服務 使用臨時憑證存取。

聯合身分是您企業目錄、Web 身分提供者的使用者，或是 AWS 服務 使用身分來源的憑證 Directory Service 存取的使用者。聯合身分會擔任角色，而該角色會提供臨時憑證。

若需集中化管理存取權限，建議使用 AWS IAM Identity Center。如需詳細資訊，請參閱 AWS IAM Identity Center 使用者指南中的[什麼是 IAM Identity Center？](#)。

IAM 使用者和群組

IAM 使用者https://docs.aws.amazon.com/IAM/latest/UserGuide/id_users.html是一種身分具備單人或應用程式的特定許可權。建議以臨時憑證取代具備長期憑證的 IAM 使用者。如需詳細資訊，請參閱《IAM 使用者指南》中的[要求人類使用者使用聯合身分提供者來 AWS 使用臨時憑證存取](#)。

[IAM 群組](#)會指定 IAM 使用者集合，使管理大量使用者的許可權更加輕鬆。如需詳細資訊，請參閱《IAM 使用者指南》中的[IAM 使用者的使用案例](#)。

IAM 角色

IAM 角色https://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles.html的身分具有特定許可權，其可以提供臨時憑證。您可以透過[從使用者切換到 IAM 角色（主控台）](#)或呼叫 AWS CLI 或 AWS API 操作來擔任角色。如需詳細資訊，請參閱《IAM 使用者指南》中的[擔任角色的方法](#)。

IAM 角色適用於聯合身分使用者存取、臨時 IAM 使用者許可、跨帳戶存取權與跨服務存取，以及在 Amazon EC2 執行的應用程式。如需詳細資訊，請參閱《IAM 使用者指南》中的[IAM 中的快帳戶資源存取](#)。

使用政策管理存取權

您可以透過建立政策並將其連接到身分或資源 AWS 來控制 AWS 中的存取。政策定義與身分或資源相關聯的許可。當委託人提出請求時 AWS，會評估這些政策。大多數政策會以 JSON 文件 AWS 的形式存放在 中。如需進一步了解 JSON 政策文件，請參閱《IAM 使用者指南》中的[JSON 政策概觀](#)。

管理員會使用政策，透過定義哪些主體可在哪些條件下對哪些資源執行動作，以指定可存取的範圍。

預設情況下，使用者和角色沒有許可。IAM 管理員會建立 IAM 政策並將其新增至角色，供使用者後續擔任。IAM 政策定義動作的許可，無論採用何種方式執行。

身分型政策

身分型政策是附加至身分 (使用者、使用者群組或角色) 的 JSON 許可政策文件。這類政策控制身分可對哪些資源執行哪些動作，以及適用的條件。如需了解如何建立身分型政策，請參閱《IAM 使用者指南》中的[透過客戶管理政策定義自訂 IAM 許可](#)。

身分型政策可分為內嵌政策 (直接內嵌於單一身分) 與受管政策 (可附加至多個身分的獨立政策)。如需了解如何在受管政策及內嵌政策之間做選擇，請參閱《IAM 使用者指南》中的[在受管政策與內嵌政策之間選擇](#)。

資源型政策

資源型政策是附加到資源的 JSON 政策文件。範例包括 IAM 角色信任政策與 Amazon S3 儲存貯體政策。在支援資源型政策的服務中，服務管理員可以使用它們來控制對特定資源的存取權限。您必須在資源型政策中[指定主體](#)。

資源型政策是位於該服務中的內嵌政策。您無法在資源型政策中使用來自 IAM 的 AWS 受管政策。

其他政策類型

AWS 支援其他政策類型，可設定更多常見政策類型授予的最大許可：

- 許可界限 — 設定身分型政策可授與 IAM 實體的最大許可。如需詳細資訊，請參閱《IAM 使用者指南》中的[IAM 實體許可界限](#)。
- 服務控制政策 (SCP) — 為 AWS Organizations 中的組織或組織單位指定最大許可。如需詳細資訊，請參閱《AWS Organizations 使用者指南》中的[服務控制政策](#)。
- 資源控制政策 (RCP) — 設定您帳戶中資源可用許可的上限。如需詳細資訊，請參閱《AWS Organizations 使用者指南》中的[資源控制政策 \(RCP\)](#)。
- 工作階段政策 — 在以程式設計方式為角色或聯合身分使用者建立臨時工作階段時，以參數形式傳遞的進階政策。如需詳細資訊，請參《IAM 使用者指南》中的[工作階段政策](#)。

多種政策類型

當多種類型的政策適用於請求時，產生的許可會更複雜而無法理解。若要了解如何 AWS 在涉及多個政策類型時決定是否允許請求，請參閱《IAM 使用者指南》中的[政策評估邏輯](#)。

Amazon Polly 如何與 IAM 搭配使用

在您使用 IAM 管理 Amazon Polly 的存取權之前，請先了解哪些 IAM 功能可與 Amazon Polly 搭配使用。

您可以搭配 Amazon Polly 使用的 IAM 功能

IAM 功能	Amazon Polly 支援
身分型政策	是
資源型政策	否
政策動作	是
政策資源	是
政策條件索引鍵 (服務特定)	否
ACL	否
ABAC(政策中的標籤)	否
臨時憑證	是
轉送 Amazon Polly 的存取工作階段 (FAS)	是
服務角色	否
服務連結角色	否

若要全面了解 Amazon Polly 和其他 AWS 服務如何與大多數 IAM 功能搭配使用，請參閱《IAM 使用者指南》中的[AWS 與 IAM 搭配使用的服務](#)。

Amazon Polly 的身分型政策

支援身分型政策：是

身分型政策是可以附加到身分 (例如 IAM 使用者、使用者群組或角色) 的 JSON 許可政策文件。這些政策可控制身分在何種條件下能對哪些資源執行哪些動作。如需了解如何建立身分型政策，請參閱《IAM 使用者指南》中的[透過客戶管理政策定義自訂 IAM 許可](#)。

使用 IAM 身分型政策，您可以指定允許或拒絕的動作和資源，以及在何種條件下允許或拒絕動作。如要了解您在 JSON 政策中使用的所有元素，請參閱《IAM 使用者指南》中的[IAM JSON 政策元素參考](#)。

Amazon Polly 的身分型政策範例

若要檢視 Amazon Polly 身分型政策的範例，請參閱 [Amazon Polly 的身分型政策範例](#)。

Amazon Polly 內的資源型政策

支援資源型政策：否

資源型政策是附加到資源的 JSON 政策文件。資源型政策的最常見範例是 IAM 角色信任政策和 Amazon S3 儲存貯體政策。在支援資源型政策的服務中，服務管理員可以使用它們來控制對特定資源的存取權限。對於附加政策的資源，政策會定義指定的主體可以對該資源執行的動作以及在何種條件下執行的動作。您必須在資源型政策中[指定主體](#)。委託人可以包含帳戶、使用者、角色、聯合身分使用者或 AWS 服務。

如需啟用跨帳戶存取權，您可以在其他帳戶內指定所有帳戶或 IAM 實體作為資源型政策的主體。如需詳細資訊，請參閱《IAM 使用者指南》中的 [IAM 中的快帳戶資源存取](#)。

Amazon Polly 的政策動作

支援政策動作：是

管理員可以使用 AWS JSON 政策來指定誰可以存取內容。也就是說，哪個主體在什麼條件下可以對什麼資源執行哪些動作。

JSON 政策的 Action 元素描述您可以用來允許或拒絕政策中存取的動作。政策會使用動作來授予執行相關聯動作的許可。

若要查看 Amazon Polly 動作的清單，請參閱《服務授權參考》中的 [Amazon Polly 定義的動作](#)。

Amazon Polly 中的政策動作在動作之前使用下列字首：

```
polly
```

若要在單一陳述式中指定多個動作，請用逗號分隔。

```
"Action": [  
    "polly:action1",  
    "polly:action2"  
]
```

若要檢視 Amazon Polly 身分型政策的範例，請參閱 [Amazon Polly 的身分型政策範例](#)。

Amazon Polly 的政策資源

支援政策資源：是

管理員可以使用 AWS JSON 政策來指定誰可以存取內容。也就是說，哪個主體在什麼條件下可以對什麼資源執行哪些動作。

Resource JSON 政策元素可指定要套用動作的物件。最佳實務是使用其 [Amazon Resource Name \(ARN\)](#) 來指定資源。若動作不支援資源層級許可，使用萬用字元 (*) 表示該陳述式適用於所有資源。

```
"Resource": "*"
```

若要查看 Amazon Polly 資源類型及其 ARNs，請參閱《服務授權參考》中的 [Amazon Polly 定義的資源](#)。若要了解您可以使用哪些動作指定每個資源的 ARN，請參閱 [Amazon Polly 定義的動作](#)。

若要檢視 Amazon Polly 身分型政策的範例，請參閱 [Amazon Polly 的身分型政策範例](#)。

Amazon Polly 的政策條件索引鍵

支援服務特定政策條件金鑰：否

管理員可以使用 AWS JSON 政策來指定誰可以存取內容。也就是說，哪個主體在什麼條件下可以對什麼資源執行哪些動作。

Condition 元素會根據定義的條件，指定陳述式的執行時機。您可以建立使用 [條件運算子](#) 的條件運算式 (例如等於或小於)，來比對政策中的條件和請求中的值。若要查看所有 AWS 全域條件索引鍵，請參閱《IAM 使用者指南》中的 [AWS 全域條件內容索引鍵](#)。

若要查看 Amazon Polly 條件索引鍵的清單，請參閱《服務授權參考》中的 [Amazon Polly 條件索引鍵](#)。若要了解您可以使用條件金鑰的動作和資源，請參閱 [Amazon Polly 定義的動作](#)。

若要檢視 Amazon Polly 身分型政策的範例，請參閱 [Amazon Polly 的身分型政策範例](#)。

Amazon Polly 中的 ACLs

支援 ACL：否

存取控制清單 (ACL) 可控制哪些主體 (帳戶成員、使用者或角色) 擁有存取某資源的許可。ACL 類似於資源型政策，但它們不使用 JSON 政策文件格式。

ABAC 與 Amazon Polly

支援 ABAC (政策中的標籤)：否

屬性型存取控制 (ABAC) 是一種授權策略，依據稱為標籤的屬性來定義許可。您可以將標籤連接至 IAM 實體 AWS 和資源，然後設計 ABAC 政策，以便在委託人的標籤符合資源上的標籤時允許操作。

如需根據標籤控制存取，請使用 `aws:ResourceTag/key-name`、`aws:RequestTag/key-name` 或 `aws:TagKeys` 條件索引鍵，在政策的[條件元素](#)中，提供標籤資訊。

如果服務支援每個資源類型的全部三個條件金鑰，則對該服務而言，值為 Yes。如果服務僅支援某些資源類型的全部三個條件金鑰，則值為 Partial。

如需 ABAC 的詳細資訊，請參閱《IAM 使用者指南》中的[使用 ABAC 授權定義許可](#)。如要查看含有設定 ABAC 步驟的教學課程，請參閱《IAM 使用者指南》中的[使用屬性型存取控制 \(ABAC\)](#)。

搭配 Amazon Polly 使用臨時登入資料

支援臨時憑證：是

臨時登入資料提供對 AWS 資源的短期存取，並在您使用聯合或切換角色時自動建立。AWS 建議您動態產生臨時登入資料，而不是使用長期存取金鑰。如需詳細資訊，請參閱《IAM 使用者指南》中的[IAM 中的臨時安全憑證與可與 IAM 搭配運作的 AWS 服務](#)。

Amazon Polly 的跨服務轉送存取工作階段 (FAS)

支援轉寄存取工作階段 (FAS)：是

轉送存取工作階段 (FAS) 使用呼叫的委託人許可 AWS 服務，結合 AWS 服務請求向下游服務提出請求。如需提出 FAS 請求時的政策詳細資訊，請參閱[轉發存取工作階段](#)。

Amazon Polly 的服務角色

支援服務角色：否

服務角色是服務擔任的[IAM 角色](#)，可代您執行動作。IAM 管理員可以從 IAM 內建立、修改和刪除服務角色。如需詳細資訊，請參閱《IAM 使用者指南》中的[建立角色以委派許可給 AWS 服務](#)。

Warning

變更服務角色的許可可能會中斷 Amazon Polly 功能。只有在 Amazon Polly 提供指引時，才能編輯服務角色。

Amazon Polly 的服務連結角色

支援服務連結角色：否

服務連結角色是連結至的一種服務角色 AWS 服務。服務可以擔任代表您執行動作的角色。服務連結角色會出現在您的中 AWS 帳戶，並由服務擁有。IAM 管理員可以檢視，但不能編輯服務連結角色的許可。

如需建立或管理服务連結角色的詳細資訊，請參閱[可搭配 IAM 運作的AWS 服務](#)。在資料表中尋找服務，其中包含服務連結角色欄中的 Yes。選擇是連結，以檢視該服務的服務連結角色文件。

Amazon Polly IAM 角色

您可以將身分型許可政策連接至 IAM 角色，以授予跨帳戶許可。例如，帳戶 A 中的管理員可以建立角色，將跨帳戶許可授予另一個 AWS 帳戶（例如帳戶 B）或 AWS 服務，如下所示：

1. 帳戶 A 管理員建立 IAM 角色，並將許可政策連接到可授與帳戶 A 中資源許可的角色。
2. 帳戶 A 管理員會將信任政策連接至將帳戶 B 識別為可擔任角色之主體的角色。
3. 帳戶 B 管理員接著可以將擔任該角色的許可委派給帳戶 B 中的任何使用者。這樣做可讓帳戶 B 中的使用者在帳戶 A 中建立或存取資源。如果您想要授予擔任該角色的服務許可，AWS 則信任政策中的委託人也可以是 AWS 服務委託人。

如需使用 IAM 來委派許可的相關資訊，請參閱《IAM 使用者指南》中的[存取管理](#)。

以下是範例政策，授予許可可以放置和取得語彙，以及列出目前可用的語彙。

Amazon Polly 支援資源層級動作的身分型政策。在某些情況下，資源可能會受到 ARN 的限制。這對於 SynthesizeSpeech、StartSpeechSynthesisTask、PutLexicon、GetLexicon 和 DeleteLexicon 操作都是如此。在這些情況下，Resource 值是由 ARN 指示。例如：`arn:aws:polly:us-east-2:account-id:lexicon/*` 做為 Resource 值對 us-east-2 區域內所有擁有的語彙指定許可。

不過，並非所有操作都使用 ARNs。DescribeVoices、ListLexiconsGetSpeechSynthesisTasks 和 ListSpeechSynthesisTasks 操作都是這種情況。

如需使用者、群組、角色和許可的詳細資訊，請參閱《IAM 使用者指南》中的[身分 \(使用者、群組和角色\)](#)。

Amazon Polly 的身分型政策範例

根據預設，使用者和角色沒有建立或修改 Amazon Polly 資源的許可。若要授予使用者對其所需資源執行動作的許可，IAM 管理員可以建立 IAM 政策。

如需了解如何使用這些範例 JSON 政策文件建立 IAM 身分型政策，請參閱《IAM 使用者指南》中的[建立 IAM 政策 \(主控台\)](#)。

如需 Amazon Polly 定義的動作和資源類型的詳細資訊，包括每種資源類型的 ARNs 格式，請參閱《服務授權參考》中的[Amazon Polly 的動作、資源和條件索引鍵](#)。

主題

- [政策最佳實務](#)
- [使用 Amazon Polly 主控台](#)
- [允許使用者檢視他們自己的許可](#)
- [AWS Amazon Polly 的 受管（預先定義）政策](#)
- [Amazon Polly 更新 AWS 受管政策](#)
- [客戶受管政策範例](#)

政策最佳實務

身分型政策會判斷您帳戶中的某個人員是否可以建立、存取或刪除 Amazon Polly 資源。這些動作可能會讓您的 AWS 帳戶產生費用。當您建立或編輯身分型政策時，請遵循下列準則及建議事項：

- 開始使用 AWS 受管政策並轉向最低權限許可 – 若要開始將許可授予您的使用者和工作負載，請使用將許可授予許多常見使用案例的 AWS 受管政策。它們可在您的 中使用 AWS 帳戶。我們建議您定義特定於使用案例 AWS 的客戶受管政策，以進一步減少許可。如需更多資訊，請參閱《IAM 使用者指南》中的[AWS 受管政策](#)或[任務職能的AWS 受管政策](#)。
- 套用最低權限許可 – 設定 IAM 政策的許可時，請僅授予執行任務所需的許可。為實現此目的，您可以定義在特定條件下可以對特定資源採取的動作，這也稱為最低權限許可。如需使用 IAM 套用許可的更多相關資訊，請參閱《IAM 使用者指南》中的[IAM 中的政策和許可](#)。
- 使用 IAM 政策中的條件進一步限制存取權 – 您可以將條件新增至政策，以限制動作和資源的存取。例如，您可以撰寫政策條件，指定必須使用 SSL 傳送所有請求。如果透過特定 例如 使用服務動作 AWS 服務，您也可以使用條件來授予其存取權 CloudFormation。如需詳細資訊，請參閱《IAM 使用者指南》中的[IAM JSON 政策元素：條件](#)。

- 使用 IAM Access Analyzer 驗證 IAM 政策，確保許可安全且可正常運作 – IAM Access Analyzer 驗證新政策和現有政策，確保這些政策遵從 IAM 政策語言 (JSON) 和 IAM 最佳實務。IAM Access Analyzer 提供 100 多項政策檢查及切實可行的建議，可協助您撰寫安全且實用的政策。如需詳細資訊，請參閱《IAM 使用者指南》中的[使用 IAM Access Analyzer 驗證政策](#)。
- 需要多重要素驗證 (MFA) – 如果您的案例需要 IAM 使用者或中的根使用者 AWS 帳戶，請開啟 MFA 以提高安全性。如需在呼叫 API 操作時請求 MFA，請將 MFA 條件新增至您的政策。如需詳細資訊，請參閱《IAM 使用者指南》中的[透過 MFA 的安全 API 存取](#)。

如需 IAM 中最佳實務的相關資訊，請參閱《IAM 使用者指南》中的 [IAM 安全最佳實務](#)。

使用 Amazon Polly 主控台

若要存取 Amazon Polly 主控台，您必須擁有一組最低許可。這些許可必須允許您列出和檢視中 Amazon Polly 資源的詳細資訊 AWS 帳戶。如果您建立比最基本必要許可更嚴格的身分型政策，則對於具有該政策的實體 (使用者或角色) 而言，主控台就無法如預期運作。

對於僅呼叫 AWS CLI 或 AWS API 的使用者，您不需要允許最低主控台許可。反之，只需允許存取符合他們嘗試執行之 API 操作的動作就可以了。

為了確保使用者和角色仍然可以使用 Amazon Polly 主控台，請將 Amazon Polly *ConsoleAccess* 或 *ReadOnly* AWS 受管政策連接到實體。如需詳細資訊，請參閱《IAM 使用者指南》中的[新增許可到使用者](#)。

若要使用 Amazon Polly 主控台，請將許可授予所有 Amazon Polly APIs。不需要額外的許可。若要取得完整主控台功能，您可以使用下列政策：

允許使用者檢視他們自己的許可

此範例會示範如何建立政策，允許 IAM 使用者檢視附加到他們使用者身分的內嵌及受管政策。此政策包含在主控台或使用或 AWS CLI AWS API 以程式設計方式完成此動作的許可。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ViewOwnUserInfo",
      "Effect": "Allow",
      "Action": [
        "iam:GetUserPolicy",
```

```

        "iam:ListGroupsWithUser",
        "iam:ListAttachedUserPolicies",
        "iam:ListUserPolicies",
        "iam:GetUser"
    ],
    "Resource": ["arn:aws:iam::*:user/${aws:username}"]
},
{
    "Sid": "NavigateInConsole",
    "Effect": "Allow",
    "Action": [
        "iam:GetGroupPolicy",
        "iam:GetPolicyVersion",
        "iam:GetPolicy",
        "iam:ListAttachedGroupPolicies",
        "iam:ListGroupPolicies",
        "iam:ListPolicyVersions",
        "iam:ListPolicies",
        "iam:ListUsers"
    ],
    "Resource": "*"
}
]
}

```

AWS Amazon Polly 的 受管（預先定義）政策

AWS 提供由 建立和管理的獨立 IAM 政策，以解決許多常見的使用案例 AWS。這些 AWS 受管政策會授予常見使用案例的必要許可，讓您不必調查需要哪些許可。如需詳細資訊，請參閱《IAM 使用者指南》中的 [AWS 受管政策](#)。

下列 AWS 受管政策是 Amazon Polly 特有的，您可以連接到您帳戶中的使用者：

- **AmazonPollyReadOnlyAccess** – 授予資源的唯讀存取權、允許列出語彙、擷取語彙、列出可用的語音和合成語音（包括將語彙套用至合成語音）。

若要檢視政策的詳細資訊，包括最新版本的 JSON 政策文件，請參閱《AWS 受管政策參考指南》中的 [AmazonPollyReadOnlyAccess](#)。

- **AmazonPollyFullAccess** – 贊助資源的完整存取權和所有支援的操作。

若要檢視政策的詳細資訊，包括最新版本的 JSON 政策文件，請參閱《AWS 受管政策參考指南》中的 [AmazonPollyFullAccess](#)。

Note

您可以登入 IAM 主控台並在該處搜尋特定政策，來檢閱這些許可政策。

您也可以建立自己的自訂 IAM 政策，以允許 Amazon Polly 動作和資源的許可。您可以將這些自訂政策連接至需要這些許可的 IAM 使用者或群組。

Amazon Polly 更新 AWS 受管政策

檢視自此服務開始追蹤 Amazon Polly AWS 受管政策更新以來的詳細資訊。如需此頁面變更的自動提醒，請訂閱 Amazon Polly 文件歷史記錄頁面上的 RSS 摘要。

Amazon Polly 更新 AWS 受管政策

政策	變更	日期
AmazonPollyReadOnlyAccess	更新 受管政策 - 新增polly:StartSpeechSynthesisStream 支援雙向串流語音合成功的 許可。	2026 年 3 月 19 日

客戶受管政策範例

在本節中，您可以找到授予各種 Amazon Polly 動作許可的使用者政策範例。當您使用 AWS SDKs 或時，這些政策即可運作 AWS CLI。當您使用主控台時，請將許可授予所有 Amazon Polly APIs。

Note

所有範例都是使用 us-east-2 區域，並且包含虛構帳戶 ID。

範例

- [範例 1：允許所有 Amazon Polly 動作](#)
- [範例 2：允許 DeleteLexicon 以外的所有 Amazon Polly 動作](#)
- [範例 3：允許 DeleteLexicon](#)
- [範例 4：允許在指定區域中刪除語彙](#)
- [範例 5：允許指定語彙的 DeleteLexicon](#)

範例 1：允許所有 Amazon Polly 動作

在您註冊 (請參閱 [Amazon Polly 入門](#)) 之後，您建立一個管理員使用者來管理您的帳戶，包括建立使用者和管理其許可。

您可以建立具有所有 Amazon Polly 動作許可的使用者。將此使用者視為使用 Amazon Polly 的服務特定管理員。您可以將以下許可政策附加到此使用者。

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Sid": "AllowAllPollyActions",
    "Effect": "Allow",
    "Action": [
      "polly:*"
    ],
    "Resource": "*"
  }]
}
```

範例 2：允許 DeleteLexicon 以外的所有 Amazon Polly 動作

下列許可政策授予使用者許可可以執行 DeleteLexicon 以外的所有動作，具有刪除在所有區域明確拒絕的許可。

範例 3：允許 DeleteLexicon

以下許可政策授予使用者許可可以刪除您擁有的任何語彙，無論專案或其所在區域。

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Sid": "AllowDeleteLexicon",
    "Effect": "Allow",
    "Action": [
      "polly:DeleteLexicon"
    ],
    "Resource": "*"
  }]
}
```

```
    "Resource": "*"
  }
]
}
```

範例 4：允許在指定區域中刪除語彙

以下許可政策授予使用者許可，以刪除任何您在單一區域 (此案例為 us-east-2) 中所擁有專案中的任何語彙，無論專案或其所在區域。

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Sid": "AllowDeleteSpecifiedRegion",
    "Effect": "Allow",
    "Action": [
      "polly:DeleteLexicon"],
    "Resource": "arn:aws:polly:us-east-2:111122223333:lexicon/*"
  ]
}
```

範例 5：允許指定語彙的 DeleteLexicon

以下許可政策授予使用者許可，以刪除任何您在特定區域 (此案例為 us-east-2) 中所擁有的特定語彙 (此案例為 myLexicon)。

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Sid": "AllowDeleteForSpecifiedLexicon",
    "Effect": "Allow",
    "Action": [
      "polly:DeleteLexicon"],
    "Resource": "arn:aws:polly:us-east-2:111122223333:lexicon/myLexicon"
  ]
}
```

```
}  
]  
}
```

Amazon Polly API 許可：動作、許可和資源參考

當您設定可連接到 IAM 身分的許可政策（以身分為基礎的政策）時，您可以使用下表清單做為參考。包含每個 Amazon Polly API 操作、您可以授予執行動作許可的對應動作，以及您可以授予許可 AWS 的資源。您在政策的 Action 欄位中指定動作，然後在政策的 Resource 欄位中指定資源值。

您可以在 Amazon Polly 政策中使用 AWS 整體條件金鑰來表達條件。如需 AWS 全系列金鑰的完整清單，請參閱《IAM 使用者指南》中的[可用金鑰](#)。

Note

若要指定動作，請使用後接 API 操作名稱的 polly 字首 (例如，polly:GetLexicon)。

Amazon Polly 支援資源層級動作的身分型政策。因此，Resource 值是由 ARN 指示。例如：arn:aws:polly:*us-east-2*:*account-id*:lexicon/* 做為 Resource 值對 us-east-2 區域內所有擁有的語彙指定許可。

由於 Amazon Polly 不支援資源層級動作的許可，因此大多數政策會指定萬用字元 (*) 做為 Resource 值。不過，如果您必須限制此萬用字元可以適當的 ARN 取代的許可為特定區域：arn:aws:polly:*region*:*account-id*:lexicon/*。

Amazon Polly API 和動作的必要許可

API 作業：[DeleteLexicon](#)

所需許可 (API 動作)：polly:DeleteLexicon

資源：arn:aws:polly:*region*:*account-id*:lexicon/*LexiconName*

API 作業：[DescribeVoices](#)

所需許可 (API 動作)：polly:DescribeVoices

資源：arn:aws:polly:*region*:*account-id*:lexicon/*voice-name*

API 作業：[GetLexicon](#)

所需許可 (API 動作)：polly:GetLexicon

資源：arn:aws:polly:*region*:*account-id*:lexicon/*voice-name*

API 作業：[ListLexicons](#)

所需許可 (API 動作)：polly:ListLexicons

資源：arn:aws:polly:*region*:*account-id*:lexicon/*

API 作業：[PutLexicon](#)

所需許可 (API 動作)：polly:ListLexicons

資源：*

API 作業：[SynthesizeSpeech](#)

所需許可 (API 動作)：polly:SynthesizeSpeech

資源：*

對 Amazon Polly 身分和存取進行故障診斷

使用以下資訊來協助您診斷和修正使用 Amazon Polly 和 IAM 時可能遇到的常見問題。

主題

- [我無權在 Amazon Polly 中執行動作](#)
- [我未獲得執行 iam:PassRole 的授權](#)
- [我想要允許以外的人員 AWS 帳戶 存取我的 Amazon Polly 資源](#)

我無權在 Amazon Polly 中執行動作

如果您收到錯誤，告知您未獲授權執行動作，您的政策必須更新，允許您執行動作。

下列範例錯誤會在mateojackson IAM 使用者嘗試使用主控台檢視一個虛構 *my-example-widget* 資源的詳細資訊，但卻無虛構 polly:*GetWidget* 許可時發生。

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
polly:GetWidget on resource: my-example-widget
```

在此情況下，必須更新 `mateojackson` 使用者的政策，允許使用 `polly:GetWidget` 動作存取 `my-example-widget` 資源。

如果您需要協助，請聯絡您的 AWS 管理員。您的管理員提供您的簽署憑證。

我未獲得執行 `iam:PassRole` 的授權

如果您收到錯誤，告知您無權執行 `iam:PassRole` 動作，您的政策必須更新，以允許您將角色傳遞給 Amazon Polly。

有些 AWS 服務可讓您將現有角色傳遞給該服務，而不是建立新的服務角色或服務連結角色。如需執行此作業，您必須擁有將角色傳遞至該服務的許可。

當名為 `marymajor` 的 IAM 使用者嘗試使用主控台在 Amazon Polly 中執行動作時，會發生下列範例錯誤。但是，動作請求服務具備服務角色授予的許可。Mary 沒有將角色傳遞給服務的許可。

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
iam:PassRole
```

在這種情況下，Mary 的政策必須更新，允許她執行 `iam:PassRole` 動作。

如果您需要協助，請聯絡您的 AWS 管理員。您的管理員提供您的簽署憑證。

我想要允許以外的人員 AWS 帳戶 存取我的 Amazon Polly 資源

您可以建立一個角色，讓其他帳戶中的使用者或您組織外部的人員存取您的資源。您可以指定要允許哪些信任物件取得該角色。針對支援基於資源的政策或存取控制清單 (ACL) 的服務，您可以使用那些政策來授予人員存取您的資源的許可。

如需進一步了解，請參閱以下內容：

- 若要了解 Amazon Polly 是否支援這些功能，請參閱 [Amazon Polly 如何與 IAM 搭配使用](#)。
- 若要了解如何在您擁有 AWS 帳戶 的資源之間提供存取權，請參閱《[IAM 使用者指南](#)》中的 [在您擁有 AWS 帳戶 的另一個 中提供存取權給 IAM 使用者](#)。
- 若要了解如何將資源的存取權提供給第三方 AWS 帳戶，請參閱《[IAM 使用者指南](#)》中的 [將存取權提供給第三方 AWS 帳戶 擁有](#)。
- 如需了解如何透過聯合身分提供存取權，請參閱《[IAM 使用者指南](#)》中的 [將存取權提供給在外部進行身分驗證的使用者 \(聯合身分\)](#)。
- 如需了解使用角色和資源型政策進行跨帳戶存取之間的差異，請參閱《[IAM 使用者指南](#)》中的 [IAM 中的跨帳戶資源存取](#)。

在 Amazon Polly 中記錄和監控

監控是維護 Amazon Polly 應用程式可靠性、可用性和效能的重要部分。若要監控 Amazon Polly API 呼叫，您可以使用 AWS CloudTrail。若要監控任務的狀態，請使用 Amazon CloudWatch Logs。

- Amazon CloudWatch 警示 – 使用 CloudWatch 警示，您可以在指定的期間內監看單一指標。如果指標超過指定的閾值，則會將通知傳送至 Amazon Simple Notification Service 主題或 AWS Auto Scaling 政策。當指標處於特定狀態時，CloudWatch 警示不會叫用動作。必須是狀態已變更並維持了所指定的時間長度，才會呼叫動作。如需詳細資訊，請參閱[整合 CloudWatch 和 Amazon Polly](#)。
- CloudTrail 日誌 – CloudTrail 提供由 Amazon Polly AWS 中的使用者、角色或服務所採取之動作的記錄。您可以使用 CloudTrail 收集的資訊，判斷向 Amazon Polly 提出的請求。您還可以判斷提出請求的來源 IP 地址、提出請求的人員和時間以及其他詳細資訊。如需詳細資訊，請參閱[使用記錄 Amazon Polly API 呼叫 AWS CloudTrail](#)。

Amazon Polly 的合規驗證

在多個合規計畫中，第三方稽核人員會評估 Amazon Polly 的安全與 AWS 合規。這些計畫包括 SOC、PCI、FedRAMP、HIPAA 等等。

如需特定合規計畫範圍內 AWS 的服務清單，請參閱[AWS 合規計畫範圍內的服務](#)。如需一般資訊，請參閱 [AWS Compliance Programs](#)。

您可以使用 下載第三方稽核報告 AWS Artifact。如需詳細資訊，請參閱中的[在 AWS Artifact 中下載報告](#)。

您使用 Amazon Polly 時的合規責任取決於資料的機密性、您公司的合規目標，以及適用的法律和法規。AWS 提供下列資源以協助合規：

- [安全與合規快速入門指南](#)：這些部署指南討論架構考量，並提供在 AWS 上部署以安全及合規為重心之基準環境的步驟。
- [HIPAA 安全與合規架構白皮書](#) – 此白皮書說明公司如何使用 AWS 來建立符合 HIPAA 規範的應用程式。
- [AWS 合規資源](#) – 此工作手冊和指南集合可能適用於您的產業和位置。
- 《AWS Config 開發人員指南》中的[使用規則評估資源](#) – AWS Config 服務會評估資源組態符合內部實務、產業準則和法規的程度。
- [AWS Security Hub CSPM](#) – AWS 此服務提供 內安全狀態的全方位檢視 AWS，可協助您檢查是否符合安全產業標準和最佳實務。

Amazon Polly 中的彈性

AWS 全球基礎設施是以 AWS 區域和可用區域為基礎建置的。AWS 區域提供多個實體隔離和隔離的可用區域，這些區域以低延遲、高輸送量和高度備援聯網連接。透過可用區域，您可以設計與操作的應用程式和資料庫，在可用區域之間自動容錯移轉而不會發生中斷。可用區域的可用性、容錯能力和擴展能力，均較單一或多個資料中心的傳統基礎設施還高。

如需 AWS 區域和可用區域的詳細資訊，請參閱 [AWS 全球基礎設施](#)。

Amazon Polly 中的基礎設施安全

作為受管服務，Amazon Polly 受到 [Amazon Web Services：安全程序概觀](#) 白皮書中所述的 AWS 全球網路安全程序的保護。

您可以使用 AWS 發佈的 API 呼叫，透過網路存取 Amazon Polly。用戶端必須支援 Transport Layer Security (TLS) 1.0 或更新版本。建議使用 TLS 1.2 或更新版本。用戶端也必須支援具備完美轉送私密 (PFS) 的密碼套件，例如臨時 Diffie-Hellman (DHE) 或橢圓曲線臨時 Diffie-Hellman (ECDHE)。現代系統 (如 Java 7 和更新版本) 大多會支援這些模式。

此外，請求必須使用存取金鑰 ID 和與 IAM 主體相關聯的私密存取金鑰來簽署。或者，您可以透過 [AWS Security Token Service](#) (AWS STS) 來產生暫時安全憑證來簽署請求。

Amazon Polly 的安全最佳實務

我們將您的信任、您內容的隱私和安全性放在第一優先。而且實作可靠且複雜的技術和實體控制，旨在避免未授權人員存取或公開您的內容，同時確保對內容的使用絕對遵守我們對您所做的承諾。如需更多資訊，請參閱 [AWS 資料隱私權常見問答集](#)。

Amazon Polly 不會保留文字提交的內容。

如需 AWS 安全性的廣泛檢視，包括合規、滲透測試、公告和資源，請造訪 [AWS Cloud Security](#) 網站。

搭配界面 VPC 端點使用 Amazon Polly

如果您使用 Amazon Virtual Private Cloud (Amazon VPC) 託管 AWS 資源，您可以在 VPC 和 Amazon Polly 之間建立私有連線。您可以使用此連線來與 Amazon Polly 合成語音，而無需周遊公有網際網路。

Amazon VPC 是一種 AWS 服務，可用來在您定義的虛擬網路中啟動 AWS 資源。您可利用 VPC 來控制您的網路設定，例如 IP 地址範圍、子網路、路由表和網路閘道。若要將 VPC 連線至 Amazon Polly，您可以定義 Amazon Polly 的介面 VPC 端點。這種類型的端點可讓您將 VPC 連線到 AWS 服務。端點為 Amazon Polly 提供可靠、可擴展的連線，而不需要網際網路閘道、網路位址轉譯 (NAT) 執行個體或 VPN 連線。如需詳細資訊，請參閱 [《Amazon VPC 使用者指南》中的什麼是 Amazon VPC](#)。

介面 VPC 端點採用一種 AWS 技術 AWS PrivateLink，可在使用具有私有 IP 地址的彈性網路介面之間 AWS 服務 進行私有通訊。如需詳細資訊，請參閱 [新增 - AWS PrivateLink for AWS 服務](#)。

下列步驟適用於 Amazon VPC 的使用者。如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的 [入門](#)。

可用性

支援 [Amazon Polly 的所有區域都支援](#) VPC 端點。如需 AWS 區域和可用區域的詳細資訊，請參閱 [AWS 全球基礎設施](#)。

FIPS 端點可在下列區域使用：

- 美國東部 (維吉尼亞北部) (us-east-1)
- 美國東部 (俄亥俄) (us-east-2)
- 美國西部 (加利佛尼亞北部) (us-west-1)
- 美國西部 (奧勒岡) (us-west-2)
- AWS GovCloud (美國西部) (us-gov-west-1)
- 加拿大 (中部) (ca-central-1)

FIPS 端點的格式為 `com.amazonaws.REGION.polly-fips`。

為 Amazon Polly 建立 VPC 端點

若要開始將 Amazon Polly 與 VPC 搭配使用，請為 Amazon Polly 建立介面 VPC 端點。要選擇的服務是 `com.amazonaws.Region.polly`。您不需要變更 Amazon Polly 的任何設定。如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的 [建立介面端點](#)。

測試 VPC 與 Amazon Polly 之間的連線

建立端點後，您可以測試連線。

測試 VPC 與 Amazon Polly 端點之間的連線

1. 連線至位於 VPC 中的 Amazon EC2 執行個體。如需連線的相關資訊，請參閱 Amazon EC2 文件中的[連線至 Linux 執行個體](#)或[連線至 Windows 執行個體](#)。
2. 從執行個體，aws polly describe-voices從 AWS CLI 使用 列出可用的 Amazon Polly 語音。

如果對命令的回應包含可用 Amazon Polly 語音的清單，則命令已成功，而且您的 VPC 端點正在運作。

控制對 Amazon Polly 端點的存取

當您建立或修改端點時，VPC 端點政策是您連接至端點的 IAM 資源政策。如果您未在建立端點時連接政策，我們會以預設政策連接以允許完整存取服務。端點政策不會覆寫或取代 IAM 使用者政策或服務特定的政策。這個另行區分的政策會控制從端點到所指定之服務的存取。

端點政策必須以 JSON 格式撰寫。

如需詳細資訊，請參閱《Amazon VPC 使用者指南》中的[使用 VPC 端點控制服務的存取](#)。

以下是 Amazon Polly 端點政策的範例。此政策可讓使用者透過 VPC 連線至 Amazon Polly，以描述語音並與 Amazon Polly 合成語音，並防止他們執行其他 Amazon Polly 動作。

```
{
  "Statement": [
    {
      "Sid": "SynthesisAndDescribeVoicesOnly",
      "Principal": "*",
      "Action": [
        "polly:DescribeVoices",
        "polly:SynthesizeSpeech"
      ],
      "Effect": "Allow",
      "Resource": "*"
    }
  ]
}
```

修改 Amazon Polly 的 VPC 端點政策

1. 開啟位於 <https://console.aws.amazon.com/vpc> 的 Amazon VPC 主控台。

2. 在導覽窗格中選擇端點。
3. 如果您尚未建立 Amazon Polly 的端點，請選擇建立端點。然後選取 `com.amazonaws.Region.polly`，然後選擇建立端點。
4. 選取 `com.amazonaws.Region.polly` 端點，然後選擇畫面下半部的政策索引標籤。
5. 選擇 Edit Policy (編輯政策)，並對政策做出變更。

VPC 內容金鑰支援

Amazon Polly 支援 `aws:SourceVpc` 和 `aws:SourceVpce` 內容金鑰，可以限制對特定 VPCs 存取。這些金鑰只有在使用者使用 VPC 端點時才會運作。如需詳細資訊，請參閱《IAM 使用者指南》中的 [適用於某些服務的金鑰](#)。

使用 記錄 Amazon Polly API 呼叫 AWS CloudTrail

Amazon Polly 已與 服務整合 AWS CloudTrail，此服務提供 Amazon Polly AWS 中使用者、角色或服務所採取動作的記錄。CloudTrail 會將 Amazon Polly 的所有 API 呼叫擷取為事件。擷取的呼叫包括來自 Amazon Polly 主控台的呼叫，以及對 Amazon Polly API 操作的程式碼呼叫。如果您建立線索，您可以將 CloudTrail 事件持續交付至 Amazon S3 儲存貯體，包括 Amazon Polly 的事件。即使您未設定追蹤，依然可以透過 CloudTrail 主控台的事件歷史記錄檢視最新事件。您可以使用 CloudTrail 所收集的資訊，判斷向 Amazon Polly 提出的請求、提出請求的 IP 地址、提出請求的人員、提出請求的時間，以及其他詳細資訊。

若要進一步了解 CloudTrail，包括如何設定及啟用，請參閱 [《AWS CloudTrail 使用者指南》](#)。

CloudTrail 中的 Amazon Polly 資訊

當您建立 AWS 帳戶時，會在您的帳戶上啟用 CloudTrail。當 Amazon Polly 中發生支援的事件活動時，該活動會與事件歷史記錄中的其他 AWS 服務事件一起記錄在 CloudTrail 事件中。您可以在 AWS 帳戶中檢視、搜尋和下載最近的事件。如需詳細資訊，請參閱《使用 CloudTrail 事件歷史記錄檢視事件》<https://docs.aws.amazon.com/awscloudtrail/latest/userguide/view-cloudtrail-events.html>。

若要不持續記錄您 AWS 帳戶中的事件，包括 Amazon Polly 的事件，請建立追蹤。線索能讓 CloudTrail 將日誌檔案交付至 Amazon S3 儲存貯體。根據預設，當您在主控台中建立線索時，線索會套用至所有 AWS 區域。線索會記錄 AWS 分割區中所有區域的事件，並將日誌檔案交付至您指定的 Amazon S3 儲存貯體。此外，您可以設定其他 AWS 服務，以進一步分析和處理 CloudTrail 日誌中收集的事件資料。如需詳細資訊，請參閱下列內容：

- [建立追蹤的概觀](#)
- [CloudTrail 支援的服務和整合](#)
- [設定 CloudTrail 的 Amazon SNS 通知](#)
- [從多個區域接收 CloudTrail 日誌檔案](#)，以及[從多個帳戶接收 CloudTrail 日誌檔案](#)

Amazon Polly 支援將下列動作記錄為 CloudTrail 日誌檔案中的事件：

- [DeleteLexicon](#)
- [DescribeVoices](#)
- [GetLexicon](#)
- [GetSpeechSynthesisTask](#)

- [ListLexicons](#)
- [ListSpeechSynthesisTasks](#)
- [PutLexicon](#)
- [StartSpeechSynthesisTask](#)
- [SynthesizeSpeech](#)

每一筆事件或日誌專案都會包含產生請求者的資訊。身分資訊可協助您判斷下列事項：

- 請求是使用根使用者還是 AWS Identity and Access Management (IAM) 使用者登入資料提出。
- 提出該請求時，是否使用了特定角色或聯合身分使用者的暫時安全憑證。
- 請求是否由其他 AWS 服務提出。

如需詳細資訊，請參閱 [CloudTrail userIdentity 元素](#)。

範例：Amazon Polly 日誌檔案項目

追蹤是一種組態，能讓事件以日誌檔案的形式交付到您指定的 Amazon S3 儲存貯體。CloudTrail 日誌檔案包含一或多個日誌專案。一個事件為任何來源提出的單一請求，並包含請求動作、請求的日期和時間、請求參數等資訊。CloudTrail 日誌檔並非依公有 API 呼叫的堆疊追蹤排序，因此不會以任何特定順序出現。

下列範例顯示示範的 CloudTrail 日誌項目 SynthesizeSpeech。

```
{
  "Records": [
    {
      "awsRegion": "us-east-2",
      "eventID": "19bd70f7-5e60-4cdc-9825-936c552278ae",
      "eventName": "SynthesizeSpeech",
      "eventSource": "polly.amazonaws.com",
      "eventTime": "2016-11-02T03:49:39Z",
      "eventType": "AwsApiCall",
      "eventVersion": "1.05",
      "recipientAccountId": "123456789012",
      "requestID": "414288c2-a1af-11e6-b17f-d7cfc06cb461",
      "requestParameters": {
        "lexiconNames": [
          "SampleLexicon"
        ]
      }
    }
  ]
}
```

```
    ],
    "engine": "neural",
    "outputFormat": "mp3",
    "sampleRate": "22050",
    "text": "*****",
    "textType": "text",
    "voiceId": "Kendra"
  },
  "responseElements": null,
  "sourceIPAddress": "1.2.3.4",
  "userAgent": "Amazon CLI/Polly 1.10 API 2016-06-10",
  "userIdentity": {
    "accessKeyId": "EXAMPLE_KEY_ID",
    "accountId": "123456789012",
    "arn": "arn:aws:iam::123456789012:user/Alice",
    "principalId": "EX_PRINCIPAL_ID",
    "type": "IAMUser",
    "userName": "Alice"
  }
}
```

整合 CloudWatch 和 Amazon Polly

當您與 Amazon Polly 互動時，它會每分鐘將下列指標和維度傳送至 CloudWatch。您可以使用下列程序來檢視 Amazon Polly 的指標。

您可以使用 CloudWatch 監控 Amazon Polly，該 CloudWatch 會收集來自 Amazon Polly 的原始資料，並將其處理為可讀且近乎即時的指標。這些統計資料會保存兩週的期間，以便您存取 historical information，並更清楚 Web 應用程式或服務的執行方式。根據預設，Amazon Polly 指標資料會以 1 分鐘的間隔傳送至 CloudWatch。如需詳細資訊，請參閱《[Amazon CloudWatch 使用者指南](#)》中的什麼是 Amazon CloudWatch。

取得 CloudWatch 指標（主控台）

1. 透過 <https://console.aws.amazon.com/cloudwatch/> 開啟 CloudWatch 主控台。
2. 在導覽窗格中，選擇指標。
3. 在 CloudWatch Metrics by Category 窗格中，在 Amazon Polly 的指標類別下，選取指標類別，然後在上方窗格中向下捲動以檢視指標的完整清單。

在上取得 CloudWatch 指標 AWS CLI

下列程式碼顯示 Amazon Polly 的可用指標。

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

上述命令會傳回類似下列的 Amazon Polly 指標清單。MetricName 元素可識別指標。

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

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

如需詳細資訊，請參閱《Amazon CloudWatch API 參考》中的 [GetMetricStatistics](#)。

Amazon Polly 指標

Amazon Polly 會為每個請求產生下列指標。這些指標會彙總，並以一分鐘的間隔傳送至 CloudWatch，以供其使用。

指標	描述
RequestCharacters	請求中的字元數。這只是計費字元，不包含 SSML 標籤。 有效維度：操作 有效統計資料：最小值、最大值、平均值、SampleCount、總和 單位：計數
ResponseLatency	發出請求與開始串流回應之間的延遲。 有效維度：操作 有效統計資料：最小值、最大值、平均值、SampleCount 單位：微秒

指標	描述
2XXCount	成功回應後傳回 HTTP 200 層級程式碼。 有效維度：操作 有效統計資料：平均、SampleCount、總和 單位：計數
4XXCount	發生錯誤時傳回 HTTP 400 層級錯誤代碼。對於每個成功的回應，都會發出零 (0)。 有效維度：操作 有效統計資料：平均、SampleCount、總和 單位：計數
5XXCount	發生錯誤時傳回 HTTP 500 層級錯誤代碼。對於每個成功的回應，都會發出零 (0)。 有效維度：操作 有效統計資料：平均、SampleCount、總和 單位：計數

Amazon Polly 指標的維度

Amazon Polly 指標使用 AWS/Polly 命名空間，並提供下列維度的指標：

維度	描述
Operation	指標會依其參考的 API 方法分組。可能的值為 SynthesizeSpeech、PutLexicon、DescribeVoices 等。

Amazon Polly API 參考

Amazon Polly 是一種 Web 服務，可讓您輕鬆地從文字合成語音。

Amazon Polly 服務提供從純文字和語音合成標記語言 (SSML) 合成高品質語音的 API 操作，以及管理發音語彙，讓您能夠取得應用程式網域的最佳結果。

已驗證的 API 呼叫必須使用簽章第 4 版簽署程序簽署。如需詳細資訊，請參閱《》中的[簽署 AWS API 請求](#) Amazon Web Services 一般參考。

主題

- [動作](#)
- [資料類型](#)
- [常見錯誤類型](#)
- [常見參數](#)

動作

支援以下動作：

- [DeleteLexicon](#)
- [DescribeVoices](#)
- [GetLexicon](#)
- [GetSpeechSynthesisTask](#)
- [ListLexicons](#)
- [ListSpeechSynthesisTasks](#)
- [PutLexicon](#)
- [StartSpeechSynthesisStream](#)
- [StartSpeechSynthesisTask](#)
- [SynthesizeSpeech](#)

DeleteLexicon

刪除存放在 中的指定發音語彙 AWS 區域。已刪除的語彙無法用於語音合成，也無法使用 GetLexicon 或 ListLexicon APIs 擷取語彙。

如需詳細資訊，請參閱[管理語彙](#)。

請求語法

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

URI 請求參數

請求會使用下列 URI 參數。

LexiconName

要刪除的語彙名稱。必須是 區域中現有的語彙。

模式：[0-9A-Za-z]{1,20}

必要：是

請求主體

請求沒有請求主體。

回應語法

```
HTTP/1.1 200
```

回應元素

如果動作成功，則服務會傳回具空 HTTP 內文的 HTTP 200 回應。

錯誤

LexiconNotFoundException

Amazon Polly 找不到指定的語彙。這可能是由於缺少語彙、其名稱拼寫錯誤或指定位於不同區域的語彙所致。

確認語彙存在、位於 區域 (請參閱[ListLexicons](#)) , 且您拼寫的名稱拼寫正確。然後再試一次。

HTTP 狀態碼 : 404

ServiceFailureException

不明條件導致服務失敗。

HTTP 狀態碼 : 500

另請參閱

如需在其中一種語言特定 AWS SDKs中使用此 API 的詳細資訊，請參閱下列內容：

- [AWS 命令列界面 V2](#)
- [AWS 適用於 .NET V4 的 SDK](#)
- [AWS 適用於 C++ 的 SDK](#)
- [AWS 適用於 Go 的 SDK v2](#)
- [AWS 適用於 Java V2 的 SDK](#)
- [AWS 適用於 JavaScript V3 的 SDK](#)
- [AWS 適用於 Kotlin 的 SDK](#)
- [AWS 適用於 PHP V3 的 SDK](#)
- [AWS 適用於 Python 的 SDK](#)
- [AWS 適用於 Ruby V3 的 SDK](#)

DescribeVoices

傳回請求語音合成時可用的語音清單。每個語音都會說指定的語言，可以是男性或女性，並以 ID 識別，這是語音名稱的 ASCII 版本。

合成語音 (SynthesizeSpeech) 時，您可以從傳回的語音清單中，提供所需語音的語音 ID `DescribeVoices`。

例如，您希望新聞閱讀器應用程式讀取特定語言的新聞，但提供使用者選擇語音的選項。使用 `DescribeVoices` 操作，您可以為使用者提供可供選擇的可用語音清單。

您可以選擇指定語言代碼來篩選可用的聲音。例如，如果您指定 `en-US`，操作會傳回所有可用美式英文語音的清單。

這項操作需要許可來執行 `polly:DescribeVoices` 動作。

請求語法

```
GET /v1/voices?  
Engine=Engine&IncludeAdditionalLanguageCodes=IncludeAdditionalLanguageCodes&LanguageCode=LanguageCode  
HTTP/1.1
```

URI 請求參數

請求會使用下列 URI 參數。

[Engine](#)

指定 Amazon Polly 在處理語音合成的輸入文字時所使用的引擎 (`standardneural`、`long-form` 或 `generative`)。

有效值: `standard` | `neural` | `long-form` | `generative`

[IncludeAdditionalLanguageCodes](#)

布林值，指出是否傳回任何使用指定語言作為額外語言的雙語語音。例如，如果您請求所有使用美式英文 (`es-US`) 的語言，並且有一個義大利語語音會說義大利文 (`it-IT`) 和美式英文，則如果您指定 `yes` 但指定 `no` 則不會包含該語音。

[LanguageCode](#)

語言識別標籤（語言名稱 ISO 3166 國家/地區代碼的 ISO 639 代碼），用於篩選傳回的語音清單。如果您未指定此選用參數，則會傳回所有可用的語音。

有效值:arb | cmn-CN | cy-GB | da-DK | de-DE | en-AU | en-GB | en-GB-WLS | en-IN | en-US | es-ES | es-MX | es-US | fr-CA | fr-FR | is-IS | it-IT | ja-JP | hi-IN | ko-KR | nb-NO | nl-NL | pl-PL | pt-BR | pt-PT | ro-RO | ru-RU | sv-SE | tr-TR | en-NZ | en-ZA | ca-ES | de-AT | yue-CN | ar-AE | fi-FI | en-IE | nl-BE | fr-BE | cs-CZ | de-CH | en-SG

[NextToken](#)

從上一個DescribeVoices操作傳回的不透明分頁字符。如果存在，則表示要在何處繼續列出。

長度限制：長度下限為 0。長度上限為 4096。

請求主體

請求沒有請求主體。

回應語法

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

{
  "NextToken": "string",
  "Voices": [
    {
      "AdditionalLanguageCodes": [ "string" ],
      "Gender": "string",
      "Id": "string",
      "LanguageCode": "string",
      "LanguageName": "string",
      "Name": "string",
      "SupportedEngines": [ "string" ]
    }
  ]
}
```

回應元素

如果動作成功，則服務傳回 HTTP 200 回應。

服務會傳回下列 JSON 格式的資料。

[NextToken](#)

在下一個請求中用於繼續語音清單的分頁字符。只有在回應被截斷時NextToken才會傳回。

類型：字串

長度限制：長度下限為 0。長度上限為 4096。

[Voices](#)

語音及其屬性的清單。

類型：[Voice](#) 物件陣列

錯誤

InvalidNextTokenException

NextToken 無效。請確認拼字正確，然後再試一次。

HTTP 狀態碼：400

ServiceFailureException

不明條件導致服務失敗。

HTTP 狀態碼：500

另請參閱

如需在其中一種語言特定 AWS SDKs中使用此 API 的詳細資訊，請參閱下列內容：

- [AWS 命令列界面 V2](#)
- [AWS 適用於 .NET V4 的 SDK](#)
- [AWS 適用於 C++ 的 SDK](#)
- [AWS 適用於 Go 的 SDK v2](#)
- [AWS 適用於 Java V2 的 SDK](#)
- [AWS 適用於 JavaScript V3 的 SDK](#)
- [AWS 適用於 Kotlin 的 SDK](#)
- [AWS 適用於 PHP V3 的 SDK](#)

- [AWS 適用於 Python 的 SDK](#)
- [AWS 適用於 Ruby V3 的 SDK](#)

GetLexicon

傳回存放在 中指定發音語彙的內容 AWS 區域。如需詳細資訊，請參閱[管理語彙](#)。

請求語法

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

URI 請求參數

請求會使用下列 URI 參數。

LexiconName

語彙的名稱。

模式：`[0-9A-Za-z]{1,20}`

必要：是

請求主體

請求沒有請求主體。

回應語法

```
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
  }
}
```

```
}  
}
```

回應元素

如果動作成功，則服務傳回 HTTP 200 回應。

服務會傳回下列 JSON 格式的資料。

[Lexicon](#)

提供語彙名稱和字串內容的語彙物件。

類型：[Lexicon](#) 物件

[LexiconAttributes](#)

語彙的中繼資料，包括使用的音標字母、語言代碼、語彙 ARN、語彙中定義的語彙數量，以及以位元組為單位的語彙大小。

類型：[LexiconAttributes](#) 物件

錯誤

LexiconNotFoundException

Amazon Polly 找不到指定的語彙。這可能是由於缺少語彙、其名稱拼寫錯誤或指定位於不同區域的語彙所致。

確認語彙存在、位於 區域（請參閱[ListLexicons](#)），且您拼寫的名稱拼寫正確。然後再試一次。

HTTP 狀態碼：404

ServiceFailureException

不明條件導致服務失敗。

HTTP 狀態碼：500

另請參閱

如需在其中一種語言特定 AWS SDKs 中使用此 API 的詳細資訊，請參閱下列內容：

- [AWS 命令列界面 V2](#)
- [AWS 適用於 .NET V4 的 SDK](#)
- [AWS 適用於 C++ 的 SDK](#)
- [AWS 適用於 Go 的 SDK v2](#)
- [AWS 適用於 Java V2 的 SDK](#)
- [AWS 適用於 JavaScript V3 的 SDK](#)
- [AWS 適用於 Kotlin 的 SDK](#)
- [AWS 適用於 PHP V3 的 SDK](#)
- [AWS 適用於 Python 的 SDK](#)
- [AWS 適用於 Ruby V3 的 SDK](#)

GetSpeechSynthesisTask

根據其 TaskID 擷取特定的 SpeechSynthesisTask 物件。此物件包含指定語音合成任務的相關資訊，包括任務的狀態，以及包含任務輸出的 S3 儲存貯體連結。

請求語法

```
GET /v1/synthesisTasks/TaskId HTTP/1.1
```

URI 請求參數

請求會使用下列 URI 參數。

TaskId

Amazon Polly 為語音合成任務產生的識別符。

模式：`^[a-zA-Z0-9_-]{1,100}$`

必要：是

請求主體

請求沒有請求主體。

回應語法

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

{
  "SynthesisTask": {
    "CreationTime": number,
    "Engine": "string",
    "LanguageCode": "string",
    "LexiconNames": [ "string" ],
    "OutputFormat": "string",
    "OutputUri": "string",
    "RequestCharacters": number,
    "SampleRate": "string",
    "SnsTopicArn": "string",
    "SpeechMarkTypes": [ "string" ],
```

```
    "TaskId": "string",  
    "TaskStatus": "string",  
    "TaskStatusReason": "string",  
    "TextType": "string",  
    "VoiceId": "string"  
  }  
}
```

回應元素

如果動作成功，則服務傳回 HTTP 200 回應。

服務會傳回下列 JSON 格式的資料。

[SynthesisTask](#)

SynthesisTask 物件，可提供來自請求任務的資訊，包括輸出格式、建立時間、任務狀態等。

類型：[SynthesisTask](#) 物件

錯誤

InvalidTaskIdException

提供的任務 ID 無效。請提供有效的任務 ID，然後再試一次。

HTTP 狀態碼：400

ServiceFailureException

不明條件導致服務失敗。

HTTP 狀態碼：500

SynthesisTaskNotFoundException

找不到具有請求任務 ID 的語音合成任務。

HTTP 狀態碼：400

另請參閱

如需在其中一種語言特定 AWS SDKs 中使用此 API 的詳細資訊，請參閱下列內容：

- [AWS 命令列界面 V2](#)
- [AWS 適用於 .NET V4 的 SDK](#)
- [AWS 適用於 C++ 的 SDK](#)
- [AWS 適用於 Go 的 SDK v2](#)
- [AWS 適用於 Java V2 的 SDK](#)
- [AWS 適用於 JavaScript V3 的 SDK](#)
- [AWS 適用於 Kotlin 的 SDK](#)
- [AWS 適用於 PHP V3 的 SDK](#)
- [AWS 適用於 Python 的 SDK](#)
- [AWS 適用於 Ruby V3 的 SDK](#)

ListLexicons

傳回存放在 中的發音語彙清單 AWS 區域。如需詳細資訊，請參閱[管理語彙](#)。

請求語法

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

URI 請求參數

請求會使用下列 URI 參數。

[NextToken](#)

從先前ListLexicons操作傳回的不透明分頁字符。如果有的話，請指出在何處繼續語彙清單。

長度限制：長度下限為 0。長度上限為 4096。

請求主體

請求沒有請求主體。

回應語法

```
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"  
}
```

回應元素

如果動作成功，則服務傳回 HTTP 200 回應。

服務會傳回下列 JSON 格式的資料。

[Lexicons](#)

語彙名稱和屬性的清單。

類型：[LexiconDescription](#) 物件陣列

[NextToken](#)

在下一個請求中用於繼續語彙清單的分頁字符。只有在回應被截斷時NextToken才會傳回。

類型：字串

長度限制：長度下限為 0。長度上限為 4096。

錯誤

InvalidNextTokenException

NextToken 無效。請確認拼字正確，然後再試一次。

HTTP 狀態碼：400

ServiceFailureException

不明條件導致服務失敗。

HTTP 狀態碼：500

另請參閱

如需在其中一種語言特定 AWS SDKs中使用此 API 的詳細資訊，請參閱下列內容：

- [AWS 命令列界面 V2](#)

- [AWS 適用於 .NET V4 的 SDK](#)
- [AWS 適用於 C++ 的 SDK](#)
- [AWS 適用於 Go 的 SDK v2](#)
- [AWS 適用於 Java V2 的 SDK](#)
- [AWS 適用於 JavaScript V3 的 SDK](#)
- [AWS 適用於 Kotlin 的 SDK](#)
- [AWS 適用於 PHP V3 的 SDK](#)
- [AWS 適用於 Python 的 SDK](#)
- [AWS 適用於 Ruby V3 的 SDK](#)

ListSpeechSynthesisTasks

傳回依建立日期排序的 SpeechSynthesisTask 物件清單。此操作可以依任務狀態篩選任務，例如，允許使用者僅列出已完成的任務。

請求語法

```
GET /v1/synthesisTasks?MaxResults=MaxResults&NextToken=NextToken&Status=Status HTTP/1.1
```

URI 請求參數

請求會使用下列 URI 參數。

[MaxResults](#)

清單操作中傳回的語音合成任務數目上限。

有效範圍：最小值為 1。最大值為 100。

[NextToken](#)

在下一個請求中使用的分頁字符，以繼續語音合成任務的清單。

長度限制：長度下限為 0。長度上限為 4096。

[Status](#)

清單操作中傳回的語音合成任務狀態

有效值:scheduled | inProgress | completed | failed

請求主體

請求沒有請求主體。

回應語法

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

```
{
  "NextToken": "string",
  "SynthesisTasks": [
    {
      "CreationTime": number,
      "Engine": "string",
      "LanguageCode": "string",
      "LexiconNames": [ "string" ],
      "OutputFormat": "string",
      "OutputUri": "string",
      "RequestCharacters": number,
      "SampleRate": "string",
      "SnsTopicArn": "string",
      "SpeechMarkTypes": [ "string" ],
      "TaskId": "string",
      "TaskStatus": "string",
      "TaskStatusReason": "string",
      "TextType": "string",
      "VoiceId": "string"
    }
  ]
}
```

回應元素

如果動作成功，則服務傳回 HTTP 200 回應。

服務會傳回下列 JSON 格式的資料。

[NextToken](#)

此請求中從上一個清單操作傳回的不透明分頁字符。如果存在，則表示要在何處繼續列出。

類型：字串

長度限制：長度下限為 0。長度上限為 4096。

[SynthesisTasks](#)

SynthesisTask 物件的清單，提供來自清單請求中指定任務的資訊，包括輸出格式、建立時間、任務狀態等。

類型：[SynthesisTask](#) 物件陣列

錯誤

InvalidNextTokenException

NextToken 無效。請確認拼字正確，然後再試一次。

HTTP 狀態碼：400

ServiceFailureException

不明條件導致服務失敗。

HTTP 狀態碼：500

另請參閱

如需在其中一種語言特定 AWS SDKs 中使用此 API 的詳細資訊，請參閱下列內容：

- [AWS 命令列界面 V2](#)
- [AWS 適用於 .NET V4 的 SDK](#)
- [AWS 適用於 C++ 的 SDK](#)
- [AWS 適用於 Go 的 SDK v2](#)
- [AWS 適用於 Java V2 的 SDK](#)
- [AWS 適用於 JavaScript V3 的 SDK](#)
- [AWS 適用於 Kotlin 的 SDK](#)
- [AWS 適用於 PHP V3 的 SDK](#)
- [AWS 適用於 Python 的 SDK](#)
- [AWS 適用於 Ruby V3 的 SDK](#)

PutLexicon

將發音語彙存放在中 AWS 區域。如果區域中已存在同名的語彙，則新語彙會覆寫該語彙。語彙操作具有最終一致性，因此可能需要一些時間才能讓語彙可供 SynthesizeSpeech 操作使用。

如需詳細資訊，請參閱[管理語彙](#)。

請求語法

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

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

URI 請求參數

請求會使用下列 URI 參數。

[LexiconName](#)

語彙的名稱。名稱必須遵循一般快速格式 **【0-9A-Za-z】{1, 20}**。也就是說，名稱是區分大小寫的英數字串，長度上限為 20 個字元。

模式：**[0-9A-Za-z]{1,20}**

必要：是

請求主體

請求接受採用 JSON 格式的下列資料。

[Content](#)

PLS 語彙的內容做為字串資料。

類型：字串

必要：是

回應語法

```
HTTP/1.1 200
```

回應元素

如果動作成功，則服務會傳回具空 HTTP 內文的 HTTP 200 回應。

錯誤

InvalidLexiconException

Amazon Polly 找不到指定的語彙。請確認語彙的名稱拼字正確，然後再試一次。

HTTP 狀態碼：400

LexiconSizeExceededException

此操作會超過指定語彙的大小上限。

HTTP 狀態碼：400

MaxLexemeLengthExceededException

此操作會超過 lexeme 的大小上限。

HTTP 狀態碼：400

MaxLexiconsNumberExceededException

此操作將超過語彙的數量上限。

HTTP 狀態碼：400

ServiceFailureException

不明條件導致服務失敗。

HTTP 狀態碼：500

UnsupportedPlsAlphabetException

語彙指定的字母不是支援的字母。有效值為 x-sampa 和 ipa。

HTTP 狀態碼：400

UnsupportedPlsLanguageException

不支援語彙中指定的語言。如需支援的語言清單，請參閱[語彙屬性](#)。

HTTP 狀態碼：400

另請參閱

如需在其中一種語言特定 AWS SDKs 中使用此 API 的詳細資訊，請參閱下列內容：

- [AWS 命令列界面 V2](#)
- [AWS 適用於 .NET V4 的 SDK](#)
- [AWS 適用於 C++ 的 SDK](#)
- [AWS 適用於 Go 的 SDK v2](#)
- [AWS 適用於 Java V2 的 SDK](#)
- [AWS 適用於 JavaScript V3 的 SDK](#)
- [AWS 適用於 Kotlin 的 SDK](#)
- [AWS 適用於 PHP V3 的 SDK](#)
- [AWS 適用於 Python 的 SDK](#)
- [AWS 適用於 Ruby V3 的 SDK](#)

StartSpeechSynthesisStream

透過雙向串流連線合成 UTF-8 輸入、純文字或 SSML。在 HTTP/2 標頭中指定合成參數、在輸入串流上以事件形式遞增傳送文字，並在可用時接收合成音訊。

此操作做為 的雙向對等項目SynthesizeSpeech：

- [SynthesizeSpeech](#)

請求語法

```
POST /v1/synthesisStream HTTP/1.1
x-amzn-Engine: Engine
x-amzn-LanguageCode: LanguageCode
x-amzn-LexiconNames: LexiconNames
x-amzn-OutputFormat: OutputFormat
x-amzn-SampleRate: SampleRate
x-amzn-VoiceId: VoiceId
Content-type: application/json

{
  "CloseStreamEvent": {
  },
  "TextEvent": {
    "FlushStreamConfiguration": {
      "Force": boolean
    },
    "Text": "string",
    "TextType": "string"
  }
}
```

URI 請求參數

請求會使用下列 URI 參數。

[Engine](#)

指定 Amazon Polly 在處理語音合成的輸入文字時要使用的引擎。目前僅支援 generative引擎。如果您指定所選引擎不支援的語音，Amazon Polly 會傳回錯誤。

有效值:standard | neural | long-form | generative

必要：是

LanguageCode

選用參數，可設定語音合成請求的語言代碼。只有在使用雙語語音時，才指定此參數。如果使用雙語語音且未指定語言代碼，Amazon Polly 會使用雙語語音的預設語言。

有效值:arb | cmn-CN | cy-GB | da-DK | de-DE | en-AU | en-GB | en-GB-WLS | en-IN | en-US | es-ES | es-MX | es-US | fr-CA | fr-FR | is-IS | it-IT | ja-JP | hi-IN | ko-KR | nb-NO | nl-NL | pl-PL | pt-BR | pt-PT | ro-RO | ru-RU | sv-SE | tr-TR | en-NZ | en-ZA | ca-ES | de-AT | yue-CN | ar-AE | fi-FI | en-IE | nl-BE | fr-BE | cs-CZ | de-CH | en-SG

LexiconNames

要在合成期間套用之服務的一或多個發音語彙的名稱。Amazon Polly 才會套用語彙。

陣列成員：最多 5 個項目。

模式：[0-9A-Za-z]{1,20}

OutputFormat

合成語音的音訊格式。目前，Amazon Polly 不支援 JSON 語音標記。

有效值:json | mp3 | ogg_opus | ogg_vorbis | pcm | mulaw | alaw

必要：是

SampleRate

音訊頻率，以 Hz 指定。

Voiceld

用於合成的語音。若要取得可用語音 IDs 的清單，請使用 [DescribeVoices](#) 操作。

有效值:Aditi | Amy | Astrid | Bianca | Brian | Camila | Carla | Carmen | Celine | Chantal | Conchita | Cristiano | Dora | Emma | Enrique | Ewa | Filiz | Gabrielle | Geraint | Giorgio | Gwyneth | Hans | Ines | Ivy | Jacek | Jan | Joanna | Joey | Justin | Karl | Kendra | Kevin | Kimberly | Lea | Liv | Lotte | Lucia | Lupe | Mads | Maja | Marlene | Mathieu | Matthew | Maxim | Mia | Miguel | Mizuki | Naja | Nicole | Olivia | Penelope | Raveena | Ricardo | Ruben | Russell | Salli | Seoyeon | Takumi | Tatyana | Vicki | Vitoria | Zeina | Zhiyu | Aria | Ayanda |

Arlet | Hannah | Arthur | Daniel | Liam | Pedro | Kajal | Hiujin | Laura
| Elin | Ida | Suvi | Ola | Hala | Andres | Sergio | Remi | Adriano
| Thiago | Ruth | Stephen | Kazuha | Tomoko | Niamh | Sofie | Lisa |
Isabelle | Zayd | Danielle | Gregory | Burcu | Jitka | Sabrina | Jasmine
| Jihye | Ambre | Beatrice | Florian | Lennart | Lorenzo | Tiffany

必要：是

請求主體

請求接受採用 JSON 格式的下列資料。

[CloseStreamEvent](#)

指出輸入串流結束的事件。

類型：[CloseStreamEvent](#) 物件

必要：否

[TextEvent](#)

包含要合成內容的文字事件。

類型：[TextEvent](#) 物件

必要：否

回應語法

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

{
  "AudioEvent": {
    "AudioChunk": blob
  },
  "ServiceFailureException": {
  },
  "ServiceQuotaExceededException": {
  },
  "StreamClosedEvent": {
```

```
    "RequestCharacters": number
  },
  "ThrottlingException": {
  },
  "ValidationException": {
  }
}
```

回應元素

如果動作成功，則服務傳回 HTTP 200 回應。

服務會傳回下列 JSON 格式的資料。

[AudioEvent](#)

包含合成語音的音訊事件。

類型：[AudioEvent](#) 物件

[ServiceFailureException](#)

不明條件導致服務失敗。

類型：例外狀況

HTTP 狀態碼：500

[ServiceQuotaExceededException](#)

超出表示服務配額的例外狀況。

類型：例外狀況

HTTP 狀態碼：402

[StreamClosedEvent](#)

具有摘要資訊的事件，表示串流已關閉。

類型：[StreamClosedEvent](#) 物件

[ThrottlingException](#)

表示請求已調節的例外狀況。

類型：例外狀況

HTTP 狀態碼：400

ValidationException

表示輸入驗證失敗的例外狀況。

類型：例外狀況

HTTP 狀態碼：400

錯誤

ServiceFailureException

不明條件導致服務失敗。

HTTP 狀態碼：500

ServiceQuotaExceededException

請求會導致超過服務配額。

quotaCode

識別特定配額的配額代碼。

serviceCode

識別原始服務的服務代碼。

HTTP 狀態碼：402

ThrottlingException

請求因請求限流而遭拒。

throttlingReasons

說明請求受到限流原因的原因清單。

HTTP 狀態碼：400

ValidationException

輸入無法滿足 服務指定的限制條件。

fields

導致驗證錯誤的欄位。

reason

請求驗證失敗的原因。

HTTP 狀態碼：400

另請參閱

如需在其中一種語言特定 AWS SDKs 中使用此 API 的詳細資訊，請參閱下列內容：

- [AWS 命令列界面 V2](#)
- [AWS 適用於 .NET V4 的 SDK](#)
- [AWS 適用於 C++ 的 SDK](#)
- [AWS 適用於 Go 的 SDK v2](#)
- [AWS 適用於 Java V2 的 SDK](#)
- [AWS 適用於 JavaScript V3 的 SDK](#)
- [AWS 適用於 Kotlin 的 SDK](#)
- [AWS 適用於 PHP V3 的 SDK](#)
- [AWS 適用於 Python 的 SDK](#)
- [AWS 適用於 Ruby V3 的 SDK](#)

StartSpeechSynthesisTask

允許透過啟動新的 來建立非同步合成任務SpeechSynthesisTask。此操作需要語音合成所需的所有標準資訊，加上服務 Amazon S3 儲存貯體的名稱，以存放合成任務的輸出和兩個選用參數 (OutputS3KeyPrefix 和 SnsTopicArn)。合成任務建立後，此操作會傳回SpeechSynthesisTask物件，其中包含此任務的識別符以及目前狀態。SpeechSynthesisTask物件會在啟動非同步合成任務後 72 小時內可用。

請求語法

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

{
  "Engine": "string",
  "LanguageCode": "string",
  "LexiconNames": [ "string" ],
  "OutputFormat": "string",
  "OutputS3BucketName": "string",
  "OutputS3KeyPrefix": "string",
  "SampleRate": "string",
  "SnsTopicArn": "string",
  "SpeechMarkTypes": [ "string" ],
  "Text": "string",
  "TextType": "string",
  "VoiceId": "string"
}
```

URI 請求參數

請求不會使用任何 URI 參數。

請求主體

請求接受採用 JSON 格式的下列資料。

Engine

指定 Amazon Polly 在處理語音合成的輸入文字時要使用的引擎 (standardneural、long-form或 generative)。使用所選引擎不支援的語音會導致錯誤。

類型：字串

有效值:standard | neural | long-form | generative

必要：否

LanguageCode

語音合成請求的選用語言代碼。只有在使用雙語語音，例如 Aditi 時，這才是必要的，可用於印度英文 (en-IN) 或印地文 (hi-IN)。

如果使用雙語語音且未指定語言代碼，Amazon Polly 會使用雙語語音的預設語言。任何語音的預設語言都是 [DescribeVoices](#) 操作針對 LanguageCode 參數傳回的語言。例如，如果未指定語言代碼，Aditi 將使用印度英文，而不是印地文。

類型：字串

有效值:arb | cmn-CN | cy-GB | da-DK | de-DE | en-AU | en-GB | en-GB-WLS | en-IN | en-US | es-ES | es-MX | es-US | fr-CA | fr-FR | is-IS | it-IT | ja-JP | hi-IN | ko-KR | nb-NO | nl-NL | pl-PL | pt-BR | pt-PT | ro-RO | ru-RU | sv-SE | tr-TR | en-NZ | en-ZA | ca-ES | de-AT | yue-CN | ar-AE | fi-FI | en-IE | nl-BE | fr-BE | cs-CZ | de-CH | en-SG

必要：否

LexiconNames

您希望服務在合成期間套用的一或多個發音語彙名稱清單。只有在語彙的語言與語音的語言相同時，才會套用語彙。

類型：字串陣列

陣列成員：最多 5 個項目。

模式：[0-9A-Za-z]{1,20}

必要：否

OutputFormat

將編碼傳回輸出的格式。對於音訊串流，這會是 mp3、ogg_vorbis、ogg_opus、mu-law、a-law 或 pcm。對於語音標記，這會是 json。

類型：字串

有效值:json | mp3 | ogg_opus | ogg_vorbis | pcm | mulaw | alaw

必要：是

OutputS3BucketName

將輸出檔案儲存到的 Amazon S3 儲存貯體名稱。

類型：字串

模式：`^[a-z0-9][\.\-a-z0-9]{1,61}[a-z0-9]$`

必要：是

OutputS3KeyPrefix

輸出語音檔案的 Amazon S3 金鑰字首。

類型：字串

模式：`^[0-9a-zA-Z\^\!\\\_\\.\\*\\'\\(\\):;\\$@=+\\,\\?&]{0,800}$`

必要：否

SampleRate

以 Hz 指定的音訊頻率。

mp3 和 ogg_vorbis 的有效值為 "8000"、"16000"、"22050" 和 "24000"。標準語音的預設值為「22050」。神經語音的預設值為「24000」。長格式語音的預設值為「24000」。生成語音的預設值為「24000」。

pcm 的有效值為 "8000" 和 "16000" 預設值為 "16000"。

ogg_opus 的有效值為 "48000"。

mu-law 和 a-law 的有效值為 "8000"。

類型：字串

必要：否

SnsTopicArn

SNS 主題的 ARN 可選擇性地用於為語音合成任務提供狀態通知。

類型：字串

模式：`^arn:aws(-(cn|iso(-b)?|us-gov))?:sns:[a-z0-9_-]{1,50}:\d{12}:([a-zA-Z0-9_-]{1,251}([a-zA-Z0-9_-]{0,5}|\.fifo))$`

必要：否

[SpeechMarkTypes](#)

針對輸入文字傳回的語音標記類型。

類型：字串陣列

陣列成員：最多 4 個項目。

有效值:sentence | ssm1 | viseme | word

必要：否

[Text](#)

要合成的輸入文字。如果您將 ssm1 指定為 TextType，請遵循輸入文字的 SSML 格式。

類型：字串

必要：是

[TextType](#)

指定輸入文字是純文字還是 SSML。預設值為純文字。

類型：字串

有效值:ssm1 | text

必要：否

[Voiceld](#)

用於合成的語音 ID。

類型：字串

有效值:Aditi | Amy | Astrid | Bianca | Brian | Camila | Carla | Carmen | Celine | Chantal | Conchita | Cristiano | Dora | Emma | Enrique | Ewa | Filiz | Gabrielle | Geraint | Giorgio | Gwyneth | Hans | Ines | Ivy | Jacek | Jan | Joanna | Joey | Justin | Karl | Kendra | Kevin | Kimberly | Lea | Liv | Lotte | Lucia | Lupe | Mads | Maja | Marlene | Mathieu

| Matthew | Maxim | Mia | Miguel | Mizuki | Naja | Nicole | Olivia
| Penelope | Raveena | Ricardo | Ruben | Russell | Salli | Seoyeon |
Takumi | Tatyana | Vicki | Vitoria | Zeina | Zhiyu | Aria | Ayanda |
Arlet | Hannah | Arthur | Daniel | Liam | Pedro | Kajal | Hiujin | Laura
| Elin | Ida | Suvi | Ola | Hala | Andres | Sergio | Remi | Adriano
| Thiago | Ruth | Stephen | Kazuha | Tomoko | Niamh | Sofie | Lisa |
Isabelle | Zayd | Danielle | Gregory | Burcu | Jitka | Sabrina | Jasmine
| Jihye | Ambre | Beatrice | Florian | Lennart | Lorenzo | Tiffany

必要：是

回應語法

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

{
  "SynthesisTask": {
    "CreationTime": number,
    "Engine": "string",
    "LanguageCode": "string",
    "LexiconNames": [ "string" ],
    "OutputFormat": "string",
    "OutputUri": "string",
    "RequestCharacters": number,
    "SampleRate": "string",
    "SnsTopicArn": "string",
    "SpeechMarkTypes": [ "string" ],
    "TaskId": "string",
    "TaskStatus": "string",
    "TaskStatusReason": "string",
    "TextType": "string",
    "VoiceId": "string"
  }
}
```

回應元素

如果動作成功，則服務傳回 HTTP 200 回應。

服務會傳回下列 JSON 格式的資料。

SynthesisTask

SynthesisTask 物件，可提供新提交語音合成任務的相關資訊和屬性。

類型：[SynthesisTask](#) 物件

錯誤

EngineNotSupportedException

此引擎與您指定的語音不相容。選擇與引擎相容的新語音，或變更引擎並重新啟動操作。

HTTP 狀態碼：400

InvalidS3BucketException

提供的 Amazon S3 儲存貯體名稱無效。請檢查您輸入的 S3 儲存貯體命名需求，然後再試一次。

HTTP 狀態碼：400

InvalidS3KeyException

提供的 Amazon S3 金鑰字首無效。請提供有效的 S3 物件金鑰名稱。

HTTP 狀態碼：400

InvalidSampleRateException

指定的取樣率無效。

HTTP 狀態碼：400

InvalidSnsTopicArnException

提供的 SNS 主題 ARN 無效。請提供有效的 SNS 主題 ARN，然後再試一次。

HTTP 狀態碼：400

InvalidSsmlException

您提供的 SSML 無效。驗證 SSML 語法、標籤和值的拼寫，然後再試一次。

HTTP 狀態碼：400

LanguageNotSupportedException

Amazon Polly 目前在此容量中不支援指定的語言。

HTTP 狀態碼：400

LexiconNotFoundException

Amazon Polly 找不到指定的語彙。這可能是由於缺少語彙、其名稱拼寫錯誤或指定位於不同區域的語彙所致。

確認語彙存在、位於 區域 (請參閱[ListLexicons](#)) , 且您拼寫的名稱拼寫正確。然後再試一次。

HTTP 狀態碼：404

MarksNotSupportedForFormatException

OutputFormat 選取的 不支援語音標記。語音標記僅適用於 json 格式的內容。

HTTP 狀態碼：400

ServiceFailureException

不明條件導致服務失敗。

HTTP 狀態碼：500

SsmlMarksNotSupportedForTextTypeException

純文字類型輸入不支援 SSML 語音標記。

HTTP 狀態碼：400

TextLengthExceededException

"Text" 參數的值超過接受的限制。對於 SynthesizeSpeech API , 輸入文字的限制總計最多為 6000 個字元, 其中不超過 3000 個字元可以計費。對於 StartSpeechSynthesisTask API , 上限為 200,000 個字元, 其中不超過 100,000 個字元可以計費。SSML 標籤不計為收費字元。

HTTP 狀態碼：400

另請參閱

如需在其中一種語言特定 AWS SDKs 中使用此 API 的詳細資訊, 請參閱下列內容:

- [AWS 命令列界面 V2](#)
- [AWS 適用於 .NET V4 的 SDK](#)
- [AWS 適用於 C++ 的 SDK](#)

- [AWS 適用於 Go 的 SDK v2](#)
- [AWS 適用於 Java V2 的 SDK](#)
- [AWS 適用於 JavaScript V3 的 SDK](#)
- [AWS 適用於 Kotlin 的 SDK](#)
- [AWS 適用於 PHP V3 的 SDK](#)
- [AWS 適用於 Python 的 SDK](#)
- [AWS 適用於 Ruby V3 的 SDK](#)

SynthesizeSpeech

將 UTF-8 輸入、純文字或 SSML 合成到位元組串流。SSML 輸入必須是有效且格式正確的 SSML。除非使用音素映射，否則某些字母可能無法用於所有語音（例如，英文語音可能完全無法讀取 Cyrillic）。如需詳細資訊，請參閱[其運作方式](#)。

請求語法

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

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

URI 請求參數

請求不會使用任何 URI 參數。

請求主體

請求接受採用 JSON 格式的下列資料。

Engine

指定 Amazon Polly 在處理語音合成的輸入文字時要使用的引擎（standard、neural（long-form、或 generative））。提供您選取的語音支援的引擎。如果您未提供引擎，預設會選取標準引擎。如果標準引擎不支援所選的語音，這將導致錯誤。如需有關 Amazon Polly 語音和每個引擎可用語音的資訊，請參閱[可用語音](#)。

類型：字串

有效值: standard | neural | long-form | generative

必要：否

LanguageCode

合成語音請求的選用語言代碼。只有在使用雙語語音，例如 Aditi 時，這才是必要的，可用於印度英文 (en-IN) 或印地文 (hi-IN)。

如果使用雙語語音且未指定語言代碼，Amazon Polly 會使用雙語語音的預設語言。任何語音的預設語言都是 [DescribeVoices](#) 操作針對 LanguageCode 參數傳回的語言。例如，如果未指定語言代碼，Aditi 將使用印度英文，而不是印地文。

類型：字串

有效值:arb | cmn-CN | cy-GB | da-DK | de-DE | en-AU | en-GB | en-GB-WLS | en-IN | en-US | es-ES | es-MX | es-US | fr-CA | fr-FR | is-IS | it-IT | ja-JP | hi-IN | ko-KR | nb-NO | nl-NL | pl-PL | pt-BR | pt-PT | ro-RO | ru-RU | sv-SE | tr-TR | en-NZ | en-ZA | ca-ES | de-AT | yue-CN | ar-AE | fi-FI | en-IE | nl-BE | fr-BE | cs-CZ | de-CH | en-SG

必要：否

LexiconNames

您希望服務在合成期間套用的一或多個發音語彙名稱清單。只有在語彙的語言與語音的語言相同時，才會套用語彙。如需存放語彙的資訊，請參閱 [PutLexicon](#)。

類型：字串陣列

陣列成員：最多 5 個項目。

模式：[0-9A-Za-z]{1,20}

必要：否

OutputFormat

將編碼傳回輸出的格式。對於音訊串流，這會是 mp3、ogg_vorbis、ogg_opus、mu-law、a-law 或 pcm。對於語音標記，這會是 json。

使用 pcm 時，傳回的內容會以帶正負號的 16 位元、1 聲道（單聲道）、小端格式顯示音訊/pcm。

類型：字串

有效值:json | mp3 | ogg_opus | ogg_vorbis | pcm | mulaw | alaw

必要：是

SampleRate

以 Hz 指定的音訊頻率。

mp3 和 ogg_vorbis 的有效值為 "8000"、"16000"、"22050"、"24000"、"44100" 和 "48000"。標準語音的預設值為「22050」。神經語音的預設值為「24000」。長格式語音的預設值為「24000」。生成語音的預設值為「24000」。

pcm 的有效值為 "8000" 和 "16000" 預設值為 "16000"。

ogg_opus 的有效值為 "48000"。

mu-law 和 a-law 的有效值為 "8000"。

類型：字串

必要：否

SpeechMarkTypes

針對輸入文字傳回的語音標記類型。

類型：字串陣列

陣列成員：最多 4 個項目。

有效值:sentence | ssm1 | viseme | word

必要：否

Text

輸入要合成的文字。如果您將 指定ssml為 TextType，請遵循輸入文字的 SSML 格式。

類型：字串

必要：是

TextType

指定輸入文字是純文字還是 SSML。預設值為純文字。如需詳細資訊，請參閱[使用 SSML](#)。

類型：字串

有效值:ssml | text

必要：否

Voiceld

用於合成的語音 ID。您可以呼叫 [DescribeVoices](#) 操作，以取得可用的語音 IDs 清單。

類型：字串

有效值:Aditi | Amy | Astrid | Bianca | Brian | Camila | Carla | Carmen | Celine | Chantal | Conchita | Cristiano | Dora | Emma | Enrique | Ewa | Filiz | Gabrielle | Geraint | Giorgio | Gwyneth | Hans | Ines | Ivy | Jacek | Jan | Joanna | Joey | Justin | Karl | Kendra | Kevin | Kimberly | Lea | Liv | Lotte | Lucia | Lupe | Mads | Maja | Marlene | Mathieu | Matthew | Maxim | Mia | Miguel | Mizuki | Naja | Nicole | Olivia | Penelope | Raveena | Ricardo | Ruben | Russell | Salli | Seoyeon | Takumi | Tatyana | Vicki | Vitoria | Zeina | Zhiyu | Aria | Ayanda | Arlet | Hannah | Arthur | Daniel | Liam | Pedro | Kajal | Hiujin | Laura | Elin | Ida | Suvi | Ola | Hala | Andres | Sergio | Remi | Adriano | Thiago | Ruth | Stephen | Kazuha | Tomoko | Niamh | Sofie | Lisa | Isabelle | Zayd | Danielle | Gregory | Burcu | Jitka | Sabrina | Jasmine | Jihye | Ambre | Beatrice | Florian | Lennart | Lorenzo | Tiffany

必要：是

回應語法

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

AudioStream

回應元素

如果動作成功，則服務傳回 HTTP 200 回應。

回應會傳回下列 HTTP 標頭。

ContentType

指定音訊串流類型。這應該反映請求中的 `OutputFormat` 參數。

- 如果您以 mp3 身分請求 OutputFormat，則ContentType傳回的 是音訊/mpeg。
- 如果您以 ogg_vorbis 身分請求 OutputFormat，則ContentType傳回的 是音訊/混音。
- 如果您以 ogg_opus 身分請求 OutputFormat，則ContentType傳回的 是音訊/混音。
- 如果您以 pcm 身分請求OutputFormat，則ContentType傳回的 是已簽章的 16 位元、1 聲道 (單聲道)、小端格式的音訊/pcm。
- 如果您以 mu-law 身分請求 OutputFormat，則ContentType傳回的 是音訊/mulaw。
- 如果您以 a-law 身分請求 OutputFormat，則ContentType傳回的 為音訊/隔離。
- 如果您以 json 身分請求 OutputFormat，則ContentType傳回的 是 application/x-json-stream。

[RequestCharacters](#)

合成的字元數。

回應傳回以下內容作為 HTTP 主體。

[AudioStream](#)

包含合成語音的串流。

錯誤

EngineNotSupportedException

此引擎與您指定的語音不相容。選擇與引擎相容的新語音，或變更引擎並重新啟動操作。

HTTP 狀態碼：400

InvalidSampleRateException

指定的取樣率無效。

HTTP 狀態碼：400

InvalidSsmlException

您提供的 SSML 無效。驗證 SSML 語法、標籤和值的拼寫，然後再試一次。

HTTP 狀態碼：400

LanguageNotSupportedException

Amazon Polly 目前在此容量中不支援指定的語言。

HTTP 狀態碼：400

LexiconNotFoundExcepion

Amazon Polly 找不到指定的語彙。這可能是由於缺少語彙、其名稱拼寫錯誤或指定位於不同區域的語彙所致。

確認語彙是否存在、位於 區域 (請參閱[ListLexicons](#))，而且您拼寫的名稱拼寫正確。然後再試一次。

HTTP 狀態碼：404

MarksNotSupportedForFormatException

OutputFormat 選取的 不支援語音標記。語音標記僅適用於 json 格式的內容。

HTTP 狀態碼：400

ServiceFailureException

不明條件導致服務失敗。

HTTP 狀態碼：500

SsmlMarksNotSupportedForTextTypeException

純文字類型輸入不支援 SSML 語音標記。

HTTP 狀態碼：400

TextLengthExceededException

"Text" 參數的值超過接受的限制。對於 SynthesizeSpeech API，輸入文字的限制總計最多為 6000 個字元，其中不超過 3000 個字元可以計費。對於 StartSpeechSynthesisTask API，上限為 200,000 個字元，其中不超過 100,000 個字元可以計費。SSML 標籤不計為收費字元。

HTTP 狀態碼：400

另請參閱

如需在其中一種語言特定 AWS SDKs中使用此 API 的詳細資訊，請參閱下列內容：

- [AWS 命令列界面 V2](#)
- [AWS 適用於 .NET V4 的 SDK](#)
- [AWS 適用於 C++ 的 SDK](#)
- [AWS 適用於 Go 的 SDK v2](#)
- [AWS 適用於 Java V2 的 SDK](#)
- [AWS 適用於 JavaScript V3 的 SDK](#)
- [AWS 適用於 Kotlin 的 SDK](#)
- [AWS 適用於 PHP V3 的 SDK](#)
- [AWS 適用於 Python 的 SDK](#)
- [AWS 適用於 Ruby V3 的 SDK](#)

資料類型

目前支援下列資料類型：

- [AudioEvent](#)
- [CloseStreamEvent](#)
- [FlushStreamConfiguration](#)
- [Lexicon](#)
- [LexiconAttributes](#)
- [LexiconDescription](#)
- [StartSpeechSynthesisStreamActionStream](#)
- [StartSpeechSynthesisStreamEventStream](#)
- [StreamClosedEvent](#)
- [SynthesisTask](#)
- [TextEvent](#)
- [ThrottlingReason](#)
- [ValidationExceptionField](#)
- [Voice](#)

AudioEvent

包含合成音訊資料的區塊。

目錄

AudioChunk

以 `OutputFormat` 參數指定的格式編碼的合成音訊資料區塊。

類型：Base64 編碼的二進位資料物件

必要：否

另請參閱

如需在其中一種語言特定 AWS SDKs 中使用此 API 的詳細資訊，請參閱下列內容：

- [AWS 適用於 C++ 的 SDK](#)
- [AWS 適用於 Java V2 的 SDK](#)
- [AWS 適用於 Ruby V3 的 SDK](#)

CloseStreamEvent

指出輸入串流的結尾。傳送此事件後，輸入串流將會關閉，並傳回所有音訊。

目錄

此例外狀況結構的成員取決於內容。

另請參閱

如需在其中一種語言特定 AWS SDKs中使用此 API 的詳細資訊，請參閱下列內容：

- [AWS 適用於 C++ 的 SDK](#)
- [AWS 適用於 Java V2 的 SDK](#)
- [AWS 適用於 Ruby V3 的 SDK](#)

FlushStreamConfiguration

控制何時在輸出串流上傳送合成音訊資料的組態。

目錄

Force

指定是否強制合成引擎立即將緩衝音訊資料寫入輸出串流。

類型：布林值

必要：否

另請參閱

如需在其中一種語言特定 AWS SDKs 中使用此 API 的詳細資訊，請參閱下列內容：

- [AWS 適用於 C++ 的 SDK](#)
- [AWS 適用於 Java V2 的 SDK](#)
- [AWS 適用於 Ruby V3 的 SDK](#)

Lexicon

提供字串格式的語彙名稱和語彙內容。如需詳細資訊，請參閱[發音語彙規格 \(PLS\) 1.0 版](#)。

目錄

Content

字串格式的語彙內容。語彙的內容必須為 PLS 格式。

類型：字串

必要：否

Name

語彙的名稱。

類型：字串

模式：`[0-9A-Za-z]{1,20}`

必要：否

另請參閱

如需在其中一種語言特定 AWS SDKs 中使用此 API 的詳細資訊，請參閱下列內容：

- [AWS 適用於 C++ 的 SDK](#)
- [AWS 適用於 Java V2 的 SDK](#)
- [AWS 適用於 Ruby V3 的 SDK](#)

LexiconAttributes

包含描述語彙的中繼資料，例如 lexemes 數目、語言程式碼等。如需詳細資訊，請參閱[管理語彙](#)。

目錄

Alphabet

語彙中使用的音標字母。有效值為 ipa 和 x-sampa。

類型：字串

必要：否

LanguageCode

語彙適用的語言程式碼。語言代碼如「en」的語彙會套用至所有英文語言 (en-GB、en-US、en-AUS、en-WLS 等)。

類型：字串

有效值:arb | cmn-CN | cy-GB | da-DK | de-DE | en-AU | en-GB | en-GB-WLS | en-IN | en-US | es-ES | es-MX | es-US | fr-CA | fr-FR | is-IS | it-IT | ja-JP | hi-IN | ko-KR | nb-NO | nl-NL | pl-PL | pt-BR | pt-PT | ro-RO | ru-RU | sv-SE | tr-TR | en-NZ | en-ZA | ca-ES | de-AT | yue-CN | ar-AE | fi-FI | en-IE | nl-BE | fr-BE | cs-CZ | de-CH | en-SG

必要：否

LastModified

語彙上次修改的日期（時間戳記值）。

類型：Timestamp

必要：否

LexemesCount

語彙中的 lexemes 數目。

類型：整數

必要：否

LexiconArn

語彙的 Amazon Resource Name (ARN)。

類型：字串

必要：否

Size

語彙的總大小，以字元為單位。

類型：整數

必要：否

另請參閱

如需在其中一種語言特定 AWS SDKs 中使用此 API 的詳細資訊，請參閱下列內容：

- [AWS 適用於 C++ 的 SDK](#)
- [AWS 適用於 Java V2 的開發套件](#)
- [AWS 適用於 Ruby V3 的 SDK](#)

LexiconDescription

描述語彙的內容。

目錄

Attributes

提供語彙中繼資料。

類型：[LexiconAttributes](#) 物件

必要：否

Name

語彙的名稱。

類型：字串

模式：`[0-9A-Za-z]{1,20}`

必要：否

另請參閱

如需在其中一種語言特定 AWS SDKs中使用此 API 的詳細資訊，請參閱下列內容：

- [AWS 適用於 C++ 的 SDK](#)
- [AWS 適用於 Java V2 的開發套件](#)
- [AWS 適用於 Ruby V3 的 SDK](#)

StartSpeechSynthesisStreamActionStream

用於傳送輸入和控制事件以管理雙向語音合成的傳入事件串流。

目錄

CloseStreamEvent

指出輸入串流結束的事件。

類型：[CloseStreamEvent](#) 物件

必要：否

TextEvent

包含要合成內容的文字事件。

類型：[TextEvent](#) 物件

必要：否

另請參閱

如需在其中一種語言特定 AWS SDKs中使用此 API 的詳細資訊，請參閱下列內容：

- [AWS 適用於 C++ 的 SDK](#)
- [AWS 適用於 Java V2 的 SDK](#)
- [AWS 適用於 Ruby V3 的 SDK](#)

StartSpeechSynthesisStreamEventStream

包含合成音訊資料和串流狀態事件的傳出事件串流。

目錄

AudioEvent

包含合成語音的音訊事件。

類型：[AudioEvent](#) 物件

必要：否

ServiceFailureException

不明條件導致服務失敗。

類型：例外狀況

HTTP 狀態碼：500

必要：否

ServiceQuotaExceededException

將超過表示服務配額的例外狀況。

類型：例外狀況

HTTP 狀態碼：402

必要：否

StreamClosedEvent

事件，其中包含摘要資訊，指出串流已關閉。

類型：[StreamClosedEvent](#) 物件

必要：否

ThrottlingException

表示請求已調節的例外狀況。

類型：例外狀況

HTTP 狀態碼：400

必要：否

ValidationException

表示輸入驗證失敗的例外狀況。

類型：例外狀況

HTTP 狀態碼：400

必要：否

另請參閱

如需在其中一種語言特定 AWS SDKs 中使用此 API 的詳細資訊，請參閱下列內容：

- [AWS 適用於 C++ 的 SDK](#)
- [AWS 適用於 Java V2 的 SDK](#)
- [AWS 適用於 Ruby V3 的 SDK](#)

StreamClosedEvent

表示合成串流已關閉並提供摘要資訊。

目錄

RequestCharacters

在串流工作階段期間合成的字元總數。

類型：整數

必要：否

另請參閱

如需在其中一種語言特定 AWS SDKs 中使用此 API 的詳細資訊，請參閱下列內容：

- [AWS 適用於 C++ 的 SDK](#)
- [AWS 適用於 Java V2 的 SDK](#)
- [AWS 適用於 Ruby V3 的 SDK](#)

SynthesisTask

SynthesisTask 物件，可提供語音合成任務的相關資訊。

目錄

CreationTime

合成任務啟動時間的時間戳記。

類型：Timestamp

必要：否

Engine

指定 Amazon Polly 在處理語音合成的輸入文字時要使用的引擎 (standardneural、long-form 或 generative)。使用所選引擎不支援的語音會導致錯誤。

類型：字串

有效值:standard | neural | long-form | generative

必要：否

LanguageCode

合成任務的選用語言程式碼。只有在使用雙語語音，例如 Aditi 時，這才是必要的，可用於印度英文 (en-IN) 或印地文 (hi-IN)。

如果使用雙語語音且未指定語言代碼，Amazon Polly 會使用雙語語音的預設語言。任何語音的預設語言都是 [DescribeVoices](#) 操作針對 LanguageCode 參數傳回的語言。例如，如果未指定語言代碼，Aditi 將使用印度英文，而不是印地文。

類型：字串

有效值:arb | cmn-CN | cy-GB | da-DK | de-DE | en-AU | en-GB | en-GB-WLS | en-IN | en-US | es-ES | es-MX | es-US | fr-CA | fr-FR | is-IS | it-IT | ja-JP | hi-IN | ko-KR | nb-NO | nl-NL | pl-PL | pt-BR | pt-PT | ro-RO | ru-RU | sv-SE | tr-TR | en-NZ | en-ZA | ca-ES | de-AT | yue-CN | ar-AE | fi-FI | en-IE | nl-BE | fr-BE | cs-CZ | de-CH | en-SG

必要：否

LexiconNames

您希望服務在合成期間套用的一或多個發音語彙名稱清單。只有在語彙的語言與語音的語言相同時，才會套用語彙。

類型：字串陣列

陣列成員：最多 5 個項目。

模式：`[0-9A-Za-z]{1,20}`

必要：否

OutputFormat

將編碼傳回輸出的格式。對於音訊串流，這會是 mp3、ogg_vorbis、ogg_opus、mu-law、a-law 或 pcm。對於語音標記，這會是 json。

類型：字串

有效值: json | mp3 | ogg_opus | ogg_vorbis | pcm | mulaw | alaw

必要：否

OutputUri

輸出語音檔案的路徑。

類型：字串

必要：否

RequestCharacters

合成的計費字元數。

類型：整數

必要：否

SampleRate

以 Hz 指定的音訊頻率。

mp3 和 ogg_vorbis 的有效值為 "8000"、"16000"、"22050" 和 "24000"。標準語音的預設值為「22050」。神經語音的預設值為「24000」。長格式語音的預設值為「24000」。生成語音的預設值為「24000」。

pcm 的有效值為 "8000" 和 "16000" 預設值為 "16000"。

ogg_opus 的有效值為 "48000"。

mu-law 和 a-law 的有效值為 "8000"。

類型：字串

必要：否

SnsTopicArn

SNS 主題的 ARN 可選擇性地用於為語音合成任務提供狀態通知。

類型：字串

模式：`^arn:aws(-(cn|iso(-b)?|us-gov))?:sns:[a-z0-9_-]{1,50}:\d{12}:[a-zA-Z0-9_-]{1,251}([a-zA-Z0-9_-]{0,5}|\.fifo)$`

必要：否

SpeechMarkTypes

針對輸入文字傳回的語音標記類型。

類型：字串陣列

陣列成員：最多 4 個項目。

有效值:sentence | ssm1 | viseme | word

必要：否

TaskId

Amazon Polly 為語音合成任務產生的識別符。

類型：字串

模式：`^[a-zA-Z0-9_-]{1,100}$`

必要：否

TaskStatus

個別語音合成任務的目前狀態。

類型：字串

有效值:scheduled | inProgress | completed | failed

必要：否

TaskStatusReason

特定語音合成任務目前狀態的原因，包括任務失敗時的錯誤。

類型：字串

必要：否

TextType

指定輸入文字是純文字還是 SSML。預設值為純文字。

類型：字串

有效值:ssml | text

必要：否

Voiceld

用於合成的語音 ID。

類型：字串

有效值:Aditi | Amy | Astrid | Bianca | Brian | Camila | Carla | Carmen | Celine | Chantal | Conchita | Cristiano | Dora | Emma | Enrique | Ewa | Filiz | Gabrielle | Geraint | Giorgio | Gwyneth | Hans | Ines | Ivy | Jacek | Jan | Joanna | Joey | Justin | Karl | Kendra | Kevin | Kimberly | Lea | Liv | Lotte | Lucia | Lupe | Mads | Maja | Marlene | Mathieu | Matthew | Maxim | Mia | Miguel | Mizuki | Naja | Nicole | Olivia | Penelope | Raveena | Ricardo | Ruben | Russell | Salli | Seoyeon | Takumi | Tatyana | Vicki | Vitoria | Zeina | Zhiyu | Aria | Ayanda | Arlet | Hannah | Arthur | Daniel | Liam | Pedro | Kajal | Hiujin | Laura | Elin | Ida | Suvi | Ola | Hala | Andres | Sergio | Remi | Adriano | Thiago | Ruth | Stephen | Kazuha | Tomoko | Niamh | Sofie | Lisa | Isabelle | Zayd | Danielle | Gregory | Burcu | Jitka | Sabrina | Jasmine | Jihye | Ambre | Beatrice | Florian | Lennart | Lorenzo | Tiffany

必要：否

另請參閱

如需在其中一種語言特定 AWS SDKs 中使用此 API 的詳細資訊，請參閱下列內容：

- [AWS 適用於 C++ 的 SDK](#)
- [AWS 適用於 Java V2 的 SDK](#)
- [AWS 適用於 Ruby V3 的 SDK](#)

TextEvent

包含要合成為語音的文字內容。

目錄

Text

要合成的文字內容。如果您將 指定ssml為 TextType，請遵循輸入文字的 SSML 格式。

類型：字串

必要：是

FlushStreamConfiguration

用於控制何時合成音訊排清至輸出串流的組態。

類型：[FlushStreamConfiguration](#) 物件

必要：否

TextType

指定輸入文字是純文字還是 SSML。預設：純文字。

類型：字串

有效值:ssml | text

必要：否

另請參閱

如需在其中一種語言特定 AWS SDKs中使用此 API 的詳細資訊，請參閱下列內容：

- [AWS 適用於 C++ 的 SDK](#)
- [AWS 適用於 Java V2 的 SDK](#)
- [AWS 適用於 Ruby V3 的 SDK](#)

ThrottlingReason

提供特定限流原因的相關資訊。

目錄

reason

解釋請求被限流原因的原因代碼。

類型：字串

必要：否

resource

導致限流的資源。

類型：字串

必要：否

另請參閱

如需在其中一種語言特定 AWS SDKs中使用此 API 的詳細資訊，請參閱下列內容：

- [AWS 適用於 C++ 的 SDK](#)
- [AWS 適用於 Java V2 的 SDK](#)
- [AWS 適用於 Ruby V3 的 SDK](#)

ValidationExceptionField

驗證失敗之欄位的相關資訊。

目錄

message

說明欄位驗證失敗原因的訊息。

類型：字串

必要：是

name

驗證失敗的欄位名稱。

類型：字串

必要：是

另請參閱

如需在其中一種語言特定 AWS SDKs中使用此 API 的詳細資訊，請參閱下列內容：

- [AWS 適用於 C++ 的 SDK](#)
- [AWS 適用於 Java V2 的 SDK](#)
- [AWS 適用於 Ruby V3 的 SDK](#)

Voice

語音的描述。

目錄

AdditionalLanguageCodes

除了預設語言之外，指定語音可使用的其他語言代碼。

例如，Aditi 的預設語言是印度英文 (en-IN)，因為它最初用於該語言。由於 Aditi 在印度英文和印地文都是雙語且流暢，因此此參數會顯示程式碼 hi-IN。

類型：字串陣列

有效值:arb | cmn-CN | cy-GB | da-DK | de-DE | en-AU | en-GB | en-GB-WLS | en-IN | en-US | es-ES | es-MX | es-US | fr-CA | fr-FR | is-IS | it-IT | ja-JP | hi-IN | ko-KR | nb-NO | nl-NL | pl-PL | pt-BR | pt-PT | ro-RO | ru-RU | sv-SE | tr-TR | en-NZ | en-ZA | ca-ES | de-AT | yue-CN | ar-AE | fi-FI | en-IE | nl-BE | fr-BE | cs-CZ | de-CH | en-SG

必要：否

Gender

語音的性別。

類型：字串

有效值:Female | Male

必要：否

Id

Amazon Polly 指派的語音 ID。這是您在呼叫 SynthesizeSpeech 操作時指定的 ID。

類型：字串

有效值:Aditi | Amy | Astrid | Bianca | Brian | Camila | Carla | Carmen | Celine | Chantal | Conchita | Cristiano | Dora | Emma | Enrique | Ewa | Filiz | Gabrielle | Geraint | Giorgio | Gwyneth | Hans | Ines | Ivy |

Jacek | Jan | Joanna | Joey | Justin | Karl | Kendra | Kevin | Kimberly
 | Lea | Liv | Lotte | Lucia | Lupe | Mads | Maja | Marlene | Mathieu
 | Matthew | Maxim | Mia | Miguel | Mizuki | Naja | Nicole | Olivia
 | Penelope | Raveena | Ricardo | Ruben | Russell | Salli | Seoyeon |
 Takumi | Tatyana | Vicki | Vitoria | Zeina | Zhiyu | Aria | Ayanda |
 Arlet | Hannah | Arthur | Daniel | Liam | Pedro | Kajal | Hiujin | Laura
 | Elin | Ida | Suvi | Ola | Hala | Andres | Sergio | Remi | Adriano
 | Thiago | Ruth | Stephen | Kazuha | Tomoko | Niamh | Sofie | Lisa |
 Isabelle | Zayd | Danielle | Gregory | Burcu | Jitka | Sabrina | Jasmine
 | Jihye | Ambre | Beatrice | Florian | Lennart | Lorenzo | Tiffany

必要：否

LanguageCode

語音的語言代碼。

類型：字串

有效值:arb | cmn-CN | cy-GB | da-DK | de-DE | en-AU | en-GB | en-GB-WLS |
 en-IN | en-US | es-ES | es-MX | es-US | fr-CA | fr-FR | is-IS | it-IT |
 ja-JP | hi-IN | ko-KR | nb-NO | nl-NL | pl-PL | pt-BR | pt-PT | ro-RO |
 ru-RU | sv-SE | tr-TR | en-NZ | en-ZA | ca-ES | de-AT | yue-CN | ar-AE |
 fi-FI | en-IE | nl-BE | fr-BE | cs-CZ | de-CH | en-SG

必要：否

LanguageName

英文語言的人類可讀名稱。

類型：字串

必要：否

Name

語音的名稱（例如 Salli、Kendra 等）。這提供您可能會在應用程式中顯示的人類可讀語音名稱。

類型：字串

必要：否

SupportedEngines

指定指定語音支援哪些引擎 standard(neural、long-form或 generative)。

類型：字串陣列

有效值:standard | neural | long-form | generative

必要：否

另請參閱

如需在其中一種語言特定 AWS SDKs中使用此 API 的詳細資訊，請參閱下列內容：

- [AWS 適用於 C++ 的 SDK](#)
- [AWS 適用於 Java V2 的 SDK](#)
- [AWS 適用於 Ruby V3 的 SDK](#)

常見錯誤類型

本節列出 AWS 此服務可能傳回的常見錯誤類型。並非所有 服務都會傳回此處列出的所有錯誤類型。如需此服務之 API 動作的特定錯誤，請參閱該 API 動作的主題。

AccessDeniedException

您沒有執行此動作的許可。確認您的 IAM 政策包含必要的許可。

HTTP 狀態碼：403

ExpiredTokenException

請求中包含的安全字符已過期。請求新的安全字符，然後再試一次。

HTTP 狀態碼：403

IncompleteSignature

請求簽章不符合 AWS 標準。確認您使用的是有效的 AWS 登入資料，而且您的請求格式正確。如果您使用的是 開發套件，請確保其為最新版本。

HTTP 狀態碼：403

InternalFailure

由於內部伺服器問題，目前無法處理請求。請稍後再試。如果問題仍然存在，請聯絡 AWS Support。

HTTP 狀態碼：500

MalformedHttpRequestException

無法處理請求內文。這通常發生在無法使用指定的內容編碼演算法解壓縮請求內文時。確認內容編碼標頭符合使用的壓縮格式。

HTTP 狀態碼：400

NotAuthorized

您沒有執行此動作的許可。確認您的 IAM 政策包含必要的許可。

HTTP 狀態碼：401

OptInRequired

AWS 您的帳戶需要訂閱此服務。確認您已在帳戶中啟用 服務。

HTTP 狀態碼：403

RequestAbortedException

在傳回回應之前，請求已中止。這通常發生在用戶端關閉連線時。

HTTP 狀態碼：400

RequestEntityTooLargeException

請求實體太大。減少請求內文的大小，然後再試一次。

HTTP 狀態碼：413

RequestTimeoutException

請求逾時。伺服器未在預期的時間範圍內收到完整的請求。請再試一次。

HTTP 狀態碼：408

ServiceUnavailable

此服務暫時無法使用。請稍後再試。

HTTP 狀態碼：503

ThrottlingException

您的請求率太高。AWS SDKs 會自動重試收到此例外狀況的請求。減少請求的頻率。

HTTP 狀態碼：400

UnknownOperationException

無法辨識動作或操作。確認動作名稱拼寫正確，且您所使用的 API 版本支援此名稱。

HTTP 狀態碼：404

UnrecognizedClientException

您提供的 X.509 憑證或 AWS 存取金鑰 ID 不存在於我們的記錄中。確認您使用的是有效的登入資料，而且尚未過期。

HTTP 狀態碼：403

ValidationError

輸入不符合必要的格式或限制條件。檢查是否包含所有必要參數，以及值是否有效。

HTTP 狀態碼：400

常見參數

以下清單內含所有動作用來簽署 Signature 第 4 版請求的參數以及查詢字串。任何專屬於特定動作的參數則列於該動作的主題中。如需 Signature 第 4 版的詳細資訊，請參閱《IAM 使用者指南》中的[簽署 AWS API 請求](#)。

X-Amz-Algorithm

建立請求簽章時所使用的雜湊演算法。

條件：當您在查詢字串中而非 HTTP 授權標頭中納入驗證資訊時，應指定此參數。

類型：字串

有效值:AWS4-HMAC-SHA256

必要：有條件

X-Amz-Credential

憑證範圍值，此為一個字串，其中包含您的存取金鑰、日期、您的目標區域、您請求的服務，以及終止字串 (“aws4_request”)。值以下列格式表示：access_key/YYYYMMDD/region/service/aws4_request。

如需詳細資訊，請參閱《IAM 使用者指南》中的[建立已簽署的 AWS API 請求](#)。

條件：當您在查詢字串中而非 HTTP 授權標頭中納入驗證資訊時，應指定此參數。

類型：字串

必要：有條件

X-Amz-Date

用來建立簽署的日期。格式必須是 ISO 8601 基本格式 (YYYYMMDD'T'HHMMSS'Z')。例如，以下日期時間是有效的 X-Amz-Date 值：20120325T120000Z

條件：對所有請求而言，X-Amz-Date 皆為選用，可用來覆寫用於簽署請求的日期。如果規定日期標頭採用 ISO 8601 基本格式，則不需要 X-Amz-Date。當使用 X-Amz-Date 時，其一律會覆寫日期標頭的值。如需詳細資訊，請參閱《IAM 使用者指南》中的[AWS API 請求簽章的元素](#)。

類型：字串

必要：有條件

X-Amz-Security-Token

透過呼叫 AWS Security Token Service () 取得的臨時安全字串 AWS STS。如需支援 AWS STS 的臨時安全憑證的服務清單，請參閱《IAM 使用者指南》中的[可搭配 IAM 運作的 AWS 服務](#)。

條件：如果您使用來自 的臨時安全登入資料 AWS STS，則必須包含安全字串。

類型：字串

必要：有條件

X-Amz-Signature

指定從要簽署的字串和衍生的簽署金鑰中計算出的十六進位編碼簽章。

條件：當您在查詢字串中而非 HTTP 授權標頭中納入驗證資訊時，應指定此參數。

類型：字串

必要：有條件

X-Amz-SignedHeaders

指定納入作為標準請求一部分的所有 HTTP 標頭。如需指定已簽章標頭的詳細資訊，請參閱《IAM 使用者指南》中的[建立已簽章的 AWS API 請求](#)。

條件：當您在查詢字串中而非 HTTP 授權標頭中納入驗證資訊時，應指定此參數。

類型：字串

必要：有條件

Amazon Polly 的文件歷史記錄

下表說明 Amazon Polly 開發人員指南每個版本的重要變更。如需有關此文件更新的通知，您可以訂閱 RSS 訂閱源。

- 文件最近更新時間：2026 年 5 月 28 日

變更	描述	日期
針對神經語音新增區域	Amazon Polly 現已在亞太區域（泰國）AWS 區域提供。此區域支援神經 TTS (NTTS) 語音。如需詳細資訊，請參閱 神經語音 。	2026 年 5 月 28 日
針對生成語音新增區域	Amazon Polly 生成語音和雙向串流現在可在其他 AWS 區域使用：歐洲（蘇黎世）。如需詳細資訊，請參閱 生成式語音 。	2026 年 5 月 28 日
雙向串流可在新區域使用	Amazon Polly 雙向串流現在可在兩個其他 AWS 區域使用：歐洲（倫敦）和加拿大（中部）。如需詳細資訊，請參閱 生成式語音 。	2026 年 4 月 20 日
已更新受管政策	Amazon Polly 已更新受管政策 AmazonPollyReadOnlyAccess，以包含雙向串流語音合成的許可。如需詳細資訊，請參閱 Amazon Polly AWS 受管政策的更新 。	2026 年 3 月 19 日
新增生成text-to-speech的新語音	Amazon Polly 現在提供 10 個額外的生成語音：Brian、Aria、Jasmine	2026 年 3 月 19 日

、Workday、Ambre、Florin、Sabrina、Lennart、Beatrice、Lorenzo。如需[生成 TTS 語音](#)的清單，請參閱生成語音。

[針對生成語音新增新區域](#)

Amazon Polly 生成語音現在可在兩個其他 AWS 區域使用：歐洲（倫敦）和加拿大（中部）。如需詳細資訊，請參閱[生成式語音](#)。

2026 年 3 月 19 日

[針對生成語音新增區域](#)

Amazon Polly 生成語音現在可在其他三個 AWS 區域使用：亞太區域（東京）、亞太區域（首爾）和亞太區域（新加坡）。如需詳細資訊，請參閱[生成式語音](#)。

2025 年 11 月 18 日

[新增生成text-to-speech的新語音](#)

Amazon Polly 現在提供六個額外的生成語音：Seoyeon、Camila、Hannah、Niamh、Laura 和 Lisa。如需[生成 TTS 語音](#)的清單，請參閱生成語音。

2025 年 11 月 14 日

[針對神經語音新增區域](#)

Amazon Polly 現已在歐洲（蘇黎世）AWS 區域提供。此區域支援神經 TTS (NTTS) 語音。如需詳細資訊，請參閱[神經語音](#)。

2025 年 10 月 20 日

[新增生成text-to-speech的新語音](#)

Amazon Polly 現在提供七種額外的生成語音：Salli、Isabelle、Céline、Liam、Gabrielle、Ola 和 Ewa。如需[生成 TTS 語音](#)的清單，請參閱生成語音。

2025 年 8 月 26 日

針對神經和標準語音新增區域

Amazon Polly 現已在亞太區域（馬來西亞）AWS 區域提供。此區域支援神經 TTS (NTTS) 和標準語音。如需詳細資訊，請參閱[神經語音](#)和[標準語音](#)。

2025 年 3 月 27 日

新增神經text-to-speech的新語音

Amazon Polly 現在提供了一個額外的韓文語音：Jihye。如需 NTTS [語音清單](#)，請參閱[神經語音](#)。

2025 年 3 月 26 日

針對神經和標準語音新增區域

Amazon Polly 現已在歐洲（西班牙）AWS 區域提供。此區域支援神經 TTS (NTTS) 和標準語音。如需詳細資訊，請參閱[神經語音](#)和[標準語音](#)。

2025 年 2 月 18 日

針對神經text-to-speech新增語音

Amazon Polly 現在提供額外的英文（新加坡）語音：Jasmine。如需 NTTS [語音清單](#)，請參閱[神經語音](#)。

2025 年 2 月 11 日

新增生成text-to-speech的新語音

Amazon Polly 現在提供七種新的生成語音：Pedro、Andrés、Sergio、Daniel、Kajal、Rémi、Bianca。如需[生成 TTS 語音](#)的清單，請參閱生成語音。

2024 年 11 月 21 日

新增的長格式語音

Amazon Polly 現在提供更長格式的語音。新增了一個額外的英文（美國）和兩個新的西班牙文（西班牙）長格式語音：Patrick、Alba 和 Raúl。如需所有[長形語音](#)的清單，請參閱長形語音。

2024 年 11 月 14 日

[新增生成text-to-speech的新語音](#)

Amazon Polly 現在提供六個新的生成語音：Ayanda、Léa、Lucia、Mía、Lupe 和 Vicki。如需[生成 TTS 語音](#)的清單，請參閱生成語音。

2024 年 11 月 6 日

[新增生成text-to-speech的新語音](#)

Amazon Polly 現在提供四種新的生成語音：Olivia、Danile、Joanna 和 Stephen。如需[生成 TTS 語音](#)的清單，請參閱生成語音。

2024 年 10 月 10 日

[針對text-to-speech新增語言](#)

Amazon Polly 現在提供兩種新的 TTS 語言：捷克文 (cs-CZ) 和德文（瑞士）(de-CH)。如需[語言清單](#)，請參閱[支援的語言](#)。

2024 年 9 月 26 日

[針對 NTTS 新增的語音](#)

Amazon Polly 現在提供兩種新的 NTTS 語音：Jitka 和 Sabrina。如需 NTTS [語音清單](#)，請參閱[神經語音](#)。

2024 年 8 月 27 日

[已新增生成式語音引擎](#)

Amazon Polly 現在提供專為較長內容設計的生成式語音引擎，在生成式變體中具有三個英文語音：Amy、Matthew 和 Ruth。如需詳細資訊，請參閱[生成語音](#)。

2024 年 3 月 28 日

[針對 NTTS 新增語音](#)

Amazon Polly 現在提供 NTTS 土耳其文語音 Burcu。如需 NTTS [語音清單](#)，請參閱[神經語音](#)。

2024 年 2 月 14 日

新增新的長格式語音引擎	Amazon Polly 現在提供專為較長內容設計的長格式語音引擎，具有三種 en-US 語音：Danile、Gregory 和 Ruth。如需詳細資訊， 請參閱長形語音 。	2023 年 11 月 16 日
針對 NTTS 新增的語音	Amazon Polly 現在提供兩種新的 NTTS 美式英文語音：Danile 和 Gregory。如需 NTTS 語音清單 ， 請參閱神經語音 。	2023 年 10 月 5 日
Amazon Polly for Windows	將不再支援 Amazon Polly Windows 語音應用程式介面 (SAPI) 外掛程式。	2023 年 9 月 26 日
已更新 Amazon Polly 的配額指引	已更新 Amazon Polly 配額指南。新增範例並釐清詞彙。如需更新， 請參閱 Amazon Polly 中的配額 。	2023 年 8 月 17 日
針對 NTTS 新增語音	Amazon Polly 現在提供 Gulf Arabic NTTS 語音 Zayd。如需 NTTS 語音清單 ， 請參閱神經語音 。	2023 年 8 月 16 日
針對 NTTS 新增語音	Amazon Polly 現在提供比利時法文 NTTS 語音 Isabelle。如需 NTTS 語音清單 ， 請參閱神經語音 。	2023 年 8 月 1 日
針對 NTTS 新增語音	Amazon Polly 現在提供比利時荷蘭文（瑞典）NTTS 語音 Lisa。如需 NTTS 語音清單 ， 請參閱神經語音 。	2023 年 6 月 7 日

針對 NTTS 新增的語音	Amazon Polly 現在提供兩種新的 NTTS 語音：愛爾蘭英文 (Niamh) 和丹麥文 (Sofie)。如需 NTTS 語音清單 ，請參閱 神經語音 。	2023 年 5 月 30 日
已更新 Amazon Polly 的 IAM 指引	更新了指南以符合 IAM 最佳實務。如需更多詳細資訊，請參閱 IAM 中的安全最佳實務 。	2023 年 4 月 19 日
WordPress 更新	將不再支援 Amazon Polly WordPress 外掛程式。	2023 年 4 月 6 日
已新增區域	Amazon Polly 現已在亞太區域（大阪）AWS 區域提供。此區域支援神經 TTS (NTTS)。如需詳細資訊，請參閱 支援 NTTS 的區域清單的功能和區域相容性 。	2023 年 4 月 5 日
針對 NTTS 新增的語音	Amazon Polly 現在提供兩種新的日文 NTTS 語音：Kazuha 和 Tomoko。如需 NTTS 語音清單 ，請參閱 神經語音 。	2023 年 2 月 7 日
針對 NTTS 新增的語音	Amazon Polly 現在提供兩種新的美國英文 NTTS 語音：Stephen 和 Ruth。如需 NTTS 語音清單 ，請參閱 神經語音 。	2023 年 1 月 31 日
針對 NTTS 新增的語音	Amazon Polly 現在為 提供了新的 NTTS 語音：巴西葡萄牙文 (Thiago)、卡斯提爾西班牙文 (Sergio)、法文 (Rémi)、義大利文 (Adriano) 和墨西哥西班牙文 (Andrés)。如需 NTTS 語音清單 ，請參閱 神經語音 。	2023 年 1 月 24 日

針對 NTTS 新增的語音	Amazon Polly 現在為阿拉伯文 (Hala) 和波蘭文 (Ola) 提供 NTTS 語音。如需 NTTS 語音清單 ，請參閱 神經語音 。	2022 年 11 月 17 日
版本 AWS PrivateLink 支援	Amazon Polly 現在提供支援 AWS PrivateLink。如需進一步了解，請參閱 搭配使用 Amazon Polly 與 VPC 端點 。	2022 年 11 月 9 日
針對 NTTS 新增語音和語言	Amazon Polly 現在為芬蘭文 (Suvi)、挪威文 (Ida) 和瑞典文 (Elin) 提供 NTTS 語音。如需 NTTS 語音清單 ，請參閱 神經語音 。	2022 年 11 月 8 日
針對 NTTS 新增語音	Amazon Polly 現在提供荷蘭 NTTS 語音月亮。如需 NTTS 語音清單 ，請參閱 神經語音 。	2022 年 11 月 2 日
已新增區域	Amazon Polly 現已在歐洲（巴黎）AWS 區域提供。此區域支援神經 TTS (NTTS)。如需詳細資訊，請參閱支援 NTTS 的區域清單的功能 和區域相容性 。	2022 年 9 月 22 日
針對 NTTS 新增語音和語言	Amazon Polly 現在提供廣語 NTTS 語音 Hiujin。如需 NTTS 語音清單 ，請參閱 神經語音 。	2022 年 9 月 20 日
已新增區域	Amazon Polly 現已在亞太區域（孟買）AWS 區域提供。此區域支援神經 TTS (NTTS)。如需詳細資訊，請參閱支援 NTTS 的區域清單的功能 和區域相容性 。	2022 年 9 月 1 日

針對 NTTS 新增語音	Amazon Polly 現在提供國語語音 Zhiyu 做為 NTTS 語音。如需 NTTS 語音清單 ，請參閱 神經 語音。	2022 年 8 月 23 日
針對 NTTS 新增語音	Amazon Polly 現在提供印地語 NTTS 語音 Kajal。如需 NTTS 語音清單 ，請參閱 神經 語音。	2022 年 7 月 27 日
針對 NTTS 新增的語音	Amazon Polly 現在提供美國西班牙文 (Pedro)、德文 (Daniel)、加拿大法文 (Liam) 和英國英文 (Arthur) 的 NTTS 語音。如需 NTTS 語音清單 ，請參閱 神經 語音。	2022 年 6 月 28 日
針對 NTTS 新增語音	Amazon Polly 現在提供葡萄牙文（巴西）語音 Vitória 做為 NTTS 語音。如需 NTTS 語音清單 ，請參閱 神經 語音。	2022 年 4 月 27 日
針對 NTTS 新增語音	Amazon Polly 現在提供葡萄牙文（歐洲）語音 Inês 做為 NTTS 語音。如需 NTTS 語音清單 ，請參閱 神經 語音。	2022 年 4 月 26 日
針對 NTTS 新增語音和語言	Amazon Polly 現在提供德文（奧地利）語言和 NTTS 語音 Hannah。如需 NTTS 語音清單 ，請參閱 神經 語音。	2022 年 4 月 19 日
針對 NTTS 新增的語音和語言	Amazon Polly 現在提供西班牙文（墨西哥）語音 Mia 做為 NTTS 語音。新增了新的語言 Catalan，以及 NTTS 語音 Arlet。如需 NTTS 語音清單 ，請參閱 神經 語音。	2022 年 3 月 22 日

針對 NTTS 新增語音	Amazon Polly 現在提供日文語音 Takumi 做為 NTTS 語音。如需 NTTS 語音清單 ，請參閱 神經 語音。	2021 年 12 月 6 日
針對 NTTS 新增語音	Amazon Polly 現在提供法文語音 Léa 做為 NTTS 語音。如需 NTTS 語音清單 ，請參閱 神經 語音。	2021 年 11 月 18 日
針對 NTTS 新增的語音	Amazon Polly 現在提供義大利文語音 Bianca 和歐洲西班牙文語音 Lucia 做為 NTTS 語音。如需 NTTS 語音清單 ，請參閱 神經 語音。	2021 年 11 月 8 日
針對 NTTS 新增語音	Amazon Polly 現在提供新的南非英文語音 Ayanda。語音只能做為 NTTS 語音使用。如需 NTTS 語音清單 ，請參閱 神經 語音。	2021 年 9 月 1 日
已新增區域	Amazon Polly 現已在非洲（開普敦）AWS 區域提供。此區域支援神經 TTS (NTTS)。如需詳細資訊，請參閱 支援 NTTS 的區域清單的功能和區域相容性 。	2021 年 9 月 1 日
新增了新的語言和語音	Amazon Polly 現在支援紐西蘭英文 (en-NZ)。新的 NTTS 語音 Aria 會說紐西蘭英文和一系列毛利語。	2021 年 8 月 24 日
新功能	Amazon Polly 讓對話說話風格成為神經 Matthew 和 Joanna 語音的預設版本。我們移除了對話式說話樣式的參考。	2021 年 6 月 28 日

針對 NTTS 新增語音	Amazon Polly 現在提供德文語音 Vicki 做為 NTTS 語音。	2021 年 6 月 15 日
新增的語音	新的女性語音 Gabrielle 已新增至法文（加拿大）(fr-CA) 地區設定。語音是高品質的，僅以 NTTS 語音的形式提供。如同所有神經語音，它僅適用於特定區域。如需區域清單，請參閱 功能和區域相容性 。	2021 年 6 月 1 日
針對 NTTS 新增語音	Amazon Polly 現在提供韓文語音 Seoyeon 做為 NTTS 語音。	2021 年 5 月 11 日
針對 NTTS 新增區域	Amazon Polly 現在在加拿大（中部）AWS 區域支援神經 TTS (NTTS)。如需詳細資訊，請參閱 NTTS 的功能和區域相容性 。	2021 年 3 月 17 日
新語音可用於新聞播報員風格	除了新聞播報員說話風格適用的 Matthew、Joanna 和 Lupe 語音之外，Amazon Polly 現在也為此說話風格提供額外的選項。使用神經引擎，您可以將英文 Amy 語音用於新聞播報員風格。如需詳細資訊，請參閱 NTTS 說話風格 。	2020 年 11 月 10 日

針對 NTTS 新增的新區域

除了 NTTS 的現有區域 (us-east-1、us-west-2、eu-west-1 和 ap-southeast-2) 之外，現在支援神經語音的四個額外區域：(ap-northeast-1 (東京)、ap-southeast-1 (新加坡)、eu-central-1 (法蘭克福) 和 eu-west-2 (倫敦))。如需詳細資訊，請參閱 [NTTS 的功能和區域相容性](#)。

2020 年 9 月 3 日

新增的語音

除了兒童語音 Ivy 和 Justin 之外，美國英文 (en-US) 新增了一個新的男性兒童語音 Kevin。這種新語音的品質非常高，只能做為 NTTS 語音使用。與所有神經語音一樣，它僅支援四個區域：us-east-1 (維吉尼亞北部)、us-west-2 (奧勒岡)、eu-west-1 (愛爾蘭) 和 ap-southeast-2 (雪梨)。如需詳細資訊，請參閱 [NTTS 語音](#)。

2020 年 6 月 16 日

新語音可用於新聞播報員風格

除了新聞播報員說話風格適用的 Matthew 和 Joanna 語音之外，Amazon Polly 現在也為此說話風格提供額外的選項。若使用神經引擎，您可以使用西班牙文 (美國) 的 Lupe 語音來表達新聞播報員風格。如需詳細資訊，請參閱 [NTTS 說話風格](#)。

2020 年 4 月 16 日

新功能	除了新聞播報員說話風格之外，Amazon Polly 現在還提供第二個 NTTS 說話風格，協助您合成更好的文字轉語音段落。交談風格會使用神經系統，讓所產生的語音呈現更具親和力與情感的交談對話風格，並且適用於多種使用情況。如需詳細資訊，請參閱 NTTS 說話風格 。	2019 年 11 月 25 日
新增的語音	加入二個新的語音：Camila (女性，巴西葡萄牙文) 和 Lupe (女性，美國西班牙文)。	2019 年 10 月 23 日
已新增新功能	新增 Amazon Polly for Windows 外掛程式 ，將完整的 Amazon Polly 語音納入 Windows SAPI 相容應用程式。	2019 年 9 月 26 日
主要新功能	除了 Amazon Polly 自推出以來支援的標準text-to-speech(TTS) 語音之外，Amazon Polly 現在還提供了改善的神經 TTS (NTTS) 系統，可提供更高品質的語音，從而為您提供最自然且人性化text-to-speech語音。如需詳細資訊，請參閱 神經文字轉語音 。	2019 年 7 月 30 日
新增的語音	新增語音：Lucia (女聲，西班牙文) 和 Bianca (女聲，義大利文)	2018 年 8 月 2 日

已新增新語言	新增語言：墨西哥西班牙文 (es-MX)。此語言使用 Mia 的女聲。	2018 年 8 月 2 日
已新增新語言	加入的新語言：印地語 (hi-IN)。此語音使用 Aditi 的母語，也用於印度英文，使 Aditi Amazon Polly 成為第一個雙語語音。	2018 年 8 月 2 日
已新增新功能	新增 長篇文字訊息的語音合成功能 (最多 100,000 個收費字元)。	2018 年 7 月 17 日
新增了 SSML 功能	新增 合成語音的時間長度上限 。	2018 年 7 月 17 日
新增的語音	加入的新語音：Léa (女性、法文)。	2018 年 6 月 5 日
區域擴展	將 Amazon Polly 服務擴展到所有商業區域。	2018 年 4 月 6 日
已新增新語言	加入的新語言：韓文 (ko-KR)。	2018 年 4 月 6 日
擴充功能	Amazon Polly WordPress 外掛程式功能，包括新增 Amazon Translate 功能。	2018 年 4 月 6 日
新增的語音	加入兩種新的語音：Aditi (女性、印度英文) 和 Seoyeon (女性、韓文)。	2017 年 11 月 15 日
新功能	新增 語音標記 功能，以及擴充的 SSML 功能。	2017 年 4 月 19 日
新的指南	這是 Amazon Polly 開發人員指南的第一個版本。	2016 年 11 月 30 日

本文為英文版的機器翻譯版本，如內容有任何歧義或不一致之處，概以英文版為準。