

# Contents

[Documentación sobre el servicio de voz](#)

[Información general](#)

[¿Qué es Speech Service?](#)

[Compatibilidad de idioma y región](#)

[Voz a texto](#)

[Información general](#)

[¿Qué es la conversión de voz a texto?](#)

[Guías de inicio rápido](#)

[Reconocimiento de voz de un archivo de audio](#)

[Reconocimiento de voz con entrada de micrófono](#)

[Reconocimiento de voz almacenada en Blob Storage](#)

[Guías paso a paso](#)

[Cambio del idioma de origen de reconocimiento de voz](#)

[Elección del modo de reconocimiento de voz](#)

[Mejora de la precisión con Custom Speech](#)

[Mejora de la precisión con listas de frases](#)

[Mejora de la precisión con modelos de inquilino](#)

[Selección de un dispositivo de entrada de micrófono específico](#)

[Uso de detección automática de idioma de origen](#)

[Uso de formatos de entrada de audio comprimidos](#)

[Uso de formatos de entrada de audio comprimidos \(iOS\)](#)

[Uso de formatos de entrada de audio comprimidos \(Android\)](#)

[Uso de reconocimiento de voz con contenedores de Docker](#)

[Referencia](#)

[API de SDK de voz](#)

[C++](#)

[C#](#)

[Java](#)

[JavaScript](#)

[Objective-C](#)

[Python](#)

[Notas de la versión](#)

[API de REST](#)

[Speech-to-text REST API](#)

[API REST de transcripción de Batch](#)

[Recursos](#)

[Preguntas más frecuentes sobre la conversión de voz a texto](#)

[Texto a voz](#)

[Información general](#)

[¿Qué es el texto a voz?](#)

[Guías de inicio rápido](#)

[Síntesis de voz en un archivo de audio](#)

[Síntesis de voz en un altavoz](#)

[Síntesis asincrónica para audio de formato largo](#)

[Guías paso a paso](#)

[Mejora de la salida con SSML](#)

[Mejora de la salida con Voz personalizada](#)

[Síntesis de voz con Long Audio API](#)

[Uso de síntesis de voz con contenedores de Docker](#)

[Referencia](#)

[SDK de voz](#)

[C++](#)

[C#](#)

[Java](#)

[JavaScript](#)

[Objective-C](#)

[Python](#)

[Notas de la versión](#)

[API de REST](#)

[Text-to-speech REST API](#)

[Recursos](#)

## Preguntas más frecuentes sobre la conversión de texto a voz

### Reconocimiento de la intención

#### Guías de inicio rápido

Reconocimiento de voz, intenciones y entidades

#### Guías paso a paso

Reconocimiento de las intenciones de voz con LUIS, C#

### Referencia

#### SDK de voz

C++

C#

Java

JavaScript

Objective-C

Python

Notas de la versión

### Servicio LUIS

Recursos para el desarrollador de LUIS

### Traducción de voz

#### Información general

¿Qué es la traducción de voz?

#### Guías de inicio rápido

Traducción de voz a texto

Traducción de voz a varios idiomas de destino

Traducción de voz a voz

### Tutoriales

Compilación de una aplicación en Flask para traducir y sintetizar texto mediante REST

### Referencia

#### SDK de voz

C++

C#

Java

JavaScript

[Objective-C](#)  
[Python](#)  
[Notas de la versión](#)

**Recursos**

[Preguntas más frecuentes sobre la traducción de voz](#)

[Transcripción de conversaciones](#)

[Información general](#)

[¿Qué es la transcripción de conversaciones?](#)

[Guías paso a paso](#)

[Transcripción de conversaciones en tiempo real](#)

[Transcripción de conversaciones asincrónica](#)

**Referencia**

[SDK de voz](#)

[C++](#)

[C#](#)

[Java](#)

[JavaScript](#)

[Objective-C](#)

[Python](#)

[Notas de la versión](#)

[API de REST](#)

[API REST de transcripción de conversaciones](#)

**Asistentes de voz**

[Información general](#)

[¿Qué es un asistente de voz?](#)

[Creación del asistente](#)

[Comandos personalizados \(versión preliminar\)](#)

[Direct Line Speech](#)

**Guías de inicio rápido**

[C# \(UWP\)](#)

[Java \(Windows, macOS, Linux\)](#)

[Java \(Android\)](#)

[Creación de un comando personalizado \(versión preliminar\)](#)

[Creación de un comando personalizado con parámetros \(versión preliminar\)](#)

[Uso de comandos personalizados con Voz personalizada \(versión preliminar\)](#)

[Uso de comandos personalizados con el SDK de voz \(versión preliminar\)](#)

## Guías paso a paso

[Realización de comandos personalizados con el SDK de voz](#)

[Adición de parámetros de validación a comandos personalizados](#)

[Adición de una confirmación a un comando personalizado](#)

[Adición de una corrección de un paso a un comando personalizado](#)

## Tutoriales

[Habilitar el bot con voz mediante el SDK de voz](#)

## Recursos

[Preguntas frecuentes sobre asistentes de voz](#)

## Personalización

### Voz a texto

[Portal de Custom Speech](#)

[Introducción a Custom Speech](#)

[Preparación de los datos](#)

[Inspección de los datos](#)

[Evaluación de la precisión](#)

[Entrenamiento de un modelo personalizado](#)

[Implementación de un modelo personalizado](#)

[Guía de creación de transcripciones con la etiqueta humana](#)

### Modelos de inquilino

[Creación, publicación y uso de un modelo](#)

## Texto a voz

[Implementación responsable](#)

[Acceso validado de Voz personalizada](#)

[Implementación de tecnología sintética](#)

[Divulgación de talento de voz](#)

[Directrices de diseño de divulgación de información](#)

[Modelos de diseño de divulgación](#)

- [Código de conducta](#)
- [Portal de Voz personalizada](#)
  - [Introducción a voz personalizada](#)
  - [Preparación de los datos](#)
  - [Creación y uso de modelos de voz personalizados](#)
  - [Grabación de ejemplos de voz](#)
- [Audio Content Creation](#)
  - [Procedimiento: Audio Content Creation](#)
  - [Directrices de nomenclatura de palabras clave personalizadas](#)
  - [Detección de palabras clave personalizadas](#)
- [Speech Service](#)
  - [Escenarios](#)
    - [Centros de atención al cliente](#)
    - [Guías paso a paso](#)
      - [Prueba gratuita de Speech Service](#)
      - [Creación de recurso del servicio de voz](#)
  - [Contenedores](#)
    - [Instalación y ejecución de contenedores](#)
    - [Configuración de contenedores](#)
    - [Kubernetes y Helm](#)
    - [Azure Container Instances](#)
  - [Ejemplos](#)
    - [Ejemplos de REST de texto a voz](#)
    - [Ejemplos de REST de transcripción de Batch](#)
  - [Referencia](#)
    - [API de REST](#)
      - [API REST de transcripción y personalización de Batch](#)
      - [API REST de transcripción de conversaciones](#)
      - [Speech-to-text REST API](#)
      - [Text-to-speech REST API](#)
    - [Documentación de Swagger](#)
  - [SDK de voz](#)
    - [Información general](#)

[Acerca del SDK de Voz](#)

[Disponibilidad de escenario](#)

[Guías de inicio rápido](#)

[Crear un proyecto](#)

[Configuración del SDK de Voz](#)

[Guías paso a paso](#)

[Habilitación del registro](#)

[Seguimiento del uso de la memoria](#)

[Ejemplos](#)

[Ejemplos del SDK de voz \(GitHub\)](#)

[Ejemplos del SDK para dispositivos de voz \(GitHub\)](#)

[Referencia](#)

[C++](#)

[C#](#)

[Java](#)

[JavaScript](#)

[Objective-C](#)

[Python](#)

[Notas de la versión del SDK de voz](#)

[Dispositivos](#)

[Información general](#)

[Introducción a Speech Devices SDK](#)

[Obtener el SDK de dispositivos de voz](#)

[Hardware](#)

[Recomendaciones de la matriz de micrófonos](#)

[Kit de Roobo v1](#)

[Guías de inicio rápido](#)

[Windows](#)

[Linux](#)

[Android](#)

[Recursos](#)

[Solución de problemas de Speech Devices SDK](#)

## Referencia

[Notas de la versión de Speech Devices SDK](#)

## Migración

[Desde Bing Speech](#)

[Desde Custom Speech Service](#)

[Desde Translator Speech API](#)

## Recursos

[Soporte técnico](#)

[Regions](#)

[Precios y límites](#)

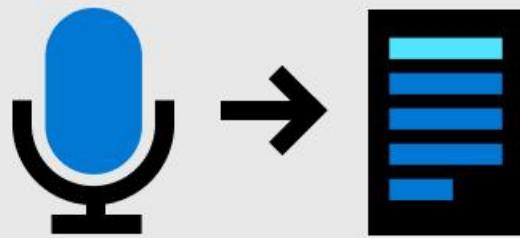
[Cumplimiento normativo](#)

[Azure Roadmap](#)

[Privacidad y cookies](#)

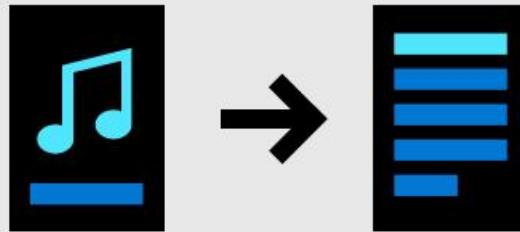
# Documentación sobre el servicio Voz

- ▶ [Introducción](#)
- ▶ [Voz a texto](#)



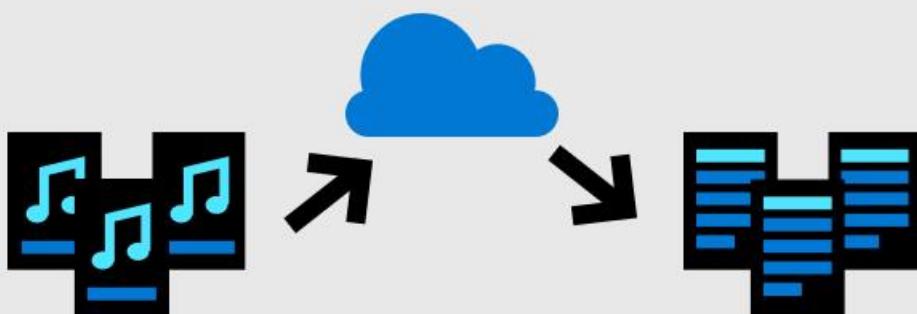
## Reconocimiento de voz con un micrófono

Use el SDK de voz para reconocer la voz de un micrófono y transcribir la salida.



## Reconocimiento de voz de un archivo de audio

Use el SDK de voz para reconocer la voz de un único archivo y transcribir la salida.



## Reconocimiento asincrónico desde blob

Use nuestro servicio REST para reconocer de forma asincrónica la voz de archivos almacenados en Azure Blob Storage.



## Compatibilidad con idiomas

Obtenga más información sobre la programación y la compatibilidad con idiomas hablados para voz a texto.



## Precios

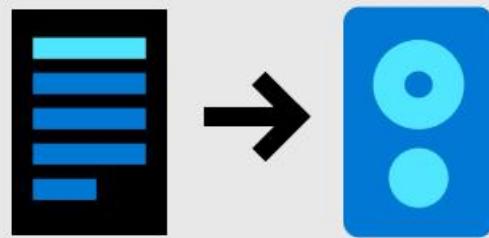
Obtenga más información sobre los costos asociados con la conversión de voz a texto.



## Lectura de los documentos

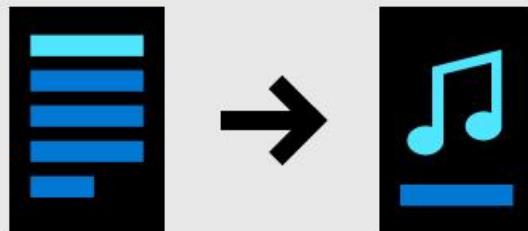
Aprenda a agregar reconocimiento de voz a sus aplicaciones, herramientas y productos. Incluye conceptos, tutoriales, referencia de API y notas de la versión.

- › [Texto a voz](#)



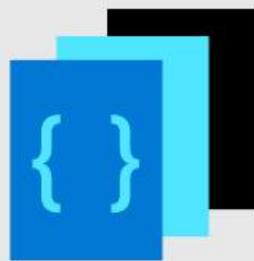
### Síntesis de voz en un altavoz

Use el SDK de voz para sintetizar voz en una salida de audio, como un altavoz.



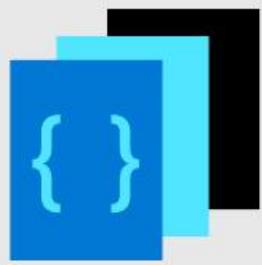
### Síntesis de voz en un archivo de audio

Use el SDK de voz para sintetizar voz en un archivo de audio.



### Lenguaje de marcado de síntesis de voz

Use el lenguaje de marcado de síntesis de voz para ajustar el tono, la prosodia y la velocidad de habla de la salida de audio.



## Compatibilidad con idiomas

Obtenga información acerca de qué idiomas se admiten para síntesis de voz.



## Precios

Obtenga más información sobre los costos asociados con la conversión de texto a voz.



## Lectura de los documentos

Aprenda a agregar síntesis de voz a sus aplicaciones, herramientas y productos. Incluye conceptos, tutoriales, referencia de API y notas de la versión.

- › Reconocimiento de la intención



### Reconocimiento de voz, intenciones y entidades

Use el SDK de Voz y Language Understanding para reconocer la voz, las intenciones y las entidades.



### Documentación de Language Understanding (LUIS)

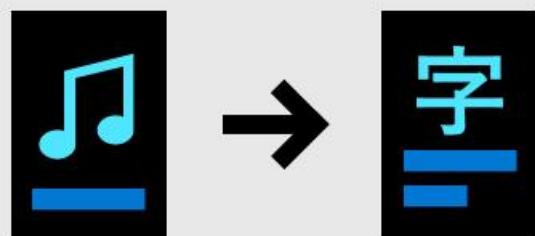
Obtenga más información sobre Language Understanding Service (LUIS) y el procesamiento de lenguaje natural (NLP).



### Portal de Language Understanding (LUIS)

Incorpore lenguaje natural a sus aplicaciones, bots y dispositivos IoT.

- › Traducción de voz



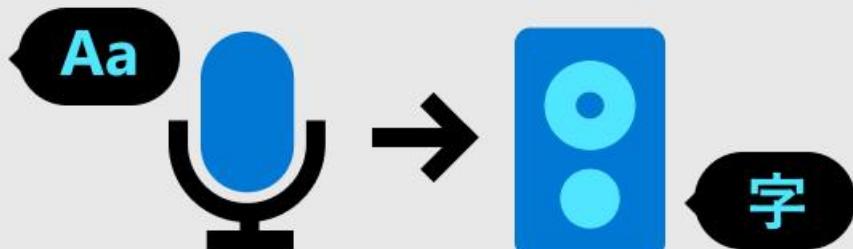
#### Traducción de voz a texto desde el micrófono

Use el SDK de voz para traducir voz a texto desde un micrófono.



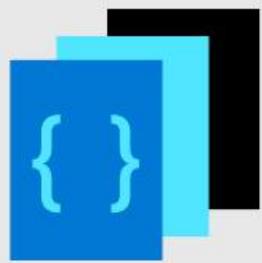
#### Traducción de voz a varios idiomas de destino

Use el SDK de voz para traducir la voz a varias salidas de idiomas de destino.



#### Traducción de voz a voz

Use el SDK de voz para traducir voz a voz.



## Compatibilidad con idiomas

Obtenga más información sobre la programación y la compatibilidad con idiomas hablados para traducción de voz.



## Precios

Obtenga más información sobre los costos asociados con la traducción de voz.



## Lectura de los documentos

Aprenda a agregar traducción de voz a sus aplicaciones, herramientas y productos. Incluye conceptos, tutoriales, referencia de API y notas de la versión.

- Transcripción de conversaciones



## Información general

Obtenga más información sobre la transcripción de conversaciones y cómo integrarla en sus productos.



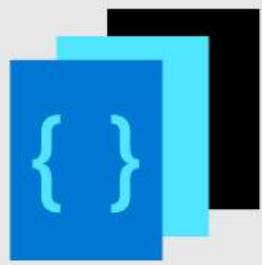
## Transcripción de una conversación en tiempo real

Obtenga información sobre cómo transcribir una conversación en tiempo real.



## Transcripción asincrónica de conversaciones

Obtenga información sobre cómo transcribir las conversaciones de forma asincrónica, sondear el estado y descargar las salidas.



## Compatibilidad con idiomas

Obtenga más información sobre la programación y la compatibilidad con idiomas hablados para transcripción de conversaciones.



## Precios

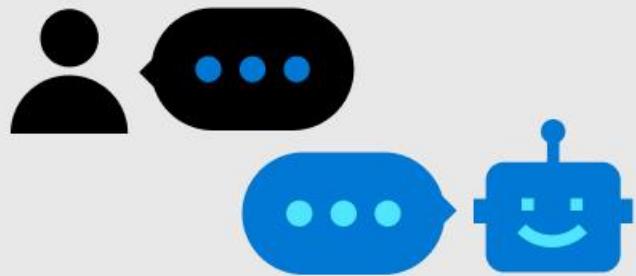
Obtenga más información acerca de los costos asociados con la transcripción de conversaciones.

### Asistentes de voz



## Información general

Obtenga más información sobre todo lo necesario para crear un asistente de voz.



### Integración con Bot Framework

Obtenga más información sobre Bot Framework, el canal Direct Line Speech y mucho más.



### Uso de comandos personalizados

Aprenda a compilar fácilmente sólidas aplicaciones de voz de control y comandos, con el fin de que los usuarios puedan completar tareas mediante su voz.

- › Soporte técnico



### Problemas de GitHub

Examine problemas ya abiertos o cree otros nuevos para el SDK de Voz en GitHub.



## Stack Overflow

Formule preguntas y obtenga ayuda de la comunidad del servicio Voz en Stack Overflow.



## Foro de UserVoice

Comparta sus ideas, sugiera mejoras o solicite nuevas características para el servicio Voz.

- ▶ Personalización
- ▶ Voz a texto



## Mejora de la precisión con listas de frases

Use listas de frases en el SDK de voz para mejorar la precisión del reconocimiento.



### Mejora de la precisión con modelos de inquilino

Genere modelos personalizados con datos de Office 365 para optimizar la precisión del reconocimiento de voz en cuanto a términos específicos de la organización.



### Mejora de la precisión con Custom Speech

Conjunto de herramientas en línea que permiten evaluar y mejorar la precisión de la conversión de voz a texto de Microsoft para las aplicaciones, herramientas y productos.



### Compatibilidad con idiomas

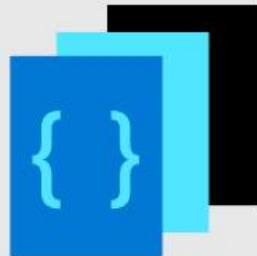
Obtenga más información sobre la programación y la compatibilidad con idiomas hablados para Custom Speech.



## Precios

Obtenga más información sobre los costos asociados con Custom Speech.

- » [Texto a voz](#)



## Mejora de síntesis con SSML

Use el lenguaje de marcado de síntesis de voz para ajustar el tono, la prosodia y la velocidad de habla de la salida de audio.



## Mejora de síntesis con Voz personalizada

Cree una voz reconocible única para las aplicaciones de texto a voz con los datos de habla disponibles.



### Mejora de síntesis con Audio Content Creation

Use la herramienta Audio Content Creation para ajustar las salidas de voz sintetizada.



### Acceso a voces neuronales personalizadas

Esta es una característica validada que permite crear una voz neuronal personalizada.

Más información sobre el acceso y las restricciones.



### Precios

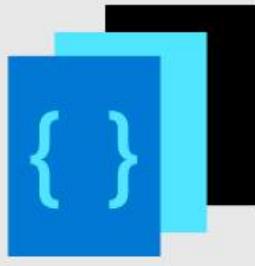
Obtenga más información sobre los costos asociados con Voz personalizada.

- ▶ Escenarios
- ▶ Casos de uso



### Transcripción para los centros de llamadas

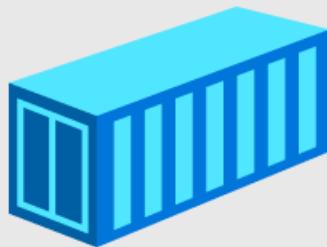
Un escenario común para la conversión de voz a texto es la transcripción de grandes volúmenes de datos de telefonía que pueden proceder de varios orígenes como, por ejemplo, un sistema de respuesta de voz interactiva (IVR).



### Disponibilidad del escenario y características

Conozca la disponibilidad del escenario y las características por plataforma y lenguaje de programación.

#### › Implementación



### Implementación del servicio Voz en contenedores

Uso de Docker para implementar el servicio Voz en una instancia de contenedor.



### Regiones admitidas

Averigüe dónde se admite el servicio Voz.

# ¿Qué es Speech Service?

14/01/2020 • 10 minutes to read • [Edit Online](#)

El servicio de voz es la unificación de las funcionalidades de conversión de voz a texto, conversión de texto a voz y traducción de voz en una sola suscripción de Azure. Es fácil habilitar voz en sus aplicaciones, herramientas y dispositivos con el [SDK de voz](#), el [SDK de dispositivos de voz](#) o las [API de REST](#).

## IMPORTANT

El Servicio de voz ha reemplazado a Bing Speech API, Translator Speech y Custom Speech. Consulte *Guías de procedimientos > Migración* para obtener instrucciones de migración.

Estas características conforman el Servicio de voz. Use los vínculos en esta tabla para obtener más información sobre los casos de uso comunes para cada característica o examinar la referencia de API.

SERVICIO	CARACTERÍSTICA	DESCRIPCIÓN	SDK	REST
<a href="#">Voz a texto</a>	Voz a texto	Voz a texto transcribe secuencias de audio a texto en tiempo real que sus aplicaciones, herramientas o dispositivos pueden usar o mostrar. Use voz a texto con <a href="#">Language Understanding (LUIS)</a> para derivar las intenciones del usuario a partir de voz transcrita y actuar en los comandos de voz.	Sí	Sí
	<a href="#">Batch Transcription</a> (Transcripción de Azure Batch)	La transcripción de Azure Batch permite la transcripción de voz a texto asincrónica de grandes volúmenes de datos. Este es un servicio basado en REST, que usa el mismo punto de conexión que la personalización y la administración de modelos.	No	Sí

SERVICIO	CARACTERÍSTICA	DESCRIPCIÓN	SDK	REST
	Transcripción de conversaciones	Permite el reconocimiento de voz en tiempo real, la identificación del hablante y la diarización. Es perfecto para transcribir reuniones en persona con la capacidad de distinguir a los oradores.	Sí	No
	Creación de modelos de voz personalizados	Si usa voz a texto para el reconocimiento y la transcripción en un entorno único, puede crear y entrenar modelos acústicos, de lenguaje y pronunciación personalizados para dirigir el sonido ambiental o vocabulario específico del sector.	No	Sí
Texto a voz	Texto a voz	Texto a voz convierte el texto de entrada en voz sintetizada similar a la humana mediante el <a href="#">Lenguaje de marcado de síntesis de voz (SSML)</a> . Elija entre voces estándar y voces neuronales (consulte <a href="#">Compatibilidad de idioma</a> ).	Sí	Sí
	Creación de voces personalizadas	Cree fuentes de voz personalizadas únicas para su marca o producto.	No	Sí
Traducción de voz	Traducción de voz	La traducción de voz habilita la traducción de voz en varios idiomas en tiempo real en sus aplicaciones, herramientas y dispositivos. Use este servicio para la traducción de voz a voz y voz a texto.	Sí	No

SERVICIO	CARACTERÍSTICA	DESCRIPCIÓN	SDK	REST
Asistentes de voz	Asistentes de voz	Los asistentes de voz que utilizan el Servicio de voz permiten a los desarrolladores crear interfaces de conversación naturales, similares a la humana, para sus aplicaciones y experiencias. El servicio del asistente de voz proporciona una interacción rápida y confiable entre un dispositivo y una implementación de asistente que usa el canal de voz de Direct Line Speech de Bot Framework o el servicio integrado de comandos personalizados (versión preliminar) para la finalización de tareas.	Sí	No

## Noticias y actualizaciones

Obtenga información sobre las novedades del Servicio de voz.

- Noviembre de 2019
  - Se han agregado dos nuevos estilos de habla, `newscast` y `customerservice` para la voz `en-US-JessaNeural`.
- Septiembre de 2019
  - SDK de Voz versión 1.7.0 publicado. Para obtener una lista completa de actualizaciones, mejoras y problemas conocidos, consulte las [Notas de la versión](#).
- Agosto de 2019
  - **Nuevo tutorial:** [Habilitar el bot con Voz mediante Speech SDK, C#](#)
  - Se ha agregado un nuevo estilo de habla, `chat`, para la voz `en-US-JessaNeural`.
- Junio de 2019
  - El SDK de Voz versión 1.6.0 publicado. Para obtener una lista completa de actualizaciones, mejoras y problemas conocidos, consulte las [Notas de la versión](#).
- Mayo de 2019: ya hay documentación disponible para [Transcripción de conversaciones](#), [Transcripción de centros de llamadas](#) y [Asistentes de voz](#).
- Mayo de 2019
  - Speech SDK versión 1.5.1 publicado. Para obtener una lista completa de actualizaciones, mejoras y problemas conocidos, consulte las [Notas de la versión](#).
  - Speech SDK versión 1.5.0 publicado. Para obtener una lista completa de actualizaciones, mejoras y problemas conocidos, consulte las [Notas de la versión](#).

## Prueba del Servicio de voz

Ofrecemos guías de inicio rápido en los lenguajes de programación más populares, cuyo diseño individual le permite ejecutar código en menos de 10 minutos. En esta tabla se incluyen las guías de inicio rápido más populares para cada característica. Use el menú de navegación izquierdo para explorar lenguajes y plataformas adicionales.

VOZ A TEXTO (SDK)	TEXTO A VOZ (SDK)	TRADUCCIÓN (SDK)
Reconocimiento de voz de un archivo de audio	Síntesis de voz en un archivo de audio	Traducción de voz a texto
Reconocimiento de voz con un micrófono	Síntesis de voz en un altavoz	Traducción de voz a varios idiomas de destino
Reconocimiento de voz almacenada en Blob Storage	Síntesis asíncrona para audio de formato largo	Traducción de voz a voz

#### NOTE

Voz a texto y texto a voz también tienen asociados puntos de conexión REST e inicios rápidos.

Una vez que haya tenido la oportunidad de usar el servicio de voz, pruebe nuestro tutorial, que le enseña a reconocer intenciones a partir de contenido de voz mediante el SDK de voz y LUIS.

- [Tutorial: Reconocimiento de intenciones a partir de contenido de voz con el SDK de voz y LUIS, C#](#)
- [Tutorial: Habilitación del bot con voz mediante el SDK de voz, C#](#)
- [Tutorial: Compilación de una aplicación de Flask para traducir texto, analizar opiniones y sintetizar la voz de texto a voz, REST](#)

## Obtención de código de ejemplo

Hay código de ejemplo para el Servicio de voz disponible en GitHub. En estos ejemplos se tratan escenarios comunes como la lectura de audio de un archivo o flujo, el reconocimiento continuo y de una sola emisión, y el trabajo con modelos personalizados. Use estos vínculos para ver ejemplos de SDK y REST:

- [Ejemplos de conversión de voz a texto, texto a voz y traducción de voz \(SDK\)](#)
- [Ejemplos de transcripción de Azure Batch \(REST\)](#)
- [Ejemplos de texto a voz \(REST\)](#)
- [Ejemplos del asistente de voz \(SDK\)](#)

## Personalización de su experiencia de voz

El Servicio de voz funciona bien con los modelos integrados; sin embargo, es posible que desee personalizar y optimizar más la experiencia para su producto o entorno. Las opciones de personalización abarcan desde la optimización de modelos acústicos a fuentes de voz únicas para su marca.

SPEECH SERVICE	PLATAFORMA	DESCRIPCIÓN
Voz a texto	Custom Speech	El reconocimiento de voz personalizado se adapta a sus necesidades y datos disponibles. Elimine las barreras del reconocimiento de voz, como el estilo de habla, el vocabulario y el ruido de fondo.

SPEECH SERVICE	PLATAFORMA	DESCRIPCIÓN
Text-to-Speech	Voz personalizada	Cree una voz reconocible única para las aplicaciones de texto a voz con los datos de habla disponibles. Puede optimizar aún más las salidas de voz ajustando un conjunto de parámetros de voz.

## Documentos de referencia

- [Acerca del SDK de Voz](#)
- [Speech Devices SDK](#)
- [API REST: Speech-to-text](#) (API de REST: Voz a texto)
- [API REST: Text-to-speech](#) (API de REST: Texto a voz)
- [API REST: Batch transcription and customization](#) (API de REST: Transcripción y personalización de Azure Batch)

## Pasos siguientes

[Obtenga una clave de suscripción gratuita a los servicios de Voz](#)

# Compatibilidad con idiomas y regiones para el servicio de voz

14/01/2020 • 19 minutes to read • [Edit Online](#)

La compatibilidad con los idiomas varía según la funcionalidad del servicio de voz. En las tablas siguientes se resume la compatibilidad con idiomas para la [conversión de voz a texto](#), [conversión de texto a voz](#) y las ofertas del servicio [Speech Translation](#).

## Voz a texto

El SDK de Voz de Microsoft y la API REST admiten los siguientes idiomas (configuraciones regionales). Para mejorar la precisión, se ofrece la personalización para un subconjunto de idiomas mediante la carga de audio y transcripciones con etiqueta humana o texto relacionado: Oraciones. La personalización de la pronunciación solo está disponible actualmente para [en-US](#) y [de-DE](#). Aprenda más sobre la personalización [aquí](#).

CONFIGURACIÓN REGIONAL	IDIOMA	COMPATIBLE	PERSONALIZABLE
<a href="#">ar-EG</a>	Árabe (Egipto), estándar moderno	Sí	Sí
<a href="#">ar-SA</a>	Árabe (Arabia Saudí)	Sí	Sí
<a href="#">ar-AE</a>	Árabe (Emiratos Árabes Unidos)	Sí	Sí
<a href="#">ar-KW</a>	Árabe (Kuwait)	Sí	Sí
<a href="#">ar-QA</a>	Árabe (Qatar)	Sí	Sí
<a href="#">ca-ES</a>	Catalán	Sí	No
<a href="#">da-DK</a>	Danés (Dinamarca)	Sí	No
<a href="#">de-DE</a>	Alemán (Alemania)	Sí	Sí
<a href="#">en-AU</a>	Inglés (Australia)	Sí	Sí
<a href="#">en-CA</a>	Inglés (Canadá)	Sí	Sí
<a href="#">en-GB</a>	Inglés (Reino Unido)	Sí	Sí
<a href="#">en-IN</a>	Inglés (India)	Sí	Sí
<a href="#">en-NZ</a>	Inglés (Nueva Zelanda)	Sí	Sí
<a href="#">en-US</a>	Spanish (Traditional Sort) - Spain	Sí	Sí

CONFIGURACIÓN REGIONAL	IDIOMA	COMPATIBLE	PERSONALIZABLE
es-ES	Español (España)	Sí	Sí
es-MX	Español (México)	Sí	Sí
fi-FI	Finés (Finlandia)	Sí	No
fr-CA	Francés (Canadá)	Sí	Sí
fr-FR	Francés (Francia)	Sí	Sí
gu-IN	Gujarati (India)	Sí	Sí
hi-IN	Hindi (India)	Sí	Sí
it-IT	Italiano (Italia)	Sí	Sí
ja-JP	Japonés (Japón)	Sí	Sí
ko-KR	Coreano (Corea)	Sí	Sí
mr-IN	Maratí (India)	Sí	Sí
nb-NO	Noruego, Bokmål (Noruego)	Sí	No
nl-NL	Neerlandés (Países Bajos)	Sí	Sí
pl-PL	Polaco (Polonia)	Sí	No
pt-BR	Portugués (Brasil)	Sí	Sí
pt-PT	Portugués (Portugal)	Sí	Sí
ru-RU	Ruso (Rusia)	Sí	Sí
sv-SE	Sueco (Suecia)	Sí	No
ta-IN	Tamil (India)	Sí	Sí
te-IN	Telugu (India)	Sí	Sí
zh-CN	Chino (mandarín, simplificado)	Sí	Sí
zh-HK	Chino (cantonés, tradicional)	Sí	Sí
zh-TW	Chino (mandarín, Taiwán)	Sí	Sí
th-TH	Tailandés (Tailandia)	Sí	No

CONFIGURACIÓN REGIONAL	IDIOMA	COMPATIBLE	PERSONALIZABLE
tr-TR	Turquía	Sí	Sí

## Texto a voz

Tanto el SDK de Voz de Microsoft como las API REST admiten estas voces, y cada una de ellas admite un idioma y un dialecto específicos, que se identifican mediante la configuración regional.

### IMPORTANT

Los precios son distintos para voces estándar, personalizadas y neuronales. Consulte la página de [precios](#) para más información.

### Voces neuronales

Texto a voz neuronal es un nuevo tipo de síntesis de voz con tecnología de redes neuronales profundas. Cuando se usa una voz neuronal, es prácticamente imposible distinguir la voz sintetizada de las grabaciones humanas.

Las voces neuronales se pueden usar para que las interacciones con los bots de chat y los asistentes de voz sean más naturales y atractivas, para convertir textos digitales (por ejemplo, los libros electrónicos) en audiolibros y para mejorar los sistemas de navegación de los automóviles. Con su prosodia natural similar a la humana y la clara articulación de las palabras, las voces neuronales reducen considerablemente la fatiga de la escucha que aparece cuando los usuarios interactúan con sistemas de inteligencia artificial.

Para más información acerca de la disponibilidad regional, consulte las [regiones](#).

CONFIGURACIÓN REGIONAL	IDIOMA	SEXO	ASIGNACIÓN DE NOMBRE DE SERVICIO COMPLETO	NOMBRE CORTO DE VOZ
de-DE	Alemán (Alemania)	Female	"Microsoft Server Speech Text to Speech Voice (de-DE, KatjaNeural)"	"de-DE-KatjaNeural"
en-US	Inglés (EE. UU.)	Male	"Microsoft Server Speech Text to Speech Voice (en-US, GuyNeural)"	"en-US-GuyNeural"
en-US	Inglés (EE. UU.)	Female	"Microsoft Server Speech Text to Speech Voice (en-US, JessaNeural)"	"en-US-JessaNeural"
it-IT	Italiano (Italia)	Female	"Microsoft Server Speech Text to Speech Voice (it-IT, ElsaNeural)"	"it-IT-ElsaNeural"
zh-CN	Chino (continental)	Female	"Microsoft Server Speech Text to Speech Voice (zh-CN, XiaoxiaoNeural)"	"zh-CN-XiaoxiaoNeural"

Para obtener información sobre cómo configurar y ajustar las voces neuronales, consulte [Lenguaje de marcado](#)

de síntesis de voz.

**NOTE**

Puede usar la asignación de nombre de servicio completo o el nombre corto de voz en las solicitudes de síntesis de voz.

## Voces estándar

Hay más de 75 voces estándar disponibles en más de 45 idiomas y configuraciones regionales, lo que le permite convertir texto en voz sintetizada. Para más información acerca de la disponibilidad regional, consulte las [regiones](#).

CONFIGURACIÓN REGIONAL	IDIOMA	SEXO	ASIGNACIÓN DE NOMBRE DE SERVICIO COMPLETO	NOMBRE CORTO
† ar-EG	Árabe (Egipto)	Female	"Microsoft Server Speech Text to Speech Voice (ar-EG, Hoda)"	"ar-EG-Hoda"
ar-SA	Árabe (Arabia Saudí)	Male	"Microsoft Server Speech Text to Speech Voice (ar-SA, Naayf)"	"ar-SA-Naayf"
bg-BG	Búlgaro	Male	"Microsoft Server Speech Text to Speech Voice (bg-BG, Ivan)"	"bg-BG-Ivan"
ca-ES	Catalán (España)	Female	"Microsoft Server Speech Text to Speech Voice (ca-ES, HerenaRUS)"	"ca-ES-HerenaRUS"
cs-CZ	Checo	Male	"Microsoft Server Speech Text to Speech Voice (cs-CZ, Jakub)"	"cs-CZ-Jakub"
da-DK	Danés	Female	"Microsoft Server Speech Text to Speech Voice (da-DK, HelleRUS)"	"da-DK-HelleRUS"
de-AT	Alemán (Austria)	Male	"Microsoft Server Speech Text to Speech Voice (de-AT, Michael)"	"de-AT-Michael"
de-CH	Alemán (Suiza)	Male	"Microsoft Server Speech Text to Speech Voice (de-CH, Karsten)"	"de-CH-Karsten"

<b>CONFIGURACIÓN REGIONAL</b>	<b>IDIOMA</b>	<b>SEXO</b>	<b>ASIGNACIÓN DE NOMBRE DE SERVICIO COMPLETO</b>	<b>NOMBRE CORTO</b>
de-DE	Alemán (Alemania)	Female	"Microsoft Server Speech Text to Speech Voice (de-DE, Hedda)"	"de-DE-Hedda"
		Female	"Microsoft Server Speech Text to Speech Voice (de-DE, HeddaRUS)"	"de-DE-HeddaRUS"
		Male	"Microsoft Server Speech Text to Speech Voice (de-DE, Stefan, Apollo)"	"de-DE-Stefan-Apollo"
el-GR	Griego	Male	"Microsoft Server Speech Text to Speech Voice (el-GR, Stefanos)"	"el-GR-Stefanos"
en-AU	Inglés (Australia)	Female	"Microsoft Server Speech Text to Speech Voice (en-AU, Catherine)"	"en-AU-Catherine"
		Female	"Microsoft Server Speech Text to Speech Voice (en-AU, HayleyRUS)"	"en-AU-HayleyRUS"
en-CA	Inglés (Canadá)	Female	"Microsoft Server Speech Text to Speech Voice (en-CA, Linda)"	"en-CA-Linda"
		Female	"Microsoft Server Speech Text to Speech Voice (en-CA, HeatherRUS)"	"en-CA-HeatherRUS"
en-GB	English (Reino Unido)	Female	"Microsoft Server Speech Text to Speech Voice (en-GB, Susan, Apollo)"	"en-GB-Susan-Apollo"
		Female	"Microsoft Server Speech Text to Speech Voice (en-GB, HazelRUS)"	"en-GB-HazelRUS"
		Male	"Microsoft Server Speech Text to Speech Voice (en-GB, George, Apollo)"	"en-GB-George-Apollo"

<b>CONFIGURACIÓN REGIONAL</b>	<b>IDIOMA</b>	<b>SEXO</b>	<b>ASIGNACIÓN DE NOMBRE DE SERVICIO COMPLETO</b>	<b>NOMBRE CORTO</b>
en-IE	Inglés (Irlanda)	Male	"Microsoft Server Speech Text to Speech Voice (en-IE, Sean)"	"en-IE-Sean"
en-IN	Inglés (India)	Female	"Microsoft Server Speech Text to Speech Voice (en-IN, Heera, Apollo)"	"en-IN-Heera-Apollo"
		Female	"Microsoft Server Speech Text to Speech Voice (en-IN, PriyaRUS)"	"en-IN-PriyaRUS"
		Male	"Microsoft Server Speech Text to Speech Voice (en-IN, Ravi, Apollo)"	"en-IN-Ravi-Apollo"
en-US	Inglés (EE. UU.)	Female	"Microsoft Server Speech Text to Speech Voice (en-US, ZiraRUS)"	"en-US-ZiraRUS"
		Female	"Microsoft Server Speech Text to Speech Voice (en-US, JessaRUS)"	"en-US-JessaRUS"
		Male	"Microsoft Server Speech Text to Speech Voice (en-US, BenjaminRUS)"	"en-US-BenjaminRUS"
		Female	"Microsoft Server Speech Text to Speech Voice (en-US, Jessa24kRUS)"	"en-US-Jessa24kRUS"
		Male	"Microsoft Server Speech Text to Speech Voice (en-US, Guy24kRUS)"	"en-US-Guy24kRUS"
es-ES	Español (España)	Female	"Microsoft Server Speech Text to Speech Voice (es-ES, Laura, Apollo)"	"es-ES-Laura-Apollo"
		Female	"Microsoft Server Speech Text to Speech Voice (es-ES, HelenaRUS)"	"es-ES-HelenaRUS"

<b>CONFIGURACIÓN REGIONAL</b>	<b>IDIOMA</b>	<b>SEXO</b>	<b>ASIGNACIÓN DE NOMBRE DE SERVICIO COMPLETO</b>	<b>NOMBRE CORTO</b>
		Male	"Microsoft Server Speech Text to Speech Voice (es-ES, Pablo, Apollo)"	"es-ES-Pablo-Apollo"
<b>es-MX</b>	Español (México)	Female	"Microsoft Server Speech Text to Speech Voice (es-MX, HildaRUS)"	"es-MX-HildaRUS"
		Male	"Microsoft Server Speech Text to Speech Voice (es-MX, Raul, Apollo)"	"es-MX-Raul-Apollo"
<b>fi-FI</b>	Finés	Female	"Microsoft Server Speech Text to Speech Voice (fi-FI, HeidiRUS)"	"fi-FI-HeidiRUS"
<b>fr-CA</b>	Francés (Canadá)	Female	"Microsoft Server Speech Text to Speech Voice (fr-CA, Caroline)"	"fr-CA-Caroline"
		Female	"Microsoft Server Speech Text to Speech Voice (fr-CA, HarmonieRUS)"	"fr-CA-HarmonieRUS"
<b>fr-CH</b>	Francés (Suiza)	Male	"Microsoft Server Speech Text to Speech Voice (fr-CH, Guillaume)"	"fr-CH-Guillaume"
<b>fr-FR</b>	Francés (Francia)	Female	"Microsoft Server Speech Text to Speech Voice (fr-FR, Julie, Apollo)"	"fr-FR-Julie-Apollo"
		Female	"Microsoft Server Speech Text to Speech Voice (fr-FR, HortenseRUS)"	"fr-FR-HortenseRUS"
		Male	"Microsoft Server Speech Text to Speech Voice (fr-FR, Paul, Apollo)"	"fr-FR-Paul-Apollo"
<b>he-IL</b>	Hebreo (Israel)	Male	"Microsoft Server Speech Text to Speech Voice (he-IL, Asaf)"	"he-IL-Asaf"

CONFIGURACIÓN REGIONAL	IDIOMA	SEXO	ASIGNACIÓN DE NOMBRE DE SERVICIO COMPLETO	NOMBRE CORTO
hi-IN	Hindi (India)	Female	"Microsoft Server Speech Text to Speech Voice (hi-IN, Kalpana, Apollo)"	"hi-IN-Kalpana-Apollo"
		Female	"Microsoft Server Speech Text to Speech Voice (hi-IN, Kalpana)"	"hi-IN-Kalpana"
		Male	"Microsoft Server Speech Text to Speech Voice (hi-IN, Hemant)"	"hi-IN-Hemant"
hr-HR	Croata	Male	"Microsoft Server Speech Text to Speech Voice (hr-HR, Matej)"	"hr-HR-Matej"
hu-HU	Húngaro	Male	"Microsoft Server Speech Text to Speech Voice (hu-HU, Szabolcs)"	"hu-HU-Szabolcs"
id-ID	Indonesio	Male	"Microsoft Server Speech Text to Speech Voice (id-ID, Andika)"	"id-ID-Andika"
it-IT	Italiano	Male	"Microsoft Server Speech Text to Speech Voice (it-IT, Cosimo, Apollo)"	"it-IT-Cosimo-Apollo"
		Female	"Microsoft Server Speech Text to Speech Voice (it-IT, LuciaRUS)"	"it-IT-LuciaRUS"
ja-JP	Japonés	Female	"Microsoft Server Speech Text to Speech Voice (ja-JP, Ayumi, Apollo)"	"ja-JP-Ayumi-Apollo"
		Male	"Microsoft Server Speech Text to Speech Voice (ja-JP, Ichiro, Apollo)"	"ja-JP-Ichiro-Apollo"
		Female	"Microsoft Server Speech Text to Speech Voice (ja-JP, HarukaRUS)"	"ja-JP-HarukaRUS"

<b>CONFIGURACIÓN REGIONAL</b>	<b>IDIOMA</b>	<b>SEXO</b>	<b>ASIGNACIÓN DE NOMBRE DE SERVICIO COMPLETO</b>	<b>NOMBRE CORTO</b>
ko-KR	Coreano	Female	"Microsoft Server Speech Text to Speech Voice (ko-KR, HeamiRUS)"	"ko-KR-HeamiRUS"
ms-MY	Malayo	Male	"Microsoft Server Speech Text to Speech Voice (ms-MY, Rizwan)"	"ms-MY-Rizwan"
nb-NO	Noruego	Female	"Microsoft Server Speech Text to Speech Voice (nb-NO, HuldaRUS)"	"nb-NO-HuldaRUS"
nl-NL	Neerlandés	Female	"Microsoft Server Speech Text to Speech Voice (nl-NL, HannaRUS)"	"nl-NL-HannaRUS"
pl-PL	Polaco	Female	"Microsoft Server Speech Text to Speech Voice (pl-PL, PaulinaRUS)"	"pl-PL-PaulinaRUS"
pt-BR	Portugués (Brasil)	Female	"Microsoft Server Speech Text to Speech Voice (pt-BR, HeloisaRUS)"	"pt-BR-HeloisaRUS"
		Male	"Microsoft Server Speech Text to Speech Voice (pt-BR, Daniel, Apollo)"	"pt-BR-Daniel-Apollo"
pt-PT	Portugués (Portugal)	Female	"Microsoft Server Speech Text to Speech Voice (pt-PT, HeliaRUS)"	"pt-PT-HeliaRUS"
ro-RO	Rumano	Male	"Microsoft Server Speech Text to Speech Voice (ro-RO, Andrei)"	"ro-RO-Andrei"
ru-RU	Ruso	Female	"Microsoft Server Speech Text to Speech Voice (ru-RU, Irina, Apollo)"	"ru-RU-Irina-Apollo"
		Male	"Microsoft Server Speech Text to Speech Voice (ru-RU, Pavel, Apollo)"	"ru-RU-Pavel-Apollo"

<b>CONFIGURACIÓN REGIONAL</b>	<b>IDIOMA</b>	<b>SEXO</b>	<b>ASIGNACIÓN DE NOMBRE DE SERVICIO COMPLETO</b>	<b>NOMBRE CORTO</b>
		Female	"Microsoft Server Speech Text to Speech Voice (ru-RU, EkaterinaRUS)"	ru-RU-EkaterinaRUS
sk-SK	Eslovaco	Male	"Microsoft Server Speech Text to Speech Voice (sk-SK, Filip)"	"sk-SK-Filip"
sl-SI	Esloveno	Male	"Microsoft Server Speech Text to Speech Voice (sl-SI, Lado)"	"sl-SI-Lado"
sv-SE	Sueco	Female	"Microsoft Server Speech Text to Speech Voice (sv-SE, HedvigRUS)"	"sv-SE-HedvigRUS"
ta-IN	Tamil (India)	Male	"Microsoft Server Speech Text to Speech Voice (ta-IN, Valluvar)"	"ta-IN-Valluvar"
te-IN	Telugu (India)	Female	"Microsoft Server Speech Text to Speech Voice (te-IN, Chitra)"	"te-IN-Chitra"
th-TH	Tailandés	Male	"Microsoft Server Speech Text to Speech Voice (th-TH, Pattara)"	"th-TH-Pattara"
tr-TR	Turco	Female	"Microsoft Server Speech Text to Speech Voice (tr-TR, SedarUS)"	"tr-TR-SedaRUS"
vi-VN	Vietnamita	Male	"Microsoft Server Speech Text to Speech Voice (vi-VN, An)"	"vi-VN-An"
zh-CN	Chino (continental)	Female	"Microsoft Server Speech Text to Speech Voice (zh-CN, HuihuiRUS)"	"zh-CN-HuihuiRUS"
		Female	"Microsoft Server Speech Text to Speech Voice (zh-CN, Yaoyao, Apollo)"	"zh-CN-Yaoyao-Apollo"

CONFIGURACIÓN REGIONAL	IDIOMA	SEXO	ASIGNACIÓN DE NOMBRE DE SERVICIO COMPLETO	NOMBRE CORTO
		Male	"Microsoft Server Speech Text to Speech Voice (zh-CN, Kangkang, Apollo)"	"zh-CN-Kangkang-Apollo"
zh-HK	Chino (RAE de Hong Kong)	Female	"Microsoft Server Speech Text to Speech Voice (zh-HK, Tracy, Apollo)"	"zh-HK-Tracy-Apollo"
		Female	"Microsoft Server Speech Text to Speech Voice (zh-HK, TracyRUS)"	"zh-HK-TracyRUS"
		Male	"Microsoft Server Speech Text to Speech Voice (zh-HK, Danny, Apollo)"	"zh-HK-Danny-Apollo"
zh-TW	Chino (Taiwán)	Female	"Microsoft Server Speech Text to Speech Voice (zh-TW, Yating, Apollo)"	"zh-TW-Yating-Apollo"
		Female	"Microsoft Server Speech Text to Speech Voice (zh-TW, HanHanRUS)"	"zh-TW-HanHanRUS"
		Male	"Microsoft Server Speech Text to Speech Voice (zh-TW, Zhiwei, Apollo)"	"zh-TW-Zhiwei-Apollo"

+ ar-EG admite el árabe estándar moderno (MSA).

#### NOTE

Puede usar la asignación de nombre de servicio completo o el nombre corto de voz en las solicitudes de síntesis de voz.

#### Personalización

La personalización de la voz está disponible para de-DE, en-GB, en-IN, en-US, es-MX, fr-FR, it-IT, pt-BR y zh-CN. Seleccione la configuración regional adecuada que coincide con los datos de entrenamiento que tiene para entrenar un modelo de voz personalizado. Por ejemplo, si los datos de grabación que tienen que hablar en inglés con acento británico, seleccione en-GB.

#### NOTE

No se admite el entrenamiento de modelos bilingües en la funcionalidad de voz personalizada, excepto en el caso de chino-inglés bilingüe. Seleccione la opción de chino-inglés bilingüe si quiere entrenar una voz china que pueda hablar también inglés. El entrenamiento de la voz de todas las configuraciones regionales se inicia con un conjunto de datos de más de 2000 expresiones, excepto en el caso de `en-US` y `zh-CN`, que puede comenzar con cualquier tamaño de datos de entrenamiento.

## Traducción de voz

**Speech Translation API** admite varios idiomas para la traducción de voz a voz y de texto a voz. El idioma de origen debe incluirse siempre en la siguiente tabla de conversión de voz en texto. Los idiomas de destino admitidos dependen de si el destino de traducción es voz o texto. Puede traducir la voz entrante en más de [60 idiomas](#). Un subconjunto de estos idiomas está disponible para la [síntesis de voz](#).

### Idiomas de texto

IDIOMA DE TEXTO	CÓDIGO DE LENGUAJE
Afrikáans	<code>af</code>
Árabe	<code>ar</code>
Bengalí	<code>bn</code>
Bosnio (latino)	<code>bs</code>
Búlgaro	<code>bg</code>
Cantonés (tradicional)	<code>yue</code>
Catalán	<code>ca</code>
Chino simplificado	<code>zh-Hans</code>
Chino tradicional	<code>zh-Hant</code>
Croata	<code>hr</code>
Checo	<code>cs</code>
Danés	<code>da</code>
Neerlandés	<code>nl</code>
Inglés	<code>en</code>
Estonio	<code>et</code>
Fiyiano	<code>fj</code>

IDIOMA DE TEXTO	CÓDIGO DE LENGUAJE
Filipino	fil
Finés	fi
Francés	fr
Alemán	de
Griego	el
Criollo haitiano	ht
Hebreo	he
Hindi	hi
Hmong Daw	mww
Húngaro	hu
Indonesio	id
Italiano	it
Japonés	ja
Kiswahili	sw
Klingon	t1h
Klingon (plqaD)	t1h-Qaak
Coreano	ko
Letón	lv
Lituano	lt
Malgache	mg
Malayo	ms
Maltés	mt
Noruego	nb
Persa	fa

IDIOMA DE TEXTO	CÓDIGO DE LENGUAJE
Polaco	pl
Portugués	pt
Otomí Querétaro	otq
Rumano	ro
Ruso	ru
Samoano	sm
Serbio (cirílico)	sr-Cyrl
Serbio (latino)	sr-Latn
Eslovaco	sk
Esloveno	sl
Español	es
Sueco	sv
Tahitiano	ty
Tamil	ta
Telugu	te
Tailandés	th
Tonganó	to
Turco	tr
Ucraniano	uk
Urdu	ur
Vietnamita	vi
Galés	cy
Maya Yucateco	yua

Pasos siguientes

- Obtener la suscripción de evaluación gratuita del servicio de voz
- Vea cómo funciona el reconocimiento de voz en C#

# ¿Qué es la conversión de voz a texto?

15/01/2020 • 6 minutes to read • [Edit Online](#)

La característica de conversión de voz en texto del servicio de voz, también conocida como reconocimiento de voz, permite la transcripción en tiempo real de las secuencias de audio en texto. Las aplicaciones, las herramientas o los dispositivos pueden consumir y mostrar este texto como una entrada de comando, así como manipularlo. Este servicio funciona con la misma tecnología de reconocimiento que Microsoft utiliza para los productos de Cortana y Office. Funciona sin problemas con las ofertas de servicio de [traducción](#) y [conversión de texto en voz](#). Si desea obtener una lista completa de los idiomas disponibles para la conversión de voz a texto, consulte [Idiomas admitidos](#).

De forma predeterminada, el servicio de conversión de voz en texto utiliza el modelo de lenguaje universal. Este modelo se entrenó con datos propiedad de Microsoft y se implementa en la nube. Resulta óptimo para escenarios de conversación y dictado. Si usa la conversión de voz en texto para el reconocimiento y la transcripción en un entorno único, puede crear y entrenar modelos acústicos, de lenguaje y pronunciación personalizados. La personalización es útil para abordar el ruido ambiente o el vocabulario específico del sector.

## NOTE

Bing Speech se ha retirado el 15 de octubre de 2019. Si sus aplicaciones, herramientas o productos usan Bing Speech API o Custom Speech, hemos creado guías para que le ayuden a migrar al servicio de voz.

- [Migración de Bing Speech al servicio de voz](#)
- [Migración de Custom Speech al servicio de voz](#)

## Introducción a la conversión de voz a texto

El servicio de conversión de voz a texto está disponible a través del [SDK de voz](#). Existen varios escenarios comunes disponibles como inicios rápidos, en diferentes lenguajes y plataformas:

- [Inicio rápido: Reconocimiento de voz con entrada de micrófono](#)
- [Inicio rápido: Reconocimiento de voz a partir de un archivo](#)
- [Inicio rápido: Reconocimiento de voz almacenada en Blob Storage](#)

Si prefiere usar el servicio REST de voz a texto, consulte [API REST](#).

## Tutoriales y ejemplo de código

Una vez que haya tenido la oportunidad de usar el servicio de voz, pruebe nuestro tutorial, que le enseña a reconocer intenciones a partir de contenido de voz mediante el SDK de voz y LUIS.

- [Tutorial: Reconocimiento de intenciones a partir de la voz con el SDK de Voz y LUIS con C#](#)

Hay un ejemplo de código para el SDK de voz disponible en GitHub. En estos ejemplos se tratan escenarios comunes como la lectura de audio de un archivo o flujo, el reconocimiento continuo y de una sola emisión, y el trabajo con modelos personalizados.

- [Ejemplos de conversión de voz a texto \(SDK\)](#)
- [Ejemplos de transcripción de Azure Batch \(REST\)](#)

## Personalización

Además del modelo de servicio de voz estándar, puede crear modelos personalizados. La personalización ayuda a eliminar las barreras del reconocimiento de voz, como el estilo de habla, el vocabulario y el ruido de fondo. Consulte [Custom Speech](#). Las opciones de personalización varían según el idioma o la configuración regional (consulte los [idiomas admitidos](#) para comprobar la compatibilidad).

## Documentos de referencia

El servicio de voz proporciona dos SDK. El primer SDK es el [SDK de voz](#) principal y proporciona la mayoría de las funcionalidades necesarias para interactuar con el servicio de voz. El segundo SDK es específico de los dispositivos, denominado correctamente [SDK de dispositivos de voz](#). Ambos SDK están disponibles en muchos idiomas.

### Documentos de referencia del SDK de voz

Use la lista siguiente para encontrar los documentos de referencia del SDK de voz adecuado:

- [SDK de C#](#)
- [SDK de C++](#)
- [SDK de Java](#)
- [SDK de Python](#)
- [SDK de JavaScript](#)
- [SDK de Objective-C](#)

#### TIP

El SDK del servicio de voz se mantiene y actualiza de forma activa. Para realizar un seguimiento de los cambios, las actualizaciones y las adiciones de características, consulte las [notas de la versión del SDK de voz](#).

### Documentos de referencia del SDK de dispositivos de voz

El [SDK de dispositivos de voz](#) es un superconjunto del SDK de voz, con funcionalidad ampliada para dispositivos específicos. Para descargar el SDK de dispositivos de voz, primero debe [elegir un kit de desarrollo](#).

### Referencias de la API de REST

Para obtener referencias de las distintas API de REST del servicio de voz, consulte la siguiente lista:

- [API REST: Speech-to-text](#) (API de REST: Voz a texto)
- [API REST: Text-to-speech](#) (API de REST: Texto a voz)
- [API REST: Transcripción y personalización de Batch](#)

## Pasos siguientes

- [Obtenga una clave de suscripción gratuita a los servicios de Voz](#)
- [Obtención del SDK de voz](#)

# Inicio rápido: Reconocimiento de voz de un archivo de audio

16/01/2020 • 33 minutes to read • [Edit Online](#)

En este inicio rápido usará el [SDK de Voz](#) para reconocer la voz que procede de un archivo de audio. Una vez que se cumplen los requisitos previos, para realizar el reconocimiento de voz a través de un archivo solo son necesarios cinco pasos:

- Cree un objeto `SpeechConfig` a partir de la clave y la región de suscripción.
- Cree un objeto `AudioConfig` que especifique el nombre de archivo .WAV.
- Cree un objeto `SpeechRecognizer` con los objetos `SpeechConfig` y `AudioConfig` anteriores.
- Con el objeto `SpeechRecognizer`, inicie el proceso de reconocimiento de una única expresión.
- Inspeccione el objeto `SpeechRecognitionResult` devuelto.

Si prefiere ponerse a trabajar de inmediato, vea o descargue todos los [ejemplos para C# del SDK de Voz](#) en GitHub. Si no, vamos a comenzar.

## Requisitos previos

Antes de comenzar, compruebe lo siguiente:

- [Ha creado un recurso de Voz de Azure](#)
- [Ha configurado el entorno de desarrollo](#)
- [Ha creado un proyecto de ejemplo vacío](#)

## Formato de entrada de audio compatible

El SDK de Voz usa el formato siguiente para la entrada de audio.

FORMATO	CÓDEC	BITRATE	VELOCIDAD DE MUESTREO	CANALES
WAV	PCM	16 bits	8 kHz o 16 kHz	1 (mono)

## Abra el proyecto en Visual Studio.

El primer paso es asegurarse de que tiene el proyecto abierto en Visual Studio.

1. Inicie Visual Studio 2019.
2. Cargue el proyecto y abra `Program.cs`.

## Inicio con código reutilizable

Vamos a agregar código que funcione como el esqueleto del proyecto. Tenga en cuenta que ha creado un método asíncrono llamado `RecognizeSpeechAsync()`.

```
using System;
using System.Threading.Tasks;
using Microsoft.CognitiveServices.Speech;

namespace helloworld
{
    class Program
    {
        public static async Task RecognizeSpeechAsync()
        {
        }

        static void Main()
        {
            RecognizeSpeechAsync().Wait();
        }
    }
}
```

## Creación de una configuración de Voz

Antes de inicializar un objeto `SpeechRecognizer`, debe crear una configuración que use la clave y la región de suscripción. Inserte este código en el método `RecognizeSpeechAsync()`.

### NOTE

En este ejemplo se usa el método `FromSubscription()` para compilar la clase `SpeechConfig`. Para ver una lista completa de los métodos disponibles, consulte [Clase SpeechConfig](#). El SDK de Voz se usará de forma predeterminada para reconocer el uso de en-us como idioma. Para más información sobre cómo elegir el idioma de origen, consulte [Especificación del idioma de origen para la conversión de voz a texto](#).

```
var config = SpeechConfig.FromSubscription("YourSubscriptionKey", "YourServiceRegion");
```

## Creación de una configuración de audio

Ahora, debe crear un objeto `AudioConfig` que apunte al archivo de audio. Este objeto se crea dentro de una instrucción `using` para garantizar la versión correcta de los recursos no administrados. Inserte este código en el método `RecognizeSpeechAsync()`, justo debajo de la configuración de Voz.

```
using (var audioInput = AudioConfig.FromWavFileInput(@"whatstheweatherlike.wav"))
{}
```

## Inicialización de un objeto `SpeechRecognizer`

Ahora, se creará el objeto `SpeechRecognizer` con los objetos `SpeechConfig` y `AudioConfig` creados anteriormente. Este objeto se crea también dentro de una instrucción `using` para garantizar la versión correcta de los recursos no administrados. Inserte este código en el método `RecognizeSpeechAsync()`, dentro de la instrucción `using` que encapsula el objeto `AudioConfig`.

```
using (var recognizer = new SpeechRecognizer(config, audioInput))
{
}
```

## Reconocimiento de una frase

En el objeto `SpeechRecognizer`, va a llamar al método `RecognizeOnceAsync()`. Este método permite que el servicio Voz sepa que solo va a enviar una frase para el reconocimiento y que, una vez que se identifica la frase, se detendrá el reconocimiento de voz.

Dentro de la instrucción `using`, agregue este código:

```
Console.WriteLine("Recognizing first result...");
var result = await recognizer.RecognizeOnceAsync();
```

## Visualización de los resultados (o errores) del reconocimiento

Cuando el servicio Voz devuelva el resultado del reconocimiento, querrá hacer algo con él. Vamos a hacer algo tan sencillo como imprimir el resultado en la consola.

Dentro de la instrucción `using`, debajo de `RecognizeOnceAsync()`, agregue este código:

```
if (result.Reason == ResultReason.RecognizedSpeech)
{
    Console.WriteLine($"We recognized: {result.Text}");
}
else if (result.Reason == ResultReason.NoMatch)
{
    Console.WriteLine($"NOMATCH: Speech could not be recognized.");
}
else if (result.Reason == ResultReason.Canceled)
{
    var cancellation = CancellationDetails.FromResult(result);
    Console.WriteLine($"CANCELED: Reason={cancellation.Reason}");

    if (cancellation.Reason == CancellationReason.Error)
    {
        Console.WriteLine($"CANCELED: ErrorCode={cancellation.ErrorCode}");
        Console.WriteLine($"CANCELED: ErrorDetails={cancellation.ErrorDetails}");
        Console.WriteLine($"CANCELED: Did you update the subscription info?");
    }
}
```

## Comprobación del código

En este momento, el código debe tener esta apariencia:

```

// Copyright (c) Microsoft. All rights reserved.
// Licensed under the MIT license. See LICENSE.md file in the project root for full license information.
//

using System;
using System.Threading.Tasks;
using Microsoft.CognitiveServices.Speech;

namespace helloworld
{
    class Program
    {
        public static async Task RecognizeSpeechAsync()
        {
            var config = SpeechConfig.FromSubscription("YourSubscriptionKey", "YourServiceRegion");

            using (var audioInput = AudioConfig.FromWavFileInput(@"whatstheweatherlike.wav"))
            {
                using (var recognizer = new SpeechRecognizer(config, audioInput))
                {
                    Console.WriteLine("Recognizing first result...");
                    var result = await recognizer.RecognizeOnceAsync();

                    if (result.Reason == ResultReason.RecognizedSpeech)
                    {
                        Console.WriteLine($"We recognized: {result.Text}");
                    }
                    else if (result.Reason == ResultReason.NoMatch)
                    {
                        Console.WriteLine($"NOMATCH: Speech could not be recognized.");
                    }
                    else if (result.Reason == ResultReason.Canceled)
                    {
                        var cancellation = CancellationDetails.FromResult(result);
                        Console.WriteLine($"CANCELED: Reason={cancellation.Reason}");

                        if (cancellation.Reason == CancellationReason.Error)
                        {
                            Console.WriteLine($"CANCELED: ErrorCode={cancellation.ErrorCode}");
                            Console.WriteLine($"CANCELED: ErrorDetails={cancellation.ErrorDetails}");
                            Console.WriteLine($"CANCELED: Did you update the subscription info?");
                        }
                    }
                }
            }

            static void Main()
            {
                RecognizeSpeechAsync().Wait();
            }
        }
    }
}

```

## Compilación y ejecución de la aplicación

Ya está listo para compilar la aplicación y probar el reconocimiento de voz con el servicio Voz.

- Compile el código:** en la barra de menús de Visual Studio, elija **Compilar > Compilar solución**.
- Inicie la aplicación:** en la barra de menús, elija **Depurar > Iniciar depuración** o presione **F5**.
- Inicie el reconocimiento:** el archivo de audio se envía al servicio Voz, se transcribe como texto y se representa en la consola.

```
Recognizing first result...
We recognized: What's the weather like?
```

## Pasos siguientes

### Exploración de ejemplos de C# en GitHub

En este inicio rápido usará el [SDK de Voz](#) para reconocer la voz que procede de un archivo de audio. Una vez que se cumplen los requisitos previos, para realizar el reconocimiento de voz a través de un archivo solo son necesarios cinco pasos:

- Cree un objeto `SpeechConfig` a partir de la clave y la región de suscripción.
- Cree un objeto `AudioConfig` que especifique el nombre de archivo .WAV.
- Cree un objeto `SpeechRecognizer` con los objetos `SpeechConfig` y `AudioConfig` anteriores.
- Con el objeto `SpeechRecognizer`, inicie el proceso de reconocimiento de una única expresión.
- Inspeccione el objeto `SpeechRecognitionResult` devuelto.

Si prefiere ponerse a trabajar de inmediato, vea o descargue todos los [ejemplos para C++ del SDK de Voz](#) en GitHub. Si no, vamos a comenzar.

### Selección del entorno de destino

- [Linux](#)
- [macOS](#)
- [Windows](#)

## Requisitos previos

Antes de comenzar, compruebe lo siguiente:

- [Ha creado un recurso de Voz de Azure](#)
- [Ha configurado el entorno de desarrollo](#)
- [Ha creado un proyecto de ejemplo vacío](#)

## Formato de entrada de audio compatible

El SDK de Voz usa el formato siguiente para la entrada de audio.

FORMATO	CÓDEC	BITRATE	VELOCIDAD DE MUESTREO	CANALES
WAV	PCM	16 bits	8 kHz o 16 kHz	1 (mono)

## Incorporación de código de ejemplo

1. Cree un archivo de código fuente C++ denominado `helloworld.cpp` y pegue el siguiente código en él.

```

// Creates an instance of a speech config with specified subscription key and service region.
// Replace with your own subscription key and service region (e.g., "westus").
auto config = SpeechConfig::FromSubscription("YourSubscriptionKey", "YourServiceRegion");

// Creates a speech recognizer using a WAV file. The default language is "en-us".
// Replace with your own audio file name.
auto audioInput = AudioConfig::FromWavFileInput("whatstheweatherlike.wav");
auto recognizer = SpeechRecognizer::FromConfig(config, audioInput);
cout << "Recognizing first result...\n";

// Starts speech recognition, and returns after a single utterance is recognized. The end of a
// single utterance is determined by listening for silence at the end or until a maximum of 15
// seconds of audio is processed. The task returns the recognition text as result.
// Note: Since RecognizeOnceAsync() returns only a single utterance, it is suitable only for single
// shot recognition like command or query.
// For long-running multi-utterance recognition, use StartContinuousRecognitionAsync() instead.
auto result = recognizer->RecognizeOnceAsync().get();

// Checks result.
if (result->Reason == ResultReason::RecognizedSpeech)
{
    cout << "RECOGNIZED: Text=" << result->Text << std::endl;
}
else if (result->Reason == ResultReason::NoMatch)
{
    cout << "NOMATCH: Speech could not be recognized." << std::endl;
}
else if (result->Reason == ResultReason::Canceled)
{
    auto cancellation = CancellationDetails::FromResult(result);
    cout << "CANCELED: Reason=" << (int)cancellation->Reason << std::endl;

    if (cancellation->Reason == CancellationReason::Error)
    {
        cout << "CANCELED: ErrorCode=" << (int)cancellation->ErrorCode << std::endl;
        cout << "CANCELED: ErrorDetails=" << cancellation->ErrorDetails << std::endl;
        cout << "CANCELED: Did you update the subscription info?" << std::endl;
    }
}
}

```

2. En este nuevo archivo, reemplace la cadena `YourSubscriptionKey` por su clave de suscripción del servicio Voz.
3. Reemplace la cadena `YourServiceRegion` por la [región](#) asociada a sus suscripción (por ejemplo, `westus` para la suscripción de evaluación gratuita).
4. Reemplace la cadena `whatstheweatherlike.wav` por su propio nombre de archivo.

#### **NOTE**

El SDK de Voz se usará de forma predeterminada para reconocer el uso de en-us como idioma. Para más información sobre cómo elegir el idioma de origen, consulte [Especificación del idioma de origen para la conversión de voz a texto](#).

## Compilación de la aplicación

#### NOTE

Asegúrese de introducir los siguientes comandos como una *única línea de comandos*. La forma más fácil de hacerlo es copiar el comando usando el botón **Copiar** junto a cada comando, y luego pegarlo en el símbolo del shell.

- En un sistema **x64** (64 bits), ejecute el siguiente comando para crear la aplicación.

```
g++ helloworld.cpp -o helloworld -I "$SPEECHSDK_ROOT/include/cxx_api" -I  
"$SPEECHSDK_ROOT/include/c_api" --std=c++14 -lpthread -lMicrosoft.CognitiveServices.Speech.core -L  
"$SPEECHSDK_ROOT/lib/x64" -l:libasound.so.2
```

- En un sistema **x86** (32 bits), ejecute el siguiente comando para crear la aplicación.

```
g++ helloworld.cpp -o helloworld -I "$SPEECHSDK_ROOT/include/cxx_api" -I  
"$SPEECHSDK_ROOT/include/c_api" --std=c++14 -lpthread -lMicrosoft.CognitiveServices.Speech.core -L  
"$SPEECHSDK_ROOT/lib/x86" -l:libasound.so.2
```

- En un sistema **ARM64** (64 bits), ejecute el siguiente comando para crear la aplicación.

```
g++ helloworld.cpp -o helloworld -I "$SPEECHSDK_ROOT/include/cxx_api" -I  
"$SPEECHSDK_ROOT/include/c_api" --std=c++14 -lpthread -lMicrosoft.CognitiveServices.Speech.core -L  
"$SPEECHSDK_ROOT/lib/arm64" -l:libasound.so.2
```

## Ejecución de la aplicación

1. Configuración de la ruta de acceso de la biblioteca del cargador para que apunte a la biblioteca del SDK de Voz.

- En un sistema **x64** (64 bits), escriba el siguiente comando.

```
export LD_LIBRARY_PATH="$LD_LIBRARY_PATH:$SPEECHSDK_ROOT/lib/x64"
```

- En un sistema **x86** (32 bits), escriba el siguiente comando.

```
export LD_LIBRARY_PATH="$LD_LIBRARY_PATH:$SPEECHSDK_ROOT/lib/x86"
```

- En un sistema **ARM64** (64 bits), escriba el siguiente comando.

```
export LD_LIBRARY_PATH="$LD_LIBRARY_PATH:$SPEECHSDK_ROOT/lib/arm64"
```

2. Ejecute la aplicación.

```
./helloworld
```

3. El archivo de audio se transmite al servicio de voz y la primera expresión del archivo se transcribe a texto, que aparece en la misma ventana.

```
Recognizing first result...  
We recognized: What's the weather like?
```

# Pasos siguientes

## Exploración de ejemplos de C++ en GitHub

En este inicio rápido usará el [SDK de Voz](#) para reconocer la voz que procede de un archivo de audio. Una vez que se cumplen los requisitos previos, para realizar el reconocimiento de voz a través de un archivo solo son necesarios cinco pasos:

- Cree un objeto `SpeechConfig` a partir de la clave y la región de suscripción.
- Cree un objeto `AudioConfig` que especifique el nombre de archivo .WAV.
- Cree un objeto `SpeechRecognizer` con los objetos `SpeechConfig` y `AudioConfig` anteriores.
- Con el objeto `SpeechRecognizer`, inicie el proceso de reconocimiento de una única expresión.
- Inspeccione el objeto `SpeechRecognitionResult` devuelto.

Si prefiere ponerse a trabajar de inmediato, vea o descargue todos los [ejemplos para Java del SDK de Voz](#) en GitHub. Si no, vamos a comenzar.

## Prerequisites

- [Ha creado un recurso de Voz de Azure](#)
- [Ha configurado el entorno de desarrollo](#)
- [Ha creado un proyecto de ejemplo vacío](#)

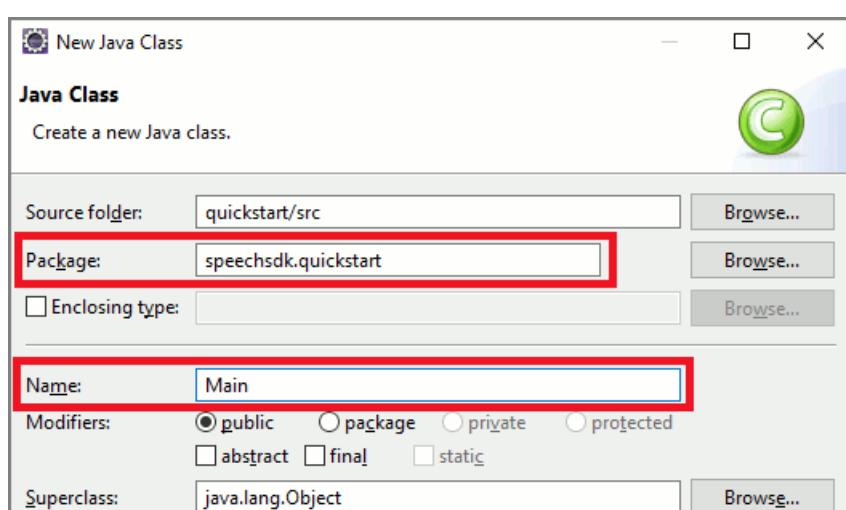
## Formato de entrada de audio compatible

El SDK de Voz usa el formato siguiente para la entrada de audio.

FORMATO	CÓDEC	BITRATE	VELOCIDAD DE MUESTREO	CANALES
WAV	PCM	16 bits	8 kHz o 16 kHz	1 (mono)

## Incorporación de código de ejemplo

1. Para agregar una nueva clase vacía al proyecto de Java, seleccione **File (Archivo) > New (Nuevo) > Class (Clase)**.
2. En la ventana **New Java Class** (Nueva clase de Java) escriba **speechsdk.quickstart** en el campo **Package** (Paquete) y **Main** en el campo **Name** (Nombre).



3. Reemplace el código en `Main.java` con el siguiente fragmento de código:

```
package speechsdk.quickstart;

import java.util.concurrent.Future;
import com.microsoft.cognitiveservices.speech.*;

/**
 * Quickstart: recognize speech using the Speech SDK for Java.
 */
public class Main {

    /**
     * @param args Arguments are ignored in this sample.
     */
    public static void main(String[] args) {
        try {
            // Replace below with your own subscription key
            String speechSubscriptionKey = "YourSubscriptionKey";
            // Replace below with your own service region (e.g., "westus").
            String serviceRegion = "YourServiceRegion";
            // Replace below with your own filename.
            String audioFileName = "whatstheweatherlike.wav";

            int exitCode = 1;
            SpeechConfig config = SpeechConfig.fromSubscription(speechSubscriptionKey, serviceRegion);
            assert(config != null);

            AudioConfig audioInput = AudioConfig.fromWavFileInput(audioFileName);
            assert(audioInput != null);

            SpeechRecognizer reco = new SpeechRecognizer(config, audioInput);
            assert(reco != null);

            System.out.println("Recognizing first result...");

            Future<SpeechRecognitionResult> task = reco.recognizeOnceAsync();
            assert(task != null);

            SpeechRecognitionResult result = task.get();
            assert(result != null);

            if (result.getReason() == ResultReason.RecognizedSpeech) {
                System.out.println("We recognized: " + result.getText());
                exitCode = 0;
            }
            else if (result.getReason() == ResultReason.NoMatch) {
                System.out.println("NOMATCH: Speech could not be recognized.");
            }
            else if (result.getReason() == ResultReason.Canceled) {
                CancellationDetails cancellation = CancellationDetails.fromResult(result);
                System.out.println("CANCELED: Reason=" + cancellation.getReason());

                if (cancellation.getReason() == CancellationReason.Error) {
                    System.out.println("CANCELED: ErrorCode=" + cancellation.getErrorCode());
                    System.out.println("CANCELED: ErrorDetails=" + cancellation.getErrorDetails());
                    System.out.println("CANCELED: Did you update the subscription info?");
                }
            }
            reco.close();

            System.exit(exitCode);
        } catch (Exception ex) {
            System.out.println("Unexpected exception: " + ex.getMessage());
            assert(false);
        }
    }
}
```

```
        System.out.println("Error occurred during recording or audio processing.");
        System.exit(1);
    }
}
```

4. Reemplace la cadena `YourSubscriptionKey` por la clave de suscripción.
5. Reemplace la cadena `YourServiceRegion` por la [región](#) asociada a sus suscripción (por ejemplo, `westus` para la suscripción de evaluación gratuita).
6. Reemplace la cadena `whatstheweatherlike.wav` por su propio nombre de archivo.
7. Guarde los cambios en el proyecto.

**NOTE**

El SDK de Voz se usará de forma predeterminada para reconocer el uso de en-us como idioma. Para más información sobre cómo elegir el idioma de origen, consulte [Especificación del idioma de origen para la conversión de voz a texto](#).

## Compilación y ejecución de la aplicación

Presione F11, o seleccione **Run (Ejecutar) > Debug (Depurar)**. Los primeros 15 segundos de la entrada de voz a través del archivo de audio se reconocen y registran en la ventana de consola.

```
Recognizing first result...
We recognized: What's the weather like?
```

## Pasos siguientes

### [Exploración de ejemplos de Java en GitHub](#)

En este inicio rápido usará el [SDK de Voz](#) para reconocer la voz que procede de un archivo de audio. Una vez que se cumplen los requisitos previos, para realizar el reconocimiento de voz a través de un archivo solo son necesarios cinco pasos:

- Cree un objeto `SpeechConfig` a partir de la clave y la región de suscripción.
- Cree un objeto `AudioConfig` que especifique el nombre de archivo .WAV.
- Cree un objeto `SpeechRecognizer` con los objetos `SpeechConfig` y `AudioConfig` anteriores.
- Con el objeto `SpeechRecognizer`, inicie el proceso de reconocimiento de una única expresión.
- Inspeccione el objeto `SpeechRecognitionResult` devuelto.

Si prefiere ponerse a trabajar de inmediato, vea o descargue todos los [ejemplos para Python del SDK de Voz](#) en GitHub. Si no, vamos a comenzar.

## Requisitos previos

Antes de comenzar, compruebe lo siguiente:

- [Ha creado un recurso de Voz de Azure](#)
- [Ha creado una aplicación de LUIS y ha obtenido una clave de punto de conexión](#)
- [Ha configurado el entorno de desarrollo](#)
- [Ha creado un proyecto de ejemplo vacío](#)

# Formato de entrada de audio compatible

El SDK de Voz usa el formato siguiente para la entrada de audio.

FORMATO	CÓDEC	BITRATE	VELOCIDAD DE MUESTREO	CANALES
WAV	PCM	16 bits	8 kHz o 16 kHz	1 (mono)

## Soporte técnico y actualizaciones

Las actualizaciones del paquete de Python del SDK de Voz se distribuirán mediante PyPI y se anunciarán en la página [Notas de la versión](#). Si hay disponible una nueva versión, puede actualizarse a ella con el comando `pip install --upgrade azure-cognitiveservices-speech`. Para comprobar qué versión está instalada actualmente, inspeccione la variable `azure.cognitiveservices.speech._version_`.

Si tiene un problema o falta una característica, consulte las [opciones de ayuda y soporte técnico](#).

## Creación de una aplicación de Python mediante el SDK de Voz

### Ejecución del ejemplo

Puede copiar el [código de ejemplo](#) de este inicio rápido en un archivo de código fuente `quickstart.py` y ejecutarlo en el IDE o en la consola:

```
python quickstart.py
```

También, puede descargar este tutorial de inicio rápido como un cuaderno de [Jupyter](#) del [repositorio de ejemplos del SDK de Voz](#) y ejecutarlo como un cuaderno.

### Código de ejemplo

#### NOTE

El SDK de Voz se usará de forma predeterminada para reconocer el uso de en-us como idioma. Para más información sobre cómo elegir el idioma de origen, consulte [Especificación del idioma de origen para la conversión de voz a texto](#).

```

import azure.cognitiveservices.speech as speechsdk

# Creates an instance of a speech config with specified subscription key and service region.
# Replace with your own subscription key and service region (e.g., "westus").
speech_key, service_region = "YourSubscriptionKey", "YourServiceRegion"
speech_config = speechsdk.SpeechConfig(subscription=speech_key, region=service_region)

# Creates an audio configuration that points to an audio file.
# Replace with your own audio filename.
audio_filename = "whatstheweatherlike.wav"
audio_input = speechsdk.AudioConfig(filename=audio_filename)

# Creates a recognizer with the given settings
speech_recognizer = speechsdk.SpeechRecognizer(speech_config=speech_config, audio_config=audio_input)

print("Recognizing first result...")

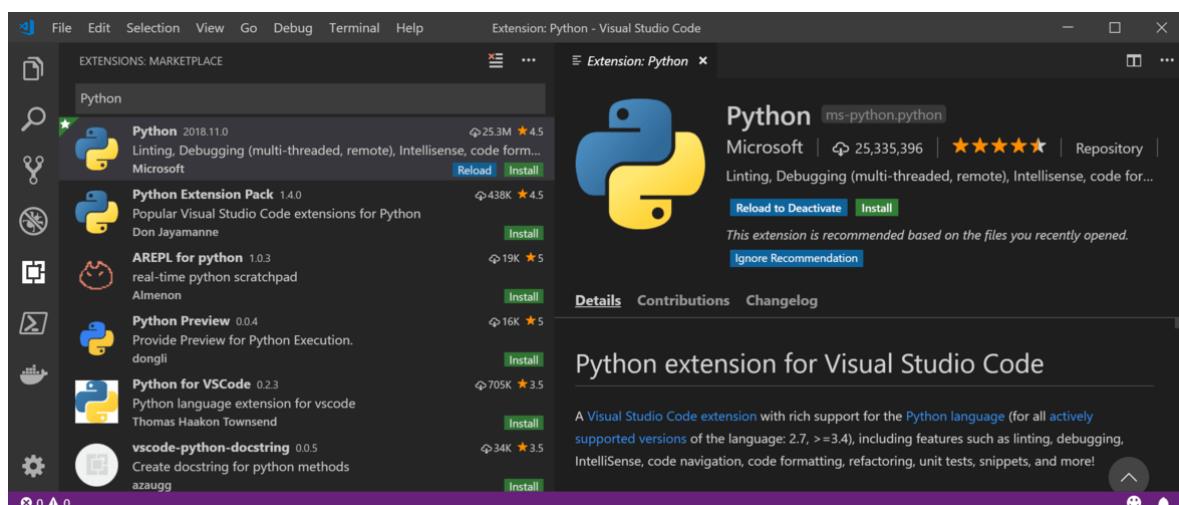
# Starts speech recognition, and returns after a single utterance is recognized. The end of a
# single utterance is determined by listening for silence at the end or until a maximum of 15
# seconds of audio is processed. The task returns the recognition text as result.
# Note: Since recognize_once() returns only a single utterance, it is suitable only for single
# shot recognition like command or query.
# For long-running multi-utterance recognition, use start_continuous_recognition() instead.
result = speech_recognizer.recognize_once()

# Checks result.
if result.reason == speechsdk.ResultReason.RecognizedSpeech:
    print("Recognized: {}".format(result.text))
elif result.reason == speechsdk.ResultReason.NoMatch:
    print("No speech could be recognized: {}".format(result.no_match_details))
elif result.reason == speechsdk.ResultReason.Canceled:
    cancellation_details = result.cancellation_details
    print("Speech Recognition canceled: {}".format(cancellation_details.reason))
    if cancellation_details.reason == speechsdk.CancellationReason.Error:
        print("Error details: {}".format(cancellation_details.error_details))

```

## Instalación y uso del SDK de Voz con Visual Studio Code

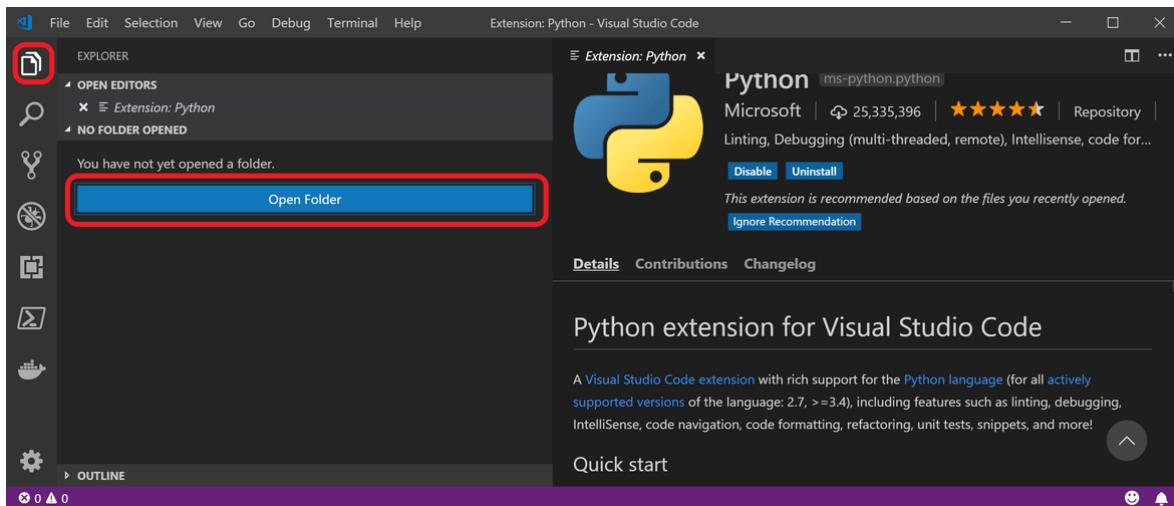
1. Descargue e instale una versión de 64 bits (3.5 o posterior) de [Python](#) en el equipo.
2. Descargue e instale [Visual Studio Code](#).
3. Abra Visual Studio Code e instale la extensión de Python. Seleccione **File > Preferences > Extensions** (Archivo > Preferencias > Extensiones) en el menú. Busque **Python**.



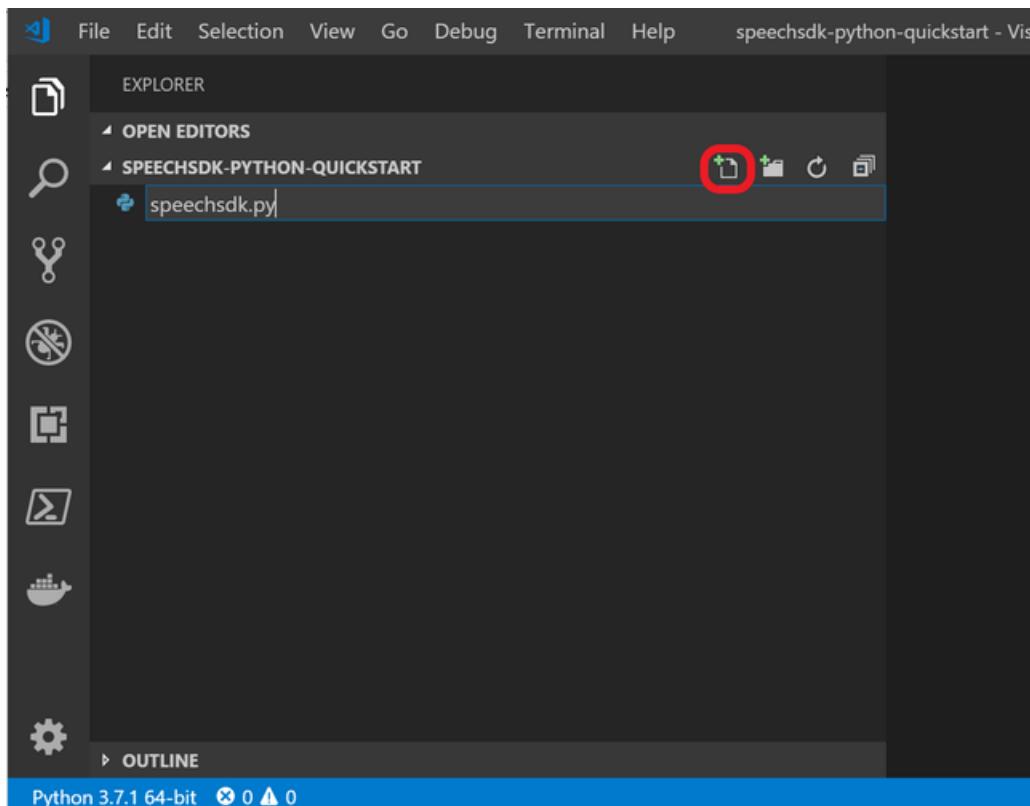
4. Cree una carpeta en la que almacenar el proyecto. Por ejemplo, puede usar para ello el Explorador de

Windows.

5. En Visual Studio Code, seleccione el ícono de **File** (Archivo). A continuación, abra la carpeta que creó.



6. Cree un archivo de código fuente de Python `speechsdk.py` mediante la selección del ícono de nuevo archivo.



7. Copie, pegue y guarde el [código de Python](#) en el archivo recién creado.
8. Inserte la información de la suscripción del servicio Voz.
9. Si se selecciona, se muestra un intérprete de Python en el lado izquierdo de la barra de estado en la parte inferior de la ventana. En caso contrario, aparecerá una lista de los intérpretes de Python disponibles. Abra la paleta de comandos (Ctrl+Mayús+P) y escriba **Python: Select Interpreter** (Seleccionar intérprete). Elija un valor apropiado.
10. Puede instalar el paquete de Python del SDK de Voz desde dentro de Visual Studio Code. Hágalo si no está instalado aún para el intérprete de Python seleccionado. Para instalar el paquete del SDK de Voz, abra un terminal. Abra de nuevo la paleta de comandos (Ctrl+Mayús+P) y escriba **Terminal: Create New Integrated Terminal** (Crear terminal integrado). En el terminal que se abre, escriba el comando

```
python -m pip install azure-cognitiveservices-speech
```

 o el que sea apropiado para su sistema.

11. Para ejecutar el código de ejemplo, haga clic con el botón derecho en algún lugar dentro del editor. Seleccione **Run Python File in Terminal** (Ejecutar archivo de Python en terminal). Los primeros 15 segundos de la entrada de voz a través del archivo de audio se reconocen y registran en la ventana de consola.

```
Recognizing first result...
We recognized: What's the weather like?
```

Si tiene problemas para seguir estas instrucciones, consulte el [tutorial de Python para Visual Studio Code](#) con información más amplia.

## Pasos siguientes

[Exploración de los ejemplos de Python en GitHub](#)

Vea o descargue todos los [ejemplos del SDK de Voz](#) en GitHub.

## Compatibilidad con plataformas y lenguajes adicionales

Si ha hecho clic en esta pestaña, es probable que no vea un inicio rápido en su lenguaje de programación favorito. No se preocupe, tenemos materiales de inicio rápido y ejemplos de código adicionales disponibles en GitHub. Use la tabla para encontrar el ejemplo correcto para su lenguaje de programación y combinación de plataforma y sistema operativo.

IDIOMA	INICIOS RÁPIDOS ADICIONALES	EJEMPLOS DE CÓDIGO
C++		<a href="#">Windows</a> , <a href="#">Linux</a> , <a href="#">macOS</a>
C#		<a href="#">.NET Framework</a> , <a href="#">.NET Core</a> , <a href="#">UWP</a> , <a href="#">Unity</a> , <a href="#">Xamarin</a>
Java		<a href="#">Android</a> , <a href="#">JRE</a>
JavaScript		<a href="#">Browser</a>
Node.js		<a href="#">Windows</a> , <a href="#">Linux</a> , <a href="#">macOS</a>
Objective-C	<a href="#">macOs</a> , <a href="#">iOS</a>	<a href="#">iOS</a> , <a href="#">macOS</a>
Python		<a href="#">Windows</a> , <a href="#">Linux</a> , <a href="#">macOS</a>
Swift	<a href="#">macOs</a> , <a href="#">iOS</a>	<a href="#">iOS</a> , <a href="#">macOS</a>

# Inicio rápido: Reconocimiento de voz a través de un micrófono

16/01/2020 • 69 minutes to read • [Edit Online](#)

En este inicio rápido usará el [SDK de Voz](#) para reconocer interactivamente la voz de la entrada de micrófono y obtener la transcripción del texto del audio capturado. Es fácil integrar esta característica en las aplicaciones o dispositivos para tareas de reconocimiento comunes, como transcribir conversaciones. También se puede usar para integraciones más complejas, como el uso de Bot Framework con el SDK de Voz para crear asistentes de voz.

Una vez que se cumplen los requisitos previos, para realizar el reconocimiento de voz a través de un micrófono solo son necesarios cuatro pasos:

- Cree un objeto `SpeechConfig` a partir de la clave y la región de suscripción.
- Cree un objeto `SpeechRecognizer` con el objeto `SpeechConfig` anterior.
- Con el objeto `SpeechRecognizer`, inicie el proceso de reconocimiento de una única expresión.
- Inspeccione el objeto `SpeechRecognitionResult` devuelto.

Si prefiere ponerse a trabajar de inmediato, vea o descargue todos los [ejemplos para C# del SDK de Voz](#) en GitHub. Si no, vamos a comenzar.

## Selección del entorno de destino

- [.NET](#)
- [Unity](#)
- [UWP](#)
- [Xamarin](#)

## Prerequisites

Antes de comenzar:

- [Ha creado un recurso de Voz de Azure](#)
- [Ha configurado el entorno de desarrollo](#)
- [Ha creado un proyecto de ejemplo vacío](#)
- Asegúrese de que tiene acceso a un micrófono para capturar el audio.

## Abra el proyecto en Visual Studio.

El primer paso es asegurarse de que tiene el proyecto abierto en Visual Studio.

1. Inicie Visual Studio 2019.
2. Cargue el proyecto y abra `Program.cs`.

## Inicio con código reutilizable

Vamos a agregar código que funcione como el esqueleto del proyecto. Tenga en cuenta que ha creado un método asíncrono llamado `RecognizeSpeechAsync()`.

```
using System;
using System.Threading.Tasks;
using Microsoft.CognitiveServices.Speech;

namespace helloworld
{
    class Program
    {
        public static async Task RecognizeSpeechAsync()
        {

        }

        static void Main()
        {
            RecognizeSpeechAsync().Wait();
            Console.WriteLine("Please press <Return> to continue.");
            Console.ReadLine();
        }
    }
}
```

## Creación de una configuración de Voz

Antes de inicializar un objeto `SpeechRecognizer`, debe crear una configuración que use la clave y la región de suscripción. Inserte este código en el método `RecognizeSpeechAsync()`.

### NOTE

En este ejemplo se usa el método `FromSubscription()` para compilar la clase `SpeechConfig`. Para ver una lista completa de los métodos disponibles, consulte [Clase SpeechConfig](#).

```
var config = SpeechConfig.FromSubscription("YourSubscriptionKey", "YourServiceRegion");
```

El SDK de Voz se usará de forma predeterminada para reconocer el uso de en-us como idioma.

Para más información sobre cómo elegir el idioma de origen, consulte [Especificación del idioma de origen para la conversión de voz a texto](#).

## Inicialización de un objeto `SpeechRecognizer`

Ahora, vamos a crear un objeto `SpeechRecognizer`. Este objeto se crea dentro de una instrucción `using` para garantizar la versión correcta de los recursos no administrados. Inserte este código en el método `RecognizeSpeechAsync()`, justo debajo de la configuración de Voz.

```
using (var recognizer = new SpeechRecognizer(config))
{
}
```

## Reconocimiento de una frase

En el objeto `SpeechRecognizer`, va a llamar al método `RecognizeOnceAsync()`. Este método permite que el servicio Voz sepa que solo va a enviar una frase para el reconocimiento y que, una vez que se identifica la frase, se detendrá el reconocimiento de voz.

Dentro de la instrucción using, agregue este código: [!code-csharp]

## Visualización de los resultados (o errores) del reconocimiento

Cuando el servicio Voz devuelva el resultado del reconocimiento, querrá hacer algo con él. Vamos a hacer algo tan sencillo como imprimir el resultado en la consola.

Dentro de la instrucción using, debajo de `RecognizeOnceAsync()`, agregue este código: [!code-csharp]

## Comprobación del código

En este momento, el código debe tener esta apariencia: [!code-csharp]

## Compilación y ejecución de la aplicación

Ya está listo para compilar la aplicación y probar el reconocimiento de voz con el servicio Voz.

1. **Compile el código:** en la barra de menús de Visual Studio, elija **Compilar > Compilar solución**.
2. **Inicie la aplicación:** en la barra de menús, elija **Depurar > Iniciar depuración** o presione **F5**.
3. **Inicie el reconocimiento:** se le pedirá que diga una frase en inglés. La voz se envía al servicio Voz, se transcribe como texto y se representa en la consola.

## Pasos siguientes

[Exploración de ejemplos de C# en GitHub](#)

En este inicio rápido usará el [SDK de Voz](#) para reconocer interactivamente la voz de la entrada de micrófono y obtener la transcripción del texto del audio capturado. Es fácil integrar esta característica en las aplicaciones o dispositivos para tareas de reconocimiento comunes, como transcribir conversaciones. También se puede usar para integraciones más complejas, como el uso de Bot Framework con el SDK de Voz para crear asistentes de voz.

Una vez que se cumplen los requisitos previos, para realizar el reconocimiento de voz a través de un micrófono solo son necesarios cuatro pasos:

- Cree un objeto `SpeechConfig` a partir de la clave y la región de suscripción.
- Cree un objeto `SpeechRecognizer` con el objeto `SpeechConfig` anterior.
- Con el objeto `SpeechRecognizer`, inicie el proceso de reconocimiento de una única expresión.
- Inspeccione el objeto `SpeechRecognitionResult` devuelto.

Si prefiere ponerse a trabajar de inmediato, vea o descargue todos los [ejemplos para C++ del SDK de Voz](#) en GitHub. Si no, vamos a comenzar.

### Selección del entorno de destino

- [Linux](#)
- [macOS](#)
- [Windows](#)

## Prerequisites

Antes de comenzar:

- [Ha creado un recurso de Voz de Azure](#)

- [Ha configurado el entorno de desarrollo](#)
- [Ha creado un proyecto de ejemplo vacío](#)
- Asegúrese de que tiene acceso a un micrófono para capturar el audio.

## Incorporación de código de ejemplo

1. Cree un archivo de código fuente C++ denominado `helloworld.cpp` y pegue el siguiente código en él.

```
#include <iostream> // cin, cout
#include <speechapi_cxx.h>

using namespace std;
using namespace Microsoft::CognitiveServices::Speech;

void recognizeSpeech() {
    // Creates an instance of a speech config with specified subscription key and
    service region.
    // Replace with your own subscription key and service region (e.g., "westus").
    auto config = SpeechConfig::FromSubscription("YourSubscriptionKey",
    "YourServiceRegion");

    // Creates a speech recognizer
    auto recognizer = SpeechRecognizer::FromConfig(config);
    cout << "Say something...\n";

    // Starts speech recognition, and returns after a single utterance is recognized.
    The end of a
    // single utterance is determined by listening for silence at the end or until a
    maximum of 15
    // seconds of audio is processed. The task returns the recognition text as result.
    // Note: Since RecognizeOnceAsync() returns only a single utterance, it is suitable
    only for single
    // shot recognition like command or query.
    // For long-running multi-utterance recognition, use
    StartContinuousRecognitionAsync() instead.
    auto result = recognizer->RecognizeOnceAsync().get();

    // Checks result.
    if (result->Reason == ResultReason::RecognizedSpeech) {
        cout << "We recognized: " << result->Text << std::endl;
    }
    else if (result->Reason == ResultReason::NoMatch) {
        cout << "NOMATCH: Speech could not be recognized." << std::endl;
    }
    else if (result->Reason == ResultReason::Canceled) {
        auto cancellation = CancellationDetails::FromResult(result);
        cout << "CANCELED: Reason=" << (int)cancellation->Reason << std::endl;

        if (cancellation->Reason == CancellationReason::Error) {
            cout << "CANCELED: ErrorCode= " << (int)cancellation->ErrorCode <<
            std::endl;
            cout << "CANCELED: ErrorDetails=" << cancellation->ErrorDetails <<
            std::endl;
            cout << "CANCELED: Did you update the subscription info?" << std::endl;
        }
    }
}

int main(int argc, char **argv) {
    setlocale(LC_ALL, "");
    recognizeSpeech();
    return 0;
}
```

2. En este nuevo archivo, reemplace la cadena `YourSubscriptionKey` por su clave de suscripción del servicio Voz.
3. Reemplace la cadena `YourServiceRegion` por la [región](#) asociada a sus suscripción (por ejemplo, `westus` para la suscripción de evaluación gratuita).

**NOTE**

El SDK de Voz se usará de forma predeterminada para reconocer el uso de en-us como idioma. Para más información sobre cómo elegir el idioma de origen, consulte [Especificación del idioma de origen para la conversión de voz a texto](#).

## Compilación de la aplicación

**NOTE**

Asegúrese de introducir los siguientes comandos como una *única línea de comandos*. La forma más fácil de hacerlo es copiar el comando usando el botón **Copiar** junto a cada comando, y luego pegarlo en el símbolo del shell.

- En un sistema **x64** (64 bits), ejecute el siguiente comando para crear la aplicación.

```
g++ helloworld.cpp -o helloworld -I "$SPEECHSDK_ROOT/include/cxx_api" -I  
"$SPEECHSDK_ROOT/include/c_api" --std=c++14 -lpthread -  
lMicrosoft.CognitiveServices.Speech.core -L "$SPEECHSDK_ROOT/lib/x64" -l:libasound.so.2
```

- En un sistema **x86** (32 bits), ejecute el siguiente comando para crear la aplicación.

```
g++ helloworld.cpp -o helloworld -I "$SPEECHSDK_ROOT/include/cxx_api" -I  
"$SPEECHSDK_ROOT/include/c_api" --std=c++14 -lpthread -  
lMicrosoft.CognitiveServices.Speech.core -L "$SPEECHSDK_ROOT/lib/x86" -l:libasound.so.2
```

- En un sistema **ARM64** (64 bits), ejecute el siguiente comando para crear la aplicación.

```
g++ helloworld.cpp -o helloworld -I "$SPEECHSDK_ROOT/include/cxx_api" -I  
"$SPEECHSDK_ROOT/include/c_api" --std=c++14 -lpthread -  
lMicrosoft.CognitiveServices.Speech.core -L "$SPEECHSDK_ROOT/lib/arm64" -  
l:libasound.so.2
```

## Ejecución la aplicación

1. Configuración de la ruta de acceso de la biblioteca del cargador para que apunte a la biblioteca del SDK de Voz.
  - En un sistema **x64** (64 bits), escriba el siguiente comando.

```
export LD_LIBRARY_PATH="$LD_LIBRARY_PATH:$SPEECHSDK_ROOT/lib/x64"
```

- En un sistema **x86** (32 bits), escriba el siguiente comando.

```
export LD_LIBRARY_PATH="$LD_LIBRARY_PATH:$SPEECHSDK_ROOT/lib/x86"
```

- En un sistema **ARM64** (64 bits), escriba el siguiente comando.

```
export LD_LIBRARY_PATH="$LD_LIBRARY_PATH:$SPEECHSDK_ROOT/lib/arm64"
```

2. Ejecute la aplicación.

```
./helloworld
```

3. En la ventana de consola, aparece un símbolo del sistema que solicita que se diga algo. Diga una oración o frase en inglés. Lo que diga se transmitirá al servicio de Voz y se transcribirá en texto, que aparece en la misma ventana.

```
Say something...
We recognized: What's the weather like?
```

## Pasos siguientes

### [Exploración de ejemplos de C++ en GitHub](#)

En este inicio rápido usará el [SDK de Voz](#) para reconocer interactivamente la voz de la entrada de micrófono y obtener la transcripción del texto del audio capturado. Es fácil integrar esta característica en las aplicaciones o dispositivos para tareas de reconocimiento comunes, como transcribir conversaciones. También se puede usar para integraciones más complejas, como el uso de Bot Framework con el SDK de Voz para crear asistentes de voz.

Una vez que se cumplen los requisitos previos, para realizar el reconocimiento de voz a través de un micrófono solo son necesarios cuatro pasos:

- Cree un objeto `SpeechConfig` a partir de la clave y la región de suscripción.
- Cree un objeto `SpeechRecognizer` con el objeto `SpeechConfig` anterior.
- Con el objeto `SpeechRecognizer`, inicie el proceso de reconocimiento de una única expresión.
- Inspeccione el objeto `SpeechRecognitionResult` devuelto.

Si prefiere ponerse a trabajar de inmediato, vea o descargue todos los [ejemplos para Java del SDK de Voz](#) en GitHub. Si no, vamos a comenzar.

### Selección del entorno de destino

- [Java Runtime](#)
- [Android](#)

## Requisitos previos

Antes de comenzar:

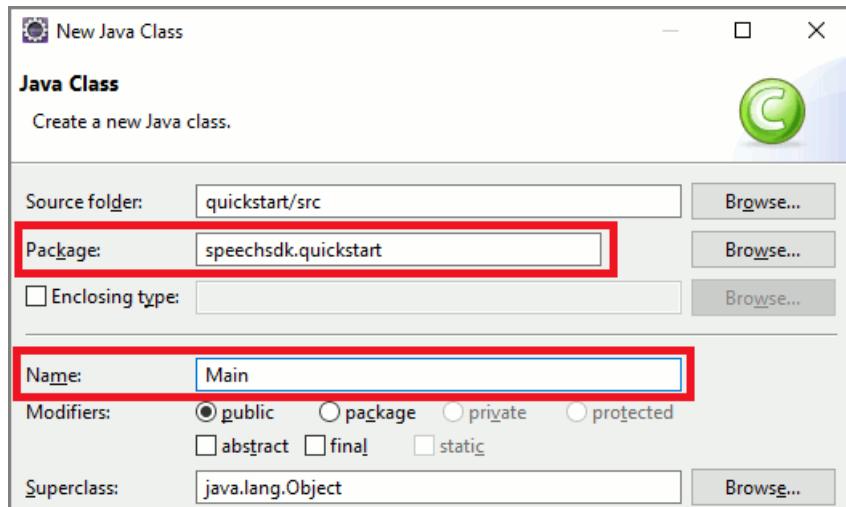
- [Ha creado un recurso de Voz de Azure](#)
- [Ha configurado el entorno de desarrollo](#)
- [Ha creado un proyecto de ejemplo vacío](#)
- Asegúrese de que tiene acceso a un micrófono para capturar el audio.

## Incorporación de código de ejemplo

1. Para agregar una nueva clase vacía al proyecto de Java, seleccione **File (Archivo) > New**

**(Nuevo) > Class (Clase) .**

2. En la ventana **New Java Class** (Nueva clase de Java) escriba **speechsdk.quickstart** en el campo **Package** (Paquete) y **Main** en el campo **Name** (Nombre).



3. Reemplace el código en **Main.java** con el siguiente fragmento de código:

```
package speechsdk.quickstart;

import java.util.concurrent.Future;
import com.microsoft.cognitiveservices.speech.*;

/**
 * Quickstart: recognize speech using the Speech SDK for Java.
 */
public class Main {

    /**
     * @param args Arguments are ignored in this sample.
     */
    public static void main(String[] args) {
        try {
            // Replace below with your own subscription key
            String speechSubscriptionKey = "YourSubscriptionKey";
            // Replace below with your own service region (e.g., "westus").
            String serviceRegion = "YourServiceRegion";

            int exitCode = 1;
            SpeechConfig config = SpeechConfig.fromSubscription(speechSubscriptionKey,
serviceRegion);
            assert(config != null);

            SpeechRecognizer reco = new SpeechRecognizer(config);
            assert(reco != null);

            System.out.println("Say something...");

            Future<SpeechRecognitionResult> task = reco.recognizeOnceAsync();
            assert(task != null);

            SpeechRecognitionResult result = task.get();
            assert(result != null);

            if (result.getReason() == ResultReason.RecognizedSpeech) {
                System.out.println("We recognized: " + result.getText());
                exitCode = 0;
            }
            else if (result.getReason() == ResultReason.NoMatch) {
                System.out.println("NOMATCH: Speech could not be recognized.");
            }
        }
    }
}
```

```
        else if (result.getReason() == ResultReason.Canceled) {
            CancellationDetails cancellation =
CancellationDetails.fromResult(result);
            System.out.println("CANCELED: Reason=" + cancellation.getReason());

            if (cancellation.getReason() == CancellationReason.Error) {
                System.out.println("CANCELED: ErrorCode=" +
cancellation.getErrorCode());
                System.out.println("CANCELED: ErrorDetails=" +
cancellation.getErrorDetails());
                System.out.println("CANCELED: Did you update the subscription
info?");
            }
        }
    }

    reco.close();

    System.exit(exitCode);
} catch (Exception ex) {
    System.out.println("Unexpected exception: " + ex.getMessage());

    assert(false);
    System.exit(1);
}
}
```

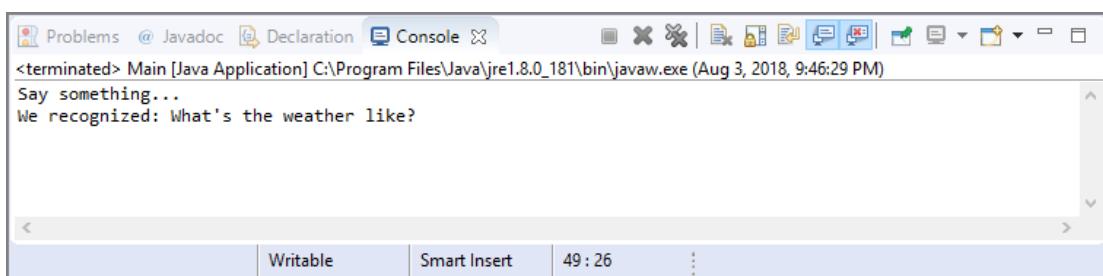
4. Reemplace la cadena `YourSubscriptionKey` por la clave de suscripción.
  5. Reemplace la cadena `YourServiceRegion` por la [región](#) asociada con la suscripción (por ejemplo, `westus` para la suscripción de evaluación gratuita).
  6. Guarde los cambios en el proyecto.

## NOTE

El SDK de Voz se usará de forma predeterminada para reconocer el uso de en-us como idioma. Para más información sobre cómo elegir el idioma de origen, consulte [Especificación del idioma de origen para la conversión de voz a texto](#).

## Compilación y ejecución de la aplicación

Presione F11, o seleccione **Run (Ejecutar) > Debug (Depurar)**. Los próximos 15 segundos de la entrada de voz del micrófono se reconocen y se registran en la ventana de consola.



## Pasos siguientes

Exploración de ejemplos de Java en GitHub

En este inicio rápido usará el [SDK de Voz](#) para reconocer interactivamente la voz de la entrada de micrófono y obtener la transcripción del texto del audio capturado. Es fácil integrar esta

característica en las aplicaciones o dispositivos para tareas de reconocimiento comunes, como transcribir conversaciones. También se puede usar para integraciones más complejas, como el uso de Bot Framework con el SDK de Voz para crear asistentes de voz.

Una vez que se cumplen los requisitos previos, para realizar el reconocimiento de voz a través de un micrófono solo son necesarios cuatro pasos:

- Cree un objeto `SpeechConfig` a partir de la clave y la región de suscripción.
- Cree un objeto `SpeechRecognizer` con el objeto `SpeechConfig` anterior.
- Con el objeto `SpeechRecognizer`, inicie el proceso de reconocimiento de una única expresión.
- Inspeccione el objeto `SpeechRecognitionResult` devuelto.

Si prefiere ponerse a trabajar de inmediato, vea o descargue todos los [ejemplos para Python del SDK de Voz](#) en GitHub. Si no, vamos a comenzar.

## Prerequisites

Antes de comenzar:

- Ha creado un recurso de Voz de Azure
- Ha configurado el entorno de desarrollo
- Ha creado un proyecto de ejemplo vacío
- Asegúrese de que tiene acceso a un micrófono para capturar el audio.

## Soporte técnico y actualizaciones

Las actualizaciones del paquete de Python del SDK de Voz se distribuirán mediante PyPI y se anunciarán en la página [Notas de la versión](#). Si hay disponible una nueva versión, puede actualizarse a ella con el comando `pip install --upgrade azure-cognitiveservices-speech`. Para comprobar qué versión está instalada actualmente, inspeccione la variable `azure.cognitiveservices.speech._version_`.

Si tiene un problema o falta una característica, consulte las [opciones de ayuda y soporte técnico](#).

## Creación de una aplicación de Python mediante el SDK de Voz

### Ejecución del ejemplo

Puede copiar el [código de ejemplo](#) de este inicio rápido en un archivo de código fuente `quickstart.py` y ejecutarlo en el IDE o en la consola:

```
python quickstart.py
```

También, puede descargar este tutorial de inicio rápido como un cuaderno de [Jupyter](#) del [repositorio de ejemplos del SDK de Voz](#) y ejecutarlo como un cuaderno.

### Código de ejemplo

#### NOTE

El SDK de Voz se usará de forma predeterminada para reconocer el uso de en-us como idioma. Para más información sobre cómo elegir el idioma de origen, consulte [Especificación del idioma de origen para la conversión de voz a texto](#).

```

import azure.cognitiveservices.speech as speechsdk

# Creates an instance of a speech config with specified subscription key and service region.
# Replace with your own subscription key and service region (e.g., "westus").
speech_key, service_region = "YourSubscriptionKey", "YourServiceRegion"
speech_config = speechsdk.SpeechConfig(subscription=speech_key, region=service_region)

# Creates a recognizer with the given settings
speech_recognizer = speechsdk.SpeechRecognizer(speech_config=speech_config)

print("Say something...")

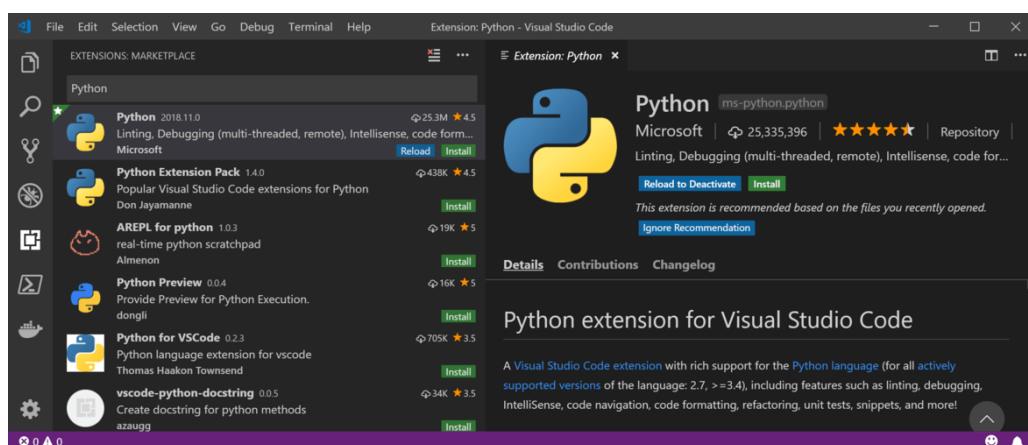
# Starts speech recognition, and returns after a single utterance is recognized. The end of a
# single utterance is determined by listening for silence at the end or until a maximum of 15
# seconds of audio is processed. The task returns the recognition text as result.
# Note: Since recognize_once() returns only a single utterance, it is suitable only for single
# shot recognition like command or query.
# For long-running multi-utterance recognition, use start_continuous_recognition() instead.
result = speech_recognizer.recognize_once()

# Checks result.
if result.reason == speechsdk.ResultReason.RecognizedSpeech:
    print("Recognized: {}".format(result.text))
elif result.reason == speechsdk.ResultReason.NoMatch:
    print("No speech could be recognized: {}".format(result.no_match_details))
elif result.reason == speechsdk.ResultReason.Canceled:
    cancellation_details = result.cancellation_details
    print("Speech Recognition canceled: {}".format(cancellation_details.reason))
    if cancellation_details.reason == speechsdk.CancellationReason.Error:
        print("Error details: {}".format(cancellation_details.error_details))

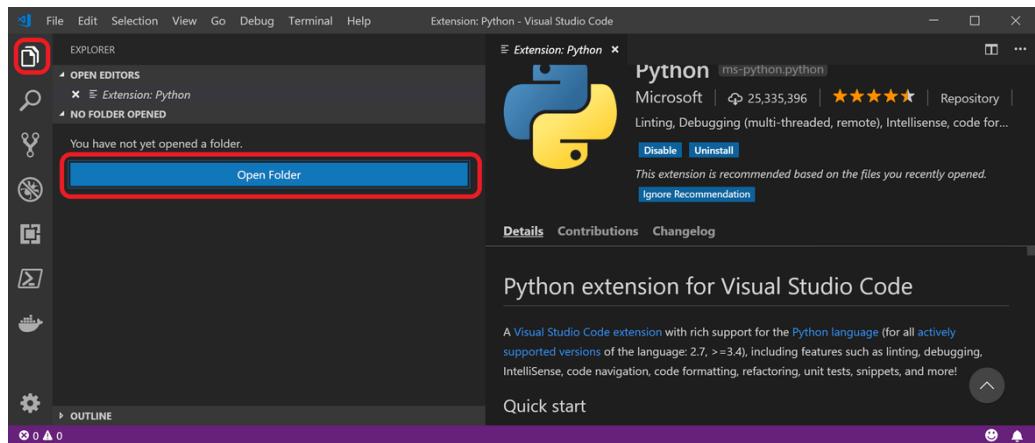
```

## Instalación y uso del SDK de Voz con Visual Studio Code

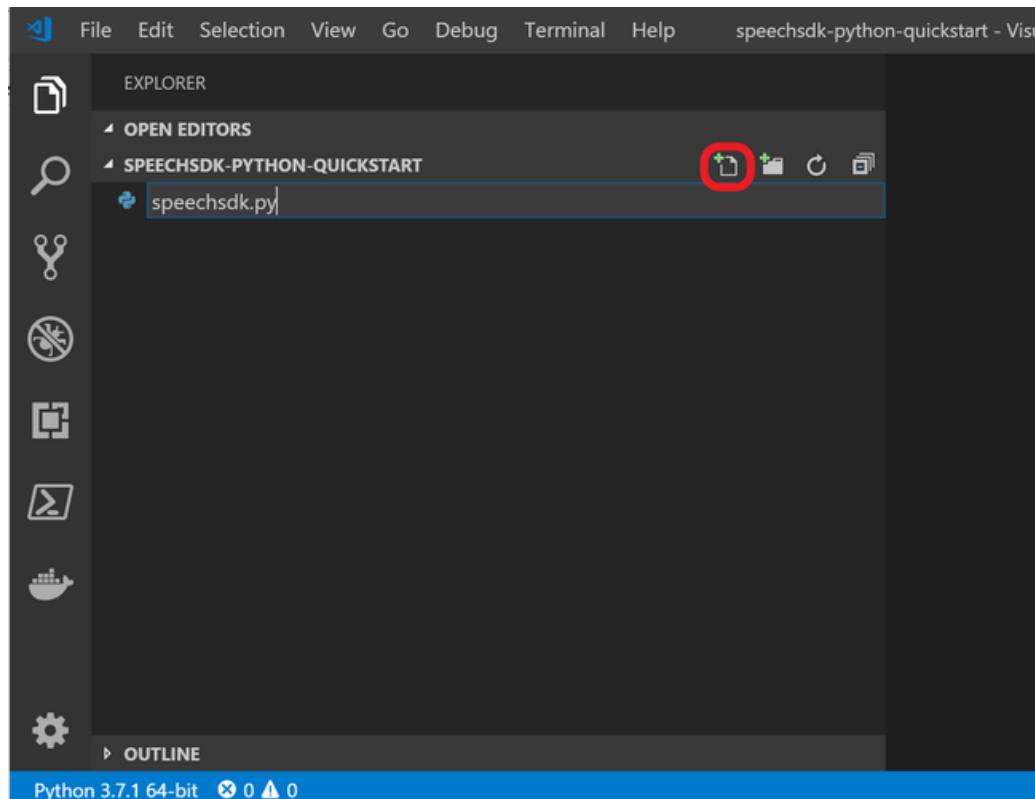
1. Descargue e instale una versión de 64 bits (3.5 o posterior) de [Python](#) en el equipo.
2. Descargue e instale [Visual Studio Code](#).
3. Abra Visual Studio Code e instale la extensión de Python. Seleccione **File > Preferences > Extensions** (Archivo > Preferencias > Extensiones) en el menú. Busque **Python**.



4. Cree una carpeta en la que almacenar el proyecto. Por ejemplo, puede usar para ello el Explorador de Windows.
5. En Visual Studio Code, seleccione el ícono de **File** (Archivo). A continuación, abra la carpeta que creó.

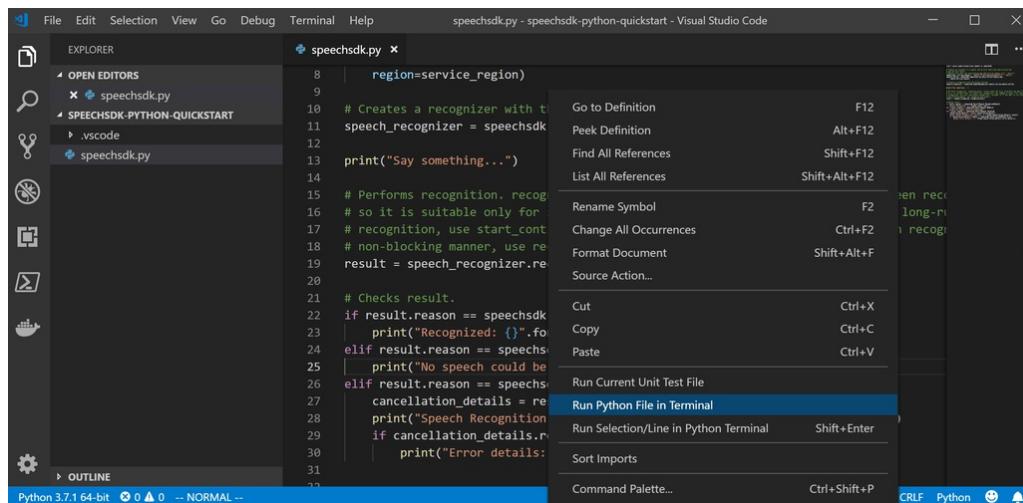


- Cree un archivo de código fuente de Python `speechsdk.py` mediante la selección del ícono de nuevo archivo.



- Copie, pegue y guarde el [código de Python](#) en el archivo recién creado.
- Inserte la información de la suscripción del servicio Voz.
- Si se selecciona, se muestra un intérprete de Python en el lado izquierdo de la barra de estado en la parte inferior de la ventana. En caso contrario, aparecerá una lista de los intérpretes de Python disponibles. Abra la paleta de comandos (Ctrl+Mayús+P) y escriba **Python: Select Interpreter** (Seleccionar intérprete). Elija un valor apropiado.
- Puede instalar el paquete de Python del SDK de Voz desde dentro de Visual Studio Code. Hágalo si no está instalado aún para el intérprete de Python seleccionado. Para instalar el paquete del SDK de Voz, abra un terminal. Abra de nuevo la paleta de comandos (Ctrl+Mayús+P) y escriba **Terminal: Create New Integrated Terminal** (Crear terminal integrado). En el terminal que se abre, escriba el comando  
`python -m pip install azure-cognitiveservices-speech` o el que sea apropiado para su sistema.
- Para ejecutar el código de ejemplo, haga clic con el botón derecho en algún lugar dentro del editor. Seleccione **Run Python File in Terminal** (Ejecutar archivo de Python en terminal).

Diga algunas palabras cuando se le pida. El texto transcrita se muestra poco después.



Si tiene problemas para seguir estas instrucciones, consulte el [tutorial de Python para Visual Studio Code](#) con información más amplia.

## Pasos siguientes

[Exploración de los ejemplos de Python en GitHub](#)

Vea o descargue todos los [ejemplos del SDK de Voz en GitHub](#).

## Compatibilidad con plataformas y lenguajes adicionales

Si ha hecho clic en esta pestaña, es probable que no vea un inicio rápido en su lenguaje de programación favorito. No se preocupe, tenemos materiales de inicio rápido y ejemplos de código adicionales disponibles en GitHub. Use la tabla para encontrar el ejemplo correcto para su lenguaje de programación y combinación de plataforma y sistema operativo.

IDIOMA	INICIOS RÁPIDOS ADICIONALES	EJEMPLOS DE CÓDIGO
C++		<a href="#">Windows, Linux, macOS</a>
C#		<a href="#">.NET Framework, .NET Core, UWP, Unity, Xamarin</a>
Java		<a href="#">Android, JRE</a>
JavaScript		<a href="#">Browser</a>
Node.js		<a href="#">Windows, Linux, macOS</a>
Objective-C	<a href="#">macOs, iOS</a>	<a href="#">iOS, macOS</a>
Python		<a href="#">Windows, Linux, macOS</a>
Swift	<a href="#">macOs, iOS</a>	<a href="#">iOS, macOS</a>

# Inicio rápido: Reconocimiento de voz almacenada en Blob Storage

16/01/2020 • 45 minutes to read • [Edit Online](#)

En este inicio rápido, usará una API REST para reconocer la voz de los archivos en un proceso por lotes. Un proceso por lotes ejecuta la transcripción de voz sin ninguna interacción del usuario. Proporciona un modelo de programación simple, sin necesidad de administrar la simultaneidad, los modelos de voz personalizados u otros detalles. Conlleva opciones de control avanzadas, a la vez que se realiza un uso eficaz de los recursos del servicio de voz de Azure.

La [información general sobre la transcripción por lotes](#) describe los detalles necesarios para usar esta característica. La API detallada está disponible como [documento de Swagger](#) debajo del encabezado `Custom Speech transcriptins`.

El siguiente inicio rápido le guiará a través de un ejemplo de uso.

Si prefiere ponerse a trabajar de inmediato, vea o descargue todos los [ejemplos para C# del SDK de Voz](#) en GitHub. Si no, vamos a comenzar.

## Prerequisites

Antes de comenzar, compruebe lo siguiente:

- [Ha creado un recurso de Voz de Azure](#)
- [Ha cargado de un archivo de origen en un blob de Azure](#)
- [Ha configurado el entorno de desarrollo](#)
- [Ha creado un proyecto de ejemplo vacío](#)

## Abra el proyecto en Visual Studio.

El primer paso es asegurarse de que tiene el proyecto abierto en Visual Studio.

1. Inicie Visual Studio 2019.
2. Cargue el proyecto y abra `Program.cs`.

## Adición de una referencia a Newtonsoft.Json

1. En el Explorador de soluciones, haga clic con el botón derecho en el proyecto **helloworld** y seleccione **Administrar paquetes NuGet** para mostrar el Administrador de paquetes NuGet.
2. En la esquina superior derecha, busque el cuadro desplegable **Origen del paquete** y asegúrese de que `nuget.org` está seleccionado.
3. En la esquina superior izquierda, seleccione **Examinar**.
4. En el cuadro de búsqueda, escriba `newtonsoft.json` y seleccione **Entrar**.
5. En los resultados de la búsqueda, seleccione el paquete **Newtonsoft.Json** y, después, seleccione **Instalar** para instalar la versión estable más reciente.
6. Acepte todos los contratos y licencias para iniciar la instalación.

Después de instalar el paquete aparecerá una confirmación en la ventana **Consola del administrador de paquetes**.

## Inicio con código reutilizable

Vamos a agregar código que funcione como el esqueleto del proyecto.

```
using Newtonsoft.Json;
using System;
using System.Collections.Generic;
using System.IO;
using System.Net;
using System.Threading.Tasks;
using System.Net.Http;
using System.Net.Http.Formatting;

namespace BatchClient
{
    class Program
    {
        // Replace with your subscription key
        private const string SubscriptionKey = "YourSubscriptionKey";

        // Update with your service region
        private const string Region = "YourServiceRegion";
        private const int Port = 443;

        // recordings and locale
        private const string Locale = "en-US";
        private const string RecordingsBlobUri = "YourFileUrl";

        //name and description
        private const string Name = "Simple transcription";
        private const string Description = "Simple transcription description";

        private const string speechToTextBasePath = "api/speechtotext/v2.0/";

        static void Main(string[] args)
        {
            TranscribeAsync().Wait();
        }

        static async Task TranscribeAsync()
        {
            Console.WriteLine("Starting transcriptions client...");
        }
    }
}
```

(Deberá reemplazar los valores de `YourSubscriptionKey`, `YourServiceRegion` y `YourFileUrl` por los suyos propios).

## Contenedores de JSON

Como las API REST toman las solicitudes en formato JSON y también devuelven resultados en formato JSON, se podría interactuar con ellas solo con el uso de cadenas, aunque no se recomienda. Para facilitar la administración de solicitudes y respuestas, se declararán algunas clases para la serialización y deserialización del código JSON.

Continúe y coloque sus declaraciones detrás de `TranscribeAsync`.

```
public sealed class ModelIdentity
{
```

```

private ModelIdentity(Guid id)
{
    this.Id = id;
}

public Guid Id { get; private set; }

public static ModelIdentity Create(Guid Id)
{
    return new ModelIdentity(Id);
}
}

public sealed class Transcription
{
    [JsonConstructor]
    private Transcription(Guid id, string name, string description, string locale, DateTime createdDateTime,
    DateTime lastActionDateTime, string status, Uri recordingsUrl, IReadOnlyDictionary<string, string>
    resultsUrls)
    {
        this.Id = id;
        this.Name = name;
        this.Description = description;
        this.CreatedDateTime = createdDateTime;
        this.LastActionDateTime = lastActionDateTime;
        this.Status = status;
        this.Locale = locale;
        this.RecordingsUrl = recordingsUrl;
        this.ResultsUrls = resultsUrls;
    }

    /// <inheritdoc />
    public string Name { get; set; }

    /// <inheritdoc />
    public string Description { get; set; }

    /// <inheritdoc />
    public string Locale { get; set; }

    /// <inheritdoc />
    public Uri RecordingsUrl { get; set; }

    /// <inheritdoc />
    public IReadOnlyDictionary<string, string> ResultsUrls { get; set; }

    public Guid Id { get; set; }

    /// <inheritdoc />
    public DateTime CreatedDateTime { get; set; }

    /// <inheritdoc />
    public DateTime LastActionDateTime { get; set; }

    /// <inheritdoc />
    public string Status { get; set; }

    public string StatusMessage { get; set; }
}

public sealed class TranscriptionDefinition
{
    private TranscriptionDefinition(string name, string description, string locale, Uri recordingsUrl,
    IEnumerable<ModelIdentity> models)
    {
        this.Name = name;
        this.Description = description;
        this.RecordingsUrl = recordingsUrl;
        this.Locale = locale;
    }
}

```

```

        this.Models = models;
        this.properties = new Dictionary<string, string>();
        this.properties.Add("PunctuationMode", "DictatedAndAutomatic");
        this.properties.Add("ProfanityFilterMode", "Masked");
        this.properties.Add("AddWordLevelTimestamps", "True");
    }

    /// <inheritdoc />
    public string Name { get; set; }

    /// <inheritdoc />
    public string Description { get; set; }

    /// <inheritdoc />
    public Uri RecordingsUrl { get; set; }

    public string Locale { get; set; }

    public IEnumerable<ModelIdentity> Models { get; set; }

    public IDictionary<string, string> properties { get; set; }

    public static TranscriptionDefinition Create(
        string name,
        string description,
        string locale,
        Uri recordingsUrl)
    {
        return TranscriptionDefinition.Create(name, description, locale, recordingsUrl, new
ModelIdentity[0]);
    }

    public static TranscriptionDefinition Create(
        string name,
        string description,
        string locale,
        Uri recordingsUrl,
        IEnumerable<ModelIdentity> models)
    {
        return new TranscriptionDefinition(name, description, locale, recordingsUrl, models);
    }
}

public class AudioFileResult
{
    public string AudioFileName { get; set; }
    public List<SegmentResult> SegmentResults { get; set; }
}

public class RootObject
{
    public List<AudioFileResult> AudioFileResults { get; set; }
}

public class NBest
{
    public double Confidence { get; set; }
    public string Lexical { get; set; }
    public string ITN { get; set; }
    public string MaskedITN { get; set; }
    public string Display { get; set; }
}

public class SegmentResult
{
    public string RecognitionStatus { get; set; }
    public string Offset { get; set; }
    public string Duration { get; set; }
    public List<NBest> NBest { get; set; }
}

```

```
}
```

## Creación y configuración de un cliente HTTP

Lo primero que necesitamos es un cliente HTTP con una dirección URL base y un conjunto de autenticación correctos. Inserte este código en `TranscribeAsync` `[!code-csharp]`

## Generación de una solicitud de transcripción

A continuación, se generará la solicitud de transcripción. Agregue este código a `TranscribeAsync` `[!code-csharp]`

## Envío de la solicitud y comprobación de su estado

Ahora, se va a publicar la solicitud en el servicio de voz y se va a comprobar el código de respuesta inicial. Este código de respuesta indicará simplemente si el servicio ha recibido la solicitud. El servicio devolverá una dirección URL en los encabezados de respuesta, que es la ubicación en la que se almacenará el estado de la transcripción.

```
Uri transcriptionLocation = null;

using (var response = await client.PostAsync(path, sc))
{
    if (!response.IsSuccessStatusCode)
    {
        Console.WriteLine("Error {0} starting transcription.", response.StatusCode);
        return;
    }

    transcriptionLocation = response.Headers.Location;
}
```

## Espere a que finalice la operación.

Dado que el servicio procesa la transcripción de forma asíncrona, es necesario sondear su estado cada cierto tiempo. Aquí se va a comprobar cada cinco segundos.

Para comprobar el estado se puede recuperar el contenido en la dirección URL que obtuvimos al publicar la solicitud. Cuando se recupera el contenido, se deserializa en una de nuestras clases auxiliares para facilitar la interacción con él.

Este es el código de sondeo con la visualización del estado de todas las tareas, excepto el de una finalización correcta, que se hará a continuación.

```

Console.WriteLine($"Created transcription at location {transcriptionLocation}.");

Console.WriteLine("Checking status.");

bool completed = false;
Transcription transcription = null;

// check for the status of our transcriptions periodically
while (!completed)
{
    // get all transcriptions for the user
    using (var response = await
client.GetAsync(transcriptionLocation.AbsolutePath).ConfigureAwait(false))
    {
        var contentType = response.Content.Headers.ContentType;

        if (response.IsSuccessStatusCode && string.Equals(contentType.MediaType, "application/json",
 StringComparison.OrdinalIgnoreCase))
        {
            transcription = await response.Content.ReadAsAsync<Transcription>().ConfigureAwait(false);
        }
        else
        {
            Console.WriteLine("Error with status {0} getting transcription result", response.StatusCode);
            continue;
        }
    }

    // for each transcription in the list we check the status
    switch (transcription.Status)
    {
        case "Failed":
            completed = true;
            Console.WriteLine("Transcription failed. Status: {0}", transcription.StatusMessage);
            break;
        case "Succeeded":
            break;

        case "Running":
            Console.WriteLine("Transcription is still running.");
            break;

        case "NotStarted":
            Console.WriteLine("Transcription has not started.");
            break;
    }

    await Task.Delay(TimeSpan.FromSeconds(5)).ConfigureAwait(false);
}

Console.WriteLine("Press any key...");
Console.ReadKey();
}

```

## Visualización de los resultados de la transcripción

Cuando el servicio haya finalizado correctamente la transcripción, los resultados se almacenarán en otra dirección URL que se puede obtener de la respuesta de estado.

Aquí se realizará una solicitud para descargar esos resultados en un archivo temporal antes de leerlos y deserializarlos. Una vez que se cargan los resultados, se pueden imprimir en la consola.

```
completed = true;

var resultsUri0 = transcription.ResultsUrls["channel_0"];

WebClient webClient = new WebClient();

var filename = Path.GetTempFileName();
webClient.DownloadFile(resultsUri0, filename);
var results0 = File.ReadAllText(filename);
var resultObject0 = JsonConvert.DeserializeObject<RootObject>(results0);
Console.WriteLine(results0);

Console.WriteLine("Transcription succeeded. Results: ");
Console.WriteLine(results0);
```

## Comprobación del código

En este momento, el código debe tener esta apariencia: (Se han agregado algunos comentarios a esta versión)  
[!code-csharp]

## Compilación y ejecución de la aplicación

Ya está listo para compilar la aplicación y probar el reconocimiento de voz con el servicio Voz.

1. **Compile el código:** en la barra de menús de Visual Studio, elija **Compilar > Compilar solución**.
2. **Inicie la aplicación:** en la barra de menús, elija **Depurar > Iniciar depuración** o presione **F5**.
3. **Inicie el reconocimiento:** se le pedirá que diga una frase en inglés. La voz se envía al servicio Voz, se transcribe como texto y se representa en la consola.

## Pasos siguientes

### Exploración de ejemplos de C# en GitHub

En este inicio rápido, usará una API REST para reconocer la voz de los archivos en un proceso por lotes. Un proceso por lotes ejecuta la transcripción de voz sin ninguna interacción del usuario. Proporciona un modelo de programación simple, sin necesidad de administrar la simultaneidad, los modelos de voz personalizados u otros detalles. Conlleva opciones de control avanzadas, a la vez que se realiza un uso eficaz de los recursos del servicio de voz de Azure.

La [información general sobre la transcripción por lotes](#) describe los detalles necesarios para usar esta característica. La API detallada está disponible como [documento de Swagger](#) debajo del encabezado `Custom Speech transcriptins`.

El siguiente inicio rápido le guiará a través de un ejemplo de uso.

Si prefiere ponerse a trabajar de inmediato, vea o descargue todos los [ejemplos para C++ del SDK de Voz](#) en GitHub. Si no, vamos a comenzar.

## Prerequisites

Antes de comenzar, compruebe lo siguiente:

- [Ha creado un recurso de Voz de Azure](#)
- [Ha cargado de un archivo de origen en un blob de Azure](#)
- [Ha configurado el entorno de desarrollo](#)
- [Ha creado un proyecto de ejemplo vacío](#)

## Abra el proyecto en Visual Studio.

El primer paso es asegurarse de que tiene el proyecto abierto en Visual Studio.

1. Inicie Visual Studio 2019.
2. Cargue el proyecto y abra `helloworld.cpp`.

## Adición de referencias

Para acelerar el desarrollo de código, se usarán un par de componentes externos:

- [SDK de REST de CPP](#): una biblioteca cliente para realizar llamadas de REST a un servicio REST.
- [nlohmann/JSON](#): práctica biblioteca de análisis, serialización y deserialización de JSON.

Ambos se pueden instalar mediante [vcpkg](#).

```
vcpkg install cpprestsdk cpprestsdk:x64-windows  
vcpkg install nlohmann-json
```

## Inicio con código reutilizable

Vamos a agregar código que funcione como el esqueleto del proyecto.

```

#include <iostream>
#include <sstream>
#include <Windows.h>
#include <locale>
#include <codecvt>
#include <string>

#include <cpprest/http_client.h>
#include <cpprest/filestream.h>
#include <nlohmann/json.hpp>

using namespace std;
using namespace utility; // Common utilities like string conversions
using namespace web; // Common features like URIs.
using namespace web::http; // Common HTTP functionality
using namespace web::http::client; // HTTP client features
using namespace concurrency::streams; // Asynchronous streams
using json = nlohmann::json;

const string_t region = U("YourServiceRegion");
const string_t subscriptionKey = U("YourSubscriptionKey");
const string name = "Simple transcription";
const string description = "Simple transcription description";
const string myLocale = "en-US";
const string recordingsBlobUri = "YourFileUrl";

void recognizeSpeech()
{
    std::wstring_convert<std::codecvt_utf8_utf16<wchar_t>> converter;
}

int wmain()
{
    recognizeSpeech();
    cout << "Please press a key to continue.\n";
    cin.get();
    return 0;
}

```

(Deberá reemplazar los valores de `YourSubscriptionKey`, `YourServiceRegion` y `YourFileUrl` por los suyos propios).

## Contenedores de JSON

Como las API REST toman las solicitudes en formato JSON y también devuelven resultados en formato JSON, se podría interactuar con ellas solo con el uso de cadenas, aunque no se recomienda. Para facilitar la administración de solicitudes y respuestas, se declararán algunas clases para la serialización y deserialización del código JSON y algunos métodos para ayudar a nlohmann/JSON.

Continúe y coloque sus declaraciones delante de `recognizeSpeech`.

```

class TranscriptionDefinition {
private:
    TranscriptionDefinition(string name,
                           string description,
                           string locale,
                           string recordingsUrl,
                           std::list<string> models) {

        Name = name;
        Description = description;
        RecordingsUrl = recordingsUrl;
    }
}
```

```

        recordingsUrl = recordingsUrl,
        Locale = locale;
        Models = models;
    }

public:
    string Name;
    string Description;
    string RecordingsUrl;
    string Locale;
    std::list<string> Models;
    std::map<string, string> properties;

    static TranscriptionDefinition Create(string name, string description, string locale, string
recordingsUrl) {
        return TranscriptionDefinition(name, description, locale, recordingsUrl, std::list<string>());
    }
    static TranscriptionDefinition Create(string name, string description, string locale, string
recordingsUrl,
        std::list<string> models) {
        return TranscriptionDefinition(name, description, locale, recordingsUrl, models);
    }
};

void to_json(nlohmann::json& j, const TranscriptionDefinition& t) {
    j = nlohmann::json{
        { "description", t.Description },
        { "locale", t.Locale },
        { "models", t.Models },
        { "name", t.Name },
        { "properties", t.properties },
        { "recordingsurl",t.RecordingsUrl }
    };
}

void from_json(const nlohmann::json& j, TranscriptionDefinition& t) {
    j.at("locale").get_to(t.Locale);
    j.at("models").get_to(t.Models);
    j.at("name").get_to(t.Name);
    j.at("properties").get_to(t.properties);
    j.at("recordingsurl").get_to(t.RecordingsUrl);
}

class Transcription {
public:
    string name;
    string description;
    string locale;
    string recordingsUrl;
    map<string, string> resultsUrls;
    string id;
    string createdDateTime;
    string lastActionDateTime;
    string status;
    string statusMessage;
};

void to_json(nlohmann::json& j, const Transcription& t) {
    j = nlohmann::json{
        { "description", t.description },
        { "locale", t.locale },
        { "createddatetime", t.createdDateTime },
        { "name", t.name },
        { "id", t.id },
        { "recordingsurl",t.recordingsUrl },
        { "resultUrls", t.resultsUrls},
        { "status", t.status},
        { "statusmessage", t.statusMessage}
    };
}

```

```

};

void from_json(const nlohmann::json& j, Transcription& t) {
    j.at("description").get_to(t.description);
    j.at("locale").get_to(t.locale);
    j.at("createdDateTime").get_to(t.createdDateTime);
    j.at("name").get_to(t.name);
    j.at("recordingsUrl").get_to(t.recordingsUrl);
    t.resultsUrls = j.at("resultsUrls").get<map<string, string>>();
    j.at("status").get_to(t.status);
    j.at("statusMessage").get_to(t.statusMessage);
}
class Result
{
public:
    string Lexical;
    string ITN;
    string MaskedITN;
    string Display;
};

void from_json(const nlohmann::json& j, Result& r) {
    j.at("Lexical").get_to(r.Lexical);
    j.at("ITN").get_to(r.ITN);
    j.at("MaskedITN").get_to(r.MaskedITN);
    j.at("Display").get_to(r.Display);
}

class NBest : public Result
{
public:
    double Confidence;
};

void from_json(const nlohmann::json& j, NBest& nb) {
    j.at("Confidence").get_to(nb.Confidence);
    j.at("Lexical").get_to(nb.Lexical);
    j.at("ITN").get_to(nb.ITN);
    j.at("MaskedITN").get_to(nb.MaskedITN);
    j.at("Display").get_to(nb.Display);
}

class SegmentResult
{
public:
    string RecognitionStatus;
    ULONG Offset;
    ULONG Duration;
    std::list<NBest> NBest;
};

void from_json(const nlohmann::json& j, SegmentResult& sr) {
    j.at("RecognitionStatus").get_to(sr.RecognitionStatus);
    j.at("Offset").get_to(sr.Offset);
    j.at("Duration").get_to(sr.Duration);
    sr.NBest = j.at("NBest").get<list<NBest>>();
}

class AudioFileResult
{
public:
    string AudioFileName;
    std::list<SegmentResult> SegmentResults;
    std::list<Result> CombinedResults;
};

void from_json(const nlohmann::json& j, AudioFileResult& arf) {
    j.at("AudioFileName").get_to(arf.AudioFileName);
    arf.SegmentResults = j.at("SegmentResults").get<list<SegmentResult>>();
    arf.CombinedResults = j.at("CombinedResults").get<list<Result>>();
}

class RootObject {
    ...
}

```

```
public:  
    std::list<AudioFileResult> AudioFileResults;  
};  
void from_json(const nlohmann::json& j, RootObject& r) {  
    r.AudioFileResults = j.at("AudioFileResults").get<list<AudioFileResult>>();  
}
```

## Creación y configuración de un cliente HTTP

Lo primero que necesitamos es un cliente HTTP con una dirección URL base y un conjunto de autenticación correctos. Inserte este código en `recognizeSpeech`.

```
utility::string_t service_url = U("https://") + region + U(".cris.ai/api/speechtotext/v2.0/Transcriptions/");  
uri u(service_url);  
  
http_client c(u);  
http_request msg(methods::POST);  
msg.headers().add(U("Content-Type"), U("application/json"));  
msg.headers().add(U("Ocp-Apim-Subscription-Key"), subscriptionKey);
```

## Generación de una solicitud de transcripción

A continuación, se generará la solicitud de transcripción. Agregue este código a `recognizeSpeech`.

```
auto transportdef = TranscriptionDefinition::Create(name, description, myLocale, recordingsBlobUri);  
  
nlohmann::json transportdefJSON = transportdef;  
  
msg.set_body(transportdefJSON.dump());
```

## Envío de la solicitud y comprobación de su estado

Ahora, se va a publicar la solicitud en el servicio de voz y se va a comprobar el código de respuesta inicial. Este código de respuesta indicará simplemente si el servicio ha recibido la solicitud. El servicio devolverá una dirección URL en los encabezados de respuesta, que es la ubicación en la que se almacenará el estado de la transcripción.

```
auto response = c.request(msg).get();  
auto statusCode = response.status_code();  
  
if (statusCode != status_codes::Accepted)  
{  
    cout << "Unexpected status code " << statusCode << endl;  
    return;  
}  
  
string_t transcriptionLocation = response.headers()[U("location")];  
  
cout << "Transcription status is located at " << converter.to_bytes(transcriptionLocation) << endl;
```

## Espere a que finalice la operación.

Dado que el servicio procesa la transcripción de forma asíncrona, es necesario sondear su estado cada cierto tiempo. Aquí se va a comprobar cada cinco segundos.

Para comprobar el estado se puede recuperar el contenido en la dirección URL que obtuvimos al publicar la solicitud. Cuando se recupera el contenido, se deserializa en una de nuestras clases auxiliares para facilitar la interacción con él.

Este es el código de sondeo con la visualización del estado de todas las tareas, excepto el de una finalización correcta, que se hará a continuación.

```
bool completed = false;

while (!completed)
{
    auto statusResponse = statusCheckClient.request(statusCheckMessage).get();
    auto statusResponseCode = statusResponse.status_code();

    if (statusResponseCode != status_codes::OK)
    {
        cout << "Fetching the transcription returned unexpected http code " << statusResponseCode << endl;
        return;
    }

    auto body = statusResponse.extract_string().get();
    nlohmann::json statusJSON = nlohmann::json::parse(body);
    Transcription transcriptonStatus = statusJSON;

    if (!_strcmp(transcriptonStatus.status.c_str(), "Failed"))
    {
        completed = true;
        cout << "Transcription has failed " << transcriptonStatus.statusMessage << endl;
    }
    else if (!_strcmp(transcriptonStatus.status.c_str(), "Succeeded"))
    {
    }
    else if (!_strcmp(transcriptonStatus.status.c_str(), "Running"))
    {
        cout << "Transcription is running." << endl;
    }
    else if (!_strcmp(transcriptonStatus.status.c_str(), "NotStarted"))
    {
        cout << "Transcription has not started." << endl;
    }

    if (!completed) {
        Sleep(5000);
    }
}
```

## Visualización de los resultados de la transcripción

Cuando el servicio haya finalizado correctamente la transcripción, los resultados se almacenarán en otra dirección URL que se puede obtener de la respuesta de estado.

Se descargará el contenido de esa dirección URL, se deserializará el código JSON y se recorrerán en bucle los resultados, con la impresión del texto en pantalla a medida que avanzamos.

```

completed = true;
cout << "Success!" << endl;
string result = transcriptionStatus.resultsUrls["channel_0"];
cout << "Transcription has completed. Results are at " << result << endl;
cout << "Fetching results" << endl;

http_client result_client(converter.from_bytes(result));
http_request resultMessage(methods::GET);
resultMessage.headers().add(U("Ocp-Apim-Subscription-Key"), subscriptionKey);

auto resultResponse = result_client.request(resultMessage).get();

auto responseCode = resultResponse.status_code();

if (responseCode != status_codes::OK)
{
    cout << "Fetching the transcription returned unexpected http code " << responseCode << endl;
    return;
}

auto resultBody = resultResponse.extract_string().get();

nlohmann::json resultJSON = nlohmann::json::parse(resultBody);
RootObject root = resultJSON;

for (AudioFileResult af : root.AudioFileResults)
{
    cout << "There were " << af.SegmentResults.size() << " results in " << af.AudioFileName << endl;

    for (SegmentResult segResult : af.SegmentResults)
    {
        cout << "Status: " << segResult.RecognitionStatus << endl;

        if (!_stricmp(segResult.RecognitionStatus.c_str(), "success") && segResult.NBest.size() > 0)
        {
            cout << "Best text result was: '" << segResult.NBest.front().Display << "'" << endl;
        }
    }
}

```

## Comprobación del código

En este momento, el código debe tener esta apariencia: (Se han agregado algunos comentarios a esta versión)

```

#include <iostream>
#include <sstream>
#include <Windows.h>
#include <locale>
#include <codecvt>
#include <string>

#include <cpprest/http_client.h>
#include <cpprest/filestream.h>
#include <nlohmann/json.hpp>

using namespace std; // Common utilities like string conversions
using namespace utility; // Common features like URIs.
using namespace web; // Common HTTP functionality
using namespace web::http; // HTTP client features
using namespace concurrency::streams; // Asynchronous streams
using json = nlohmann::json;

const string_t region = U("YourServiceRegion");
const string_t subscriptionKey = U("YourSubscriptionKey");

```

```

const string name = "Simple transcription";
const string description = "Simple transcription description";
const string myLocale = "en-US";
const string recordingsBlobUri = "YourFileUrl";

class TranscriptionDefinition {
private:
    TranscriptionDefinition(string name,
                           string description,
                           string locale,
                           string recordingsUrl,
                           std::list<string> models) {

        Name = name;
        Description = description;
        RecordingsUrl = recordingsUrl;
        Locale = locale;
        Models = models;
    }

public:
    string Name;
    string Description;
    string RecordingsUrl;
    string Locale;
    std::list<string> Models;
    std::map<string, string> properties;

    static TranscriptionDefinition Create(string name, string description, string locale, string
recordingsUrl) {
        return TranscriptionDefinition(name, description, locale, recordingsUrl, std::list<string>());
    }
    static TranscriptionDefinition Create(string name, string description, string locale, string
recordingsUrl,
                           std::list<string> models) {
        return TranscriptionDefinition(name, description, locale, recordingsUrl, models);
    }
};

void to_json(nlohmann::json& j, const TranscriptionDefinition& t) {
    j = nlohmann::json{
        { "description", t.Description },
        { "locale", t.Locale },
        { "models", t.Models },
        { "name", t.Name },
        { "properties", t.properties },
        { "recordingsurl", t.RecordingsUrl }
    };
}

void from_json(const nlohmann::json& j, TranscriptionDefinition& t) {
    j.at("locale").get_to(t.Locale);
    j.at("models").get_to(t.Models);
    j.at("name").get_to(t.Name);
    j.at("properties").get_to(t.properties);
    j.at("recordingsurl").get_to(t.RecordingsUrl);
}

class Transcription {
public:
    string name;
    string description;
    string locale;
    string recordingsUrl;
    map<string, string> resultsUrls;
    string id;
    string createdDateTime;
    string lastActionDateTime;
    string status;
}

```

```

        string statusMessage;
    };

    void to_json(nlohmann::json& j, const Transcription& t) {
        j = nlohmann::json{
            { "description", t.description },
            { "locale", t.locale },
            { "createddatetime", t.dateTime },
            { "name", t.name },
            { "id", t.id },
            { "recordingsurl", t.recordingsUrl },
            { "resultUrls", t.resultsUrls },
            { "status", t.status },
            { "statusmessage", t.statusMessage }
        };
    };

    void from_json(const nlohmann::json& j, Transcription& t) {
        j.at("description").get_to(t.description);
        j.at("locale").get_to(t.locale);
        j.at("createdDateTime").get_to(t.dateTime);
        j.at("name").get_to(t.name);
        j.at("recordingsUrl").get_to(t.recordingsUrl);
        t.resultsUrls = j.at("resultsUrls").get<map<string, string>>();
        j.at("status").get_to(t.status);
        j.at("statusMessage").get_to(t.statusMessage);
    }

    class Result
    {
    public:
        string Lexical;
        string ITN;
        string MaskedITN;
        string Display;
    };
    void from_json(const nlohmann::json& j, Result& r) {
        j.at("Lexical").get_to(r.Lexical);
        j.at("ITN").get_to(r.ITN);
        j.at("MaskedITN").get_to(r.MaskedITN);
        j.at("Display").get_to(r.Display);
    }

    class NBest : public Result
    {
    public:
        double Confidence;
    };
    void from_json(const nlohmann::json& j, NBest& nb) {
        j.at("Confidence").get_to(nb.Confidence);
        j.at("Lexical").get_to(nb.Lexical);
        j.at("ITN").get_to(nb.ITN);
        j.at("MaskedITN").get_to(nb.MaskedITN);
        j.at("Display").get_to(nb.Display);
    }

    class SegmentResult
    {
    public:
        string RecognitionStatus;
        ULONG Offset;
        ULONG Duration;
        std::list<NBest> NBest;
    };
    void from_json(const nlohmann::json& j, SegmentResult& sr) {
        j.at("RecognitionStatus").get_to(sr.RecognitionStatus);
        j.at("Offset").get_to(sr.Offset);
        j.at("Duration").get_to(sr.Duration);
        sr.NBest = j.at("NBest").get<list<NBest>>();
    }
}

```

```

        }

    class AudioFileResult
    {
public:
    string AudioFileName;
    std::list<SegmentResult> SegmentResults;
    std::list<Result> CombinedResults;
};

void from_json(const nlohmann::json& j, AudioFileResult& arf) {
    j.at("AudioFileName").get_to(arf.AudioFileName);
    arf.SegmentResults = j.at("SegmentResults").get<list<SegmentResult>>();
    arf.CombinedResults = j.at("CombinedResults").get<list<Result>>();
}

class RootObject {
public:
    std::list<AudioFileResult> AudioFileResults;
};

void from_json(const nlohmann::json& j, RootObject& r) {
    r.AudioFileResults = j.at("AudioFileResults").get<list<AudioFileResult>>();
}

void recognizeSpeech()
{
    std::wstring_convert<std::codecvt_utf8_utf16<wchar_t > > converter;

    utility::string_t service_url = U("https://") + region +
U(".cris.ai/api/speechtotext/v2.0/Transcriptions/");
    uri u(service_url);

    http_client c(u);
    http_request msg(methods::POST);
    msg.headers().add(U("Content-Type"), U("application/json"));
    msg.headers().add(U("Ocp-Apim-Subscription-Key"), subscriptionKey);

    auto transportdef = TranscriptionDefinition::Create(name, description, myLocale, recordingsBlobUri);

    nlohmann::json transportdefJSON = transportdef;

    msg.set_body(transportdefJSON.dump());

    auto response = c.request(msg).get();
    auto statusCode = response.status_code();

    if (statusCode != status_codes::Accepted)
    {
        cout << "Unexpected status code " << statusCode << endl;
        return;
    }

    string_t transcriptionLocation = response.headers()[U("location")];

    cout << "Transcription status is located at " << converter.to_bytes(transcriptionLocation) << endl;

    http_client statusCheckClient(transcriptionLocation);
    http_request statusCheckMessage(methods::GET);
    statusCheckMessage.headers().add(U("Ocp-Apim-Subscription-Key"), subscriptionKey);

    bool completed = false;

    while (!completed)
    {
        auto statusResponse = statusCheckClient.request(statusCheckMessage).get();
        auto statusResponseCode = statusResponse.status_code();

        if (statusResponseCode != status_codes::OK)
        {
            cout << "Fetching the transcription returned unsupported http code " << statusResponseCode <<

```

```

cout << " Fetching the transcription returned unexpected http code " << statusResponseCode <<
endl;
return;
}

auto body = statusResponse.extract_string().get();
nlohmann::json statusJSON = nlohmann::json::parse(body);
Transcription transcriptionStatus = statusJSON;

if (!_strcmp(transcriptionStatus.status.c_str(), "Failed"))
{
    completed = true;
    cout << "Transcription has failed " << transcriptionStatus.statusMessage << endl;
}
else if (!_strcmp(transcriptionStatus.status.c_str(), "Succeeded"))
{
    completed = true;
    cout << "Success!" << endl;
    string result = transcriptionStatus.resultsUrls["channel_0"];
    cout << "Transcription has completed. Results are at " << result << endl;
    cout << "Fetching results" << endl;

    http_client result_client(converter.from_bytes(result));
    http_request resultMessage(methods::GET);
    resultMessage.headers().add(U("Ocp-Apim-Subscription-Key"), subscriptionKey);

    auto resultResponse = result_client.request(resultMessage).get();

    auto responseCode = resultResponse.status_code();

    if (responseCode != status_codes::OK)
    {
        cout << "Fetching the transcription returned unexpected http code " << responseCode << endl;
        return;
    }

    auto responseBody = resultResponse.extract_string().get();

    nlohmann::json resultJSON = nlohmann::json::parse(responseBody);
    RootObject root = resultJSON;

    for (AudioFileResult af : root.AudioFileResults)
    {
        cout << "There were " << af.SegmentResults.size() << " results in " << af.AudioFileName <<
endl;

        for (SegmentResult segResult : af.SegmentResults)
        {
            cout << "Status: " << segResult.RecognitionStatus << endl;

            if (!_strcmp(segResult.RecognitionStatus.c_str(), "success") && segResult.NBest.size() >
0)
            {
                cout << "Best text result was: '" << segResult.NBest.front().Display << "'" << endl;
            }
        }
    }
    else if (!_strcmp(transcriptionStatus.status.c_str(), "Running"))
    {
        cout << "Transcription is running." << endl;
    }
    else if (!_strcmp(transcriptionStatus.status.c_str(), "NotStarted"))
    {
        cout << "Transcription has not started." << endl;
    }
}

if (!completed) {
    Sleep(5000);
}

```

```
        }

    }

int wmain()
{
    recognizeSpeech();
    cout << "Please press a key to continue.\n";
    cin.get();
    return 0;
}
```

## Compilación y ejecución de la aplicación

Ya está listo para compilar la aplicación y probar el reconocimiento de voz con el servicio Voz.

## Pasos siguientes

[Exploración de ejemplos de C++ en GitHub](#)

En este inicio rápido, usará una API REST para reconocer la voz de los archivos en un proceso por lotes. Un proceso por lotes ejecuta la transcripción de voz sin ninguna interacción del usuario. Proporciona un modelo de programación simple, sin necesidad de administrar la simultaneidad, los modelos de voz personalizados u otros detalles. Conlleva opciones de control avanzadas, a la vez que se realiza un uso eficaz de los recursos del servicio de voz de Azure.

La [información general sobre la transcripción por lotes](#) describe los detalles necesarios para usar esta característica. La API detallada está disponible como [documento de Swagger](#) debajo del encabezado `Custom Speech transcriptins`.

El siguiente inicio rápido le guiará a través de un ejemplo de uso.

Si prefiere ponerse a trabajar de inmediato, vea o descargue todos los [ejemplos para Java del SDK de Voz](#) en GitHub. Si no, vamos a comenzar.

## Prerequisites

Antes de comenzar, compruebe lo siguiente:

- [Ha creado un recurso de Voz de Azure](#)
- [Ha cargado de un archivo de origen en un blob de Azure](#)
- [Ha configurado el entorno de desarrollo](#)
- [Ha creado un proyecto de ejemplo vacío](#)

## Apertura del proyecto en Eclipse

El primer paso es asegurarse de que tiene el proyecto abierto en Eclipse.

1. Inicie Eclipse
2. Cargue el proyecto y abra `Main.java`.

## Adición de una referencia a Gson

En este inicio rápido, se usará un serializador o deserializador JSON externos. Para Java, hemos elegido [Gson](#).

Abra el archivo pom.xml y agregue la siguiente referencia: [!code-xml]

# Inicio con código reutilizable

Vamos a agregar código que funcione como el esqueleto del proyecto.

```
package quickstart;

import java.io.IOException;
import java.io.InputStreamReader;
import java.io.OutputStream;
import java.net.HttpURLConnection;
import java.net.URL;
import java.util.Date;
import java.util.Dictionary;
import java.util.Hashtable;
import java.util.UUID;

import com.google.gson.Gson;
public class Main {

    private static String region = "YourServiceRegion";
    private static String subscriptionKey = "YourSubscriptionKey";
    private static String Locale = "en-US";
    private static String RecordingsBlobUri = "YourFileUrl";
    private static String Name = "Simple transcription";
    private static String Description = "Simple transcription description";

    public static void main(String[] args) throws IOException, InterruptedException {
        System.out.println("Starting transcriptions client...");
    }
}
```

(Deberá reemplazar los valores de `YourSubscriptionKey`, `YourServiceRegion` y `YourFileUrl` por los suyos propios).

## Contenedores de JSON

Como las API REST toman las solicitudes en formato JSON y también devuelven resultados en formato JSON, se podría interactuar con ellas solo con el uso de cadenas, aunque no se recomienda. Para facilitar la administración de solicitudes y respuestas, se declararán algunas clases para la serialización y deserialización del código JSON.

Continúe y coloque sus declaraciones delante de `Main`.

```
final class Transcription {
    public String name;
    public String description;
    public String locale;
    public URL recordingsUrl;
    public Hashtable<String, String> resultsUrls;
    public UUID id;
    public Date createdDateTime;
    public Date lastActionDateTime;
    public String status;
    public String statusMessage;
}

final class TranscriptionDefinition {
    private TranscriptionDefinition(String name, String description, String locale, URL recordingsUrl,
        ModelIdentity[] models) {
        this.Name = name;
        this.Description = description;
        this.RecordingsUrl = recordingsUrl;
        this.Locale = locale;
    }
}
```

```

        this.Models = models;
        this.properties = new Hashtable<String, String>();
        this.properties.put("PunctuationMode", "DictatedAndAutomatic");
        this.properties.put("ProfanityFilterMode", "Masked");
        this.properties.put("AddWordLevelTimestamps", "True");
    }

    public String Name;
    public String Description;
    public URL RecordingsUrl;
    public String Locale;
    public ModelIdentity[] Models;
    public Dictionary<String, String> properties;

    public static TranscriptionDefinition Create(String name, String description, String locale, URL
recordingsUrl) {
        return TranscriptionDefinition.Create(name, description, locale, recordingsUrl, new
ModelIdentity[0]);
    }

    public static TranscriptionDefinition Create(String name, String description, String locale, URL
recordingsUrl,
        ModelIdentity[] models) {
        return new TranscriptionDefinition(name, description, locale, recordingsUrl, models);
    }
}

final class ModelIdentity {
    private ModelIdentity(UUID id) {
        this.Id = id;
    }

    public UUID Id;

    public static ModelIdentity Create(UUID Id) {
        return new ModelIdentity(Id);
    }
}

class AudioFileResult {
    public String AudioFileName;
    public SegmentResult[] SegmentResults;
}

class RootObject {
    public AudioFileResult[] AudioFileResults;
}

class NBest {
    public double Confidence;
    public String Lexical;
    public String ITN;
    public String MaskedITN;
    public String Display;
}

class SegmentResult {
    public String RecognitionStatus;
    public String Offset;
    public String Duration;
    public NBest[] NBest;
}

```

## Creación y configuración de un cliente HTTP

Lo primero que necesitamos es un cliente HTTP con una dirección URL base y un conjunto de autenticación

correctos. Inserte este código en `Main` [!code-java]

## Generación de una solicitud de transcripción

A continuación, se generará la solicitud de transcripción. Agregue este código a `Main` [!code-java]

## Envío de la solicitud y comprobación de su estado

Ahora, se va a publicar la solicitud en el servicio de voz y se va a comprobar el código de respuesta inicial. Este código de respuesta indicará simplemente si el servicio ha recibido la solicitud. El servicio devolverá una dirección URL en los encabezados de respuesta, que es la ubicación en la que se almacenará el estado de la transcripción.

```
Gson gson = new Gson();

OutputStream stream = postConnection.getOutputStream();
stream.write(gson.toJson(definition).getBytes());
stream.flush();

int statusCode = postConnection.getResponseCode();

if (statusCode != HttpURLConnection.HTTP_ACCEPTED) {
    System.out.println("Unexpected status code " + statusCode);
    return;
}
```

## Espere a que finalice la operación.

Dado que el servicio procesa la transcripción de forma asíncrona, es necesario sondear su estado cada cierto tiempo. Aquí se va a comprobar cada cinco segundos.

Para comprobar el estado se puede recuperar el contenido en la dirección URL que obtuvimos al publicar la solicitud. Cuando se recupera el contenido, se deserializa en una de nuestras clases auxiliares para facilitar la interacción con él.

Este es el código de sondeo con la visualización del estado de todas las tareas, excepto el de una finalización correcta, que se hará a continuación.

```

String transcriptionLocation = postConnection.getHeaderField("location");

System.out.println("Transcription is located at " + transcriptionLocation);

URL transcriptionUrl = new URL(transcriptionLocation);

boolean completed = false;
while (!completed) {
    HttpURLConnection getConnection = (HttpURLConnection) transcriptionUrl.openConnection();
    getConnection.setRequestProperty("Ocp-Apim-Subscription-Key", subscriptionKey);
    getConnection.setRequestMethod("GET");

    int responseCode = getConnection.getResponseCode();

    if (responseCode != HttpURLConnection.HTTP_OK) {
        System.out.println("Fetching the transcription returned unexpected http code " + responseCode);
        return;
    }

    Transcription t = gson.fromJson(new InputStreamReader(getConnection.getInputStream()),
        Transcription.class);

    switch (t.status) {
        case "Failed":
            completed = true;
            System.out.println("Transcription has failed " + t.statusMessage);
            break;
        case "Succeeded":
            break;
        case "Running":
            System.out.println("Transcription is running.");
            break;
        case "NotStarted":
            System.out.println("Transcription has not started.");
            break;
    }

    if (!completed) {
        Thread.sleep(5000);
    }
}

```

## Visualización de los resultados de la transcripción

Cuando el servicio haya finalizado correctamente la transcripción, los resultados se almacenarán en otra dirección URL que se puede obtener de la respuesta de estado.

Se descargará el contenido de esa dirección URL, se deserializará el código JSON y se recorrerán en bucle los resultados, con la impresión del texto en pantalla a medida que avanzamos.

```

package quickstart;

import java.io.IOException;
import java.io.InputStreamReader;
import java.io.OutputStream;
import java.net.HttpURLConnection;
import java.net.URL;
import java.util.Date;
import java.util.Dictionary;
import java.util.Hashtable;
import java.util.UUID;

import com.google.gson.Gson;

```

```

final class Transcription {
    public String name;
    public String description;
    public String locale;
    public URL recordingsUrl;
    public Hashtable<String, String> resultsUrls;
    public UUID id;
    public Date createdDateTime;
    public Date lastActionDateTime;
    public String status;
    public String statusMessage;
}

final class TranscriptionDefinition {
    private TranscriptionDefinition(String name, String description, String locale, URL recordingsUrl,
        ModelIdentity[] models) {
        this.Name = name;
        this.Description = description;
        this.RecordingsUrl = recordingsUrl;
        this.Locale = locale;
        this.Models = models;
        this.properties = new Hashtable<String, String>();
        this.properties.put("PunctuationMode", "DictatedAndAutomatic");
        this.properties.put("ProfanityFilterMode", "Masked");
        this.properties.put("AddWordLevelTimestamps", "True");
    }

    public String Name;
    public String Description;
    public URL RecordingsUrl;
    public String Locale;
    public ModelIdentity[] Models;
    public Dictionary<String, String> properties;

    public static TranscriptionDefinition Create(String name, String description, String locale, URL
recordingsUrl) {
        return TranscriptionDefinition.Create(name, description, locale, recordingsUrl, new
ModelIdentity[0]);
    }

    public static TranscriptionDefinition Create(String name, String description, String locale, URL
recordingsUrl,
        ModelIdentity[] models) {
        return new TranscriptionDefinition(name, description, locale, recordingsUrl, models);
    }
}

final class ModelIdentity {
    private ModelIdentity(UUID id) {
        this.Id = id;
    }

    public UUID Id;

    public static ModelIdentity Create(UUID Id) {
        return new ModelIdentity(Id);
    }
}

class AudioFileResult {
    public String AudioFileName;
    public SegmentResult[] SegmentResults;
}

class RootObject {
    public AudioFileResult[] AudioFileResults;
}

```

```

class NBest {
    public double Confidence;
    public String Lexical;
    public String ITN;
    public String MaskedITN;
    public String Display;
}

class SegmentResult {
    public String RecognitionStatus;
    public String Offset;
    public String Duration;
    public NBest[] NBest;
}

public class Main {

    private static String region = "YourServiceRegion";
    private static String subscriptionKey = "YourSubscriptionKey";
    private static String Locale = "en-US";
    private static String RecordingsBlobUri = "YourFileUrl";
    private static String Name = "Simple transcription";
    private static String Description = "Simple transcription description";

    public static void main(String[] args) throws IOException, InterruptedException {
        System.out.println("Starting transcriptions client...");
        String url = "https://" + region + ".cris.ai/api/speechtotext/v2.0/Transcriptions/";
        URL serviceUrl = new URL(url);

        HttpURLConnection postConnection = (HttpURLConnection) serviceUrl.openConnection();
        postConnection.setDoOutput(true);
        postConnection.setRequestMethod("POST");
        postConnection.setRequestProperty("Content-Type", "application/json");
        postConnection.setRequestProperty("Ocp-Apim-Subscription-Key", subscriptionKey);

        TranscriptionDefinition definition = TranscriptionDefinition.Create(Name, Description, Locale,
            new URL(RecordingsBlobUri));

        Gson gson = new Gson();

        OutputStream stream = postConnection.getOutputStream();
        stream.write(gson.toJson(definition).getBytes());
        stream.flush();

        int statusCode = postConnection.getResponseCode();

        if (statusCode != HttpURLConnection.HTTP_ACCEPTED) {
            System.out.println("Unexpected status code " + statusCode);
            return;
        }

        String transcriptionLocation = postConnection.getHeaderField("location");

        System.out.println("Transcription is located at " + transcriptionLocation);

        URL transcriptionUrl = new URL(transcriptionLocation);

        boolean completed = false;
        while (!completed) {
            HttpURLConnection getConnection = (HttpURLConnection) transcriptionUrl.openConnection();
            getConnection.setRequestProperty("Ocp-Apim-Subscription-Key", subscriptionKey);
            getConnection.setRequestMethod("GET");

            int responseCode = getConnection.getResponseCode();

            if (responseCode != HttpURLConnection.HTTP_OK) {
                System.out.println("Fetching the transcription returned unexpected http code " +
responseCode);
            }
        }
    }
}

```

## Comprobación del código

En este momento, el código debe tener esta apariencia: (Se han agregado algunos comentarios a esta versión)  
[!code-java[](~/samples-cognitive-services-speech-sdk/quickstart/java/jre/from-blob/src/quickstart/Main.java)]

## Compilación y ejecución de la aplicación

Ya está listo para compilar la aplicación y probar el reconocimiento de voz con el servicio Voz.

## Pasos siguientes

### Exploración de ejemplos de Java en GitHub

En este inicio rápido, usará una API REST para reconocer la voz de los archivos en un proceso por lotes. Un proceso por lotes ejecuta la transcripción de voz sin ninguna interacción del usuario. Proporciona un modelo de programación simple, sin necesidad de administrar la simultaneidad, los modelos de voz personalizados u otros detalles. Conlleva opciones de control avanzadas, a la vez que se realiza un uso eficaz de los recursos del servicio de voz de Azure.

La [información general sobre la transcripción por lotes](#) describe los detalles necesarios para usar esta característica. La API detallada está disponible como [documento de Swagger](#) debajo del encabezado `Custom Speech transcriptins`.

El siguiente inicio rápido le guiará a través de un ejemplo de uso.

Si prefiere ponerse a trabajar de inmediato, vea o descargue todos los [ejemplos para Python del SDK de Voz](#) en GitHub. Si no, vamos a comenzar.

## Prerequisites

Antes de comenzar, compruebe lo siguiente:

- [Ha creado un recurso de Voz de Azure](#)
- [Ha cargado de un archivo de origen en un blob de Azure](#)
- [Ha configurado el entorno de desarrollo](#)
- [Ha creado un proyecto de ejemplo vacío](#)

## Descarga e instalación de la biblioteca cliente de API

Para ejecutar el ejemplo, debe generar la biblioteca de Python para la API REST que se genera mediante [Swagger](#).

Siga estos pasos para la instalación:

1. Ir a <https://editor.swagger.io>.
2. Haga clic en **File** (Archivo) y, luego, en **Import URL** (URL de importación).
3. Escriba la dirección URL de Swagger, incluida la región de la suscripción del servicio de voz:  
`https://<your-region>.cris.ai/docs/v2.0/swagger`.
4. Haga clic en **Generate Client** (Generar cliente) y seleccione **Python**.
5. Guarde la biblioteca cliente.
6. Extraiga el archivo python-client-generated.zip descargado en alguna parte del sistema de archivos.
7. Instale el módulo python-client extraído en el entorno de Python mediante pip:  
`pip install path/to/package/python-client`.
8. El paquete instalado tiene el nombre `swagger_client`. Puede comprobar que la instalación ha funcionado con el comando `python -c "import swagger_client"`.

**Nota:** Debido a un [error conocido en la generación automática de Swagger](#), pueden producirse errores al

importar el paquete `swagger_client`. Para corregirlos, elimine la línea con el contenido

```
from swagger_client.models.model import Model # noqa: F401,E501
```

del archivo `swagger_client/models/model.py` y la línea con el contenido

```
from swagger_client.models.inner_error import InnerError # noqa: F401,E501
```

del archivo `swagger_client/models/inner_error.py` dentro del paquete instalado. El mensaje de error le indicará dónde se encuentran estos archivos para su instalación.

## Instalación de otras dependencias

En el ejemplo se usa la biblioteca `requests`. Puede instalarla con el comando

```
pip install requests
```

## Inicio con código reutilizable

Vamos a agregar código que funcione como el esqueleto del proyecto.

```
#!/usr/bin/env python
# coding: utf-8
from typing import List

import logging
import sys
import requests
import time
import swagger_client as cris_client

logging.basicConfig(stream=sys.stdout, level=logging.DEBUG, format"%(message)s")

# Your subscription key and region for the speech service
SUBSCRIPTION_KEY = "YourSubscriptionKey"
SERVICE_REGION = "YourServiceRegion"

NAME = "Simple transcription"
DESCRIPTION = "Simple transcription description"

LOCALE = "en-US"
RECORDINGS_BLOB_URI = "<Your SAS Uri to the recording>"

# Set subscription information when doing transcription with custom models
ADAPTED_ACOUSTIC_ID = None # guid of a custom acoustic model
ADAPTED_LANGUAGE_ID = None # guid of a custom language model

def transcribe():
    logging.info("Starting transcription client...")

if __name__ == "__main__":
    transcribe()
```

(Deberá reemplazar los valores de `YourSubscriptionKey`, `YourServiceRegion` y `YourFileUrl` por los tuyos

propios).

## Creación y configuración de un cliente HTTP

Lo primero que necesitamos es un cliente HTTP con una dirección URL base y un conjunto de autenticación correctos. Inserte este código en `transcribe` [!code-python]

## Generación de una solicitud de transcripción

A continuación, se generará la solicitud de transcripción. Agregue este código a `transcribe` [!code-python]

## Envío de la solicitud y comprobación de su estado

Ahora, se va a publicar la solicitud en el servicio de voz y se va a comprobar el código de respuesta inicial. Este código de respuesta indicará simplemente si el servicio ha recibido la solicitud. El servicio devolverá una dirección URL en los encabezados de respuesta, que es la ubicación en la que se almacenará el estado de la transcripción.

```
data, status, headers = transcription_api.create_transcription_with_http_info(transcription_definition)

# extract transcription location from the headers
transcription_location: str = headers["location"]

# get the transcription Id from the location URI
created_transcription: str = transcription_location.split('/')[-1]

logging.info("Created new transcription with id {}".format(created_transcription))
```

## Espere a que finalice la operación.

Dado que el servicio procesa la transcripción de forma asíncrona, es necesario sondear su estado cada cierto tiempo. Aquí se va a comprobar cada cinco segundos.

Se enumerarán todas las transcripciones que está procesando este recurso del servicio de voz y se buscará la que se ha creado.

Este es el código de sondeo con la visualización del estado de todas las tareas, excepto el de una finalización correcta, que se hará a continuación.

```

logging.info("Checking status.")

completed = False

while not completed:
    running, not_started = 0, 0

    # get all transcriptions for the user
    transcriptions: List[cris_client.Transcription] = transcription_api.get_transcriptions()

    # for each transcription in the list we check the status
    for transcription in transcriptions:
        if transcription.status in ("Failed", "Succeeded"):
            # we check to see if it was the transcription we created from this client
            if created_transcription != transcription.id:
                continue

        completed = True

        if transcription.status == "Succeeded":
            else:
                logging.info("Transcription failed :{}.".format(transcription.status_message))
                break
        elif transcription.status == "Running":
            running += 1
        elif transcription.status == "NotStarted":
            not_started += 1

    logging.info("Transcriptions status: "
                "completed (this transcription): {}, {} running, {} not started yet".format(
                    completed, running, not_started))

    # wait for 5 seconds
    time.sleep(5)

```

## Visualización de los resultados de la transcripción

Cuando el servicio haya finalizado correctamente la transcripción, los resultados se almacenarán en otra dirección URL que se puede obtener de la respuesta de estado.

Aquí obtendremos ese archivo JSON de resultados y lo mostraremos.

```

results_uri = transcription.results_urls["channel_0"]
results = requests.get(results_uri)
logging.info("Transcription succeeded. Results: ")
logging.info(results.content.decode("utf-8"))

```

## Comprobación del código

En este momento, el código debe tener esta apariencia: (Se han agregado algunos comentarios a esta versión)  
[!code-python]

## Compilación y ejecución de la aplicación

Ya está listo para compilar la aplicación y probar el reconocimiento de voz con el servicio Voz.

## Pasos siguientes

[Exploración de los ejemplos de Python en GitHub](#)

Vea o descargue todos los [ejemplos del SDK de Voz](#) en GitHub.

## Compatibilidad con plataformas y lenguajes adicionales

Si ha hecho clic en esta pestaña, es probable que no vea un inicio rápido en su lenguaje de programación favorito. No se preocupe, tenemos materiales de inicio rápido y ejemplos de código adicionales disponibles en GitHub. Use la tabla para encontrar el ejemplo correcto para su lenguaje de programación y combinación de plataforma y sistema operativo.

IDIOMA	INICIOS RÁPIDOS ADICIONALES	EJEMPLOS DE CÓDIGO
C++		<a href="#">Inicios rápidos</a> , <a href="#">Ejemplos</a>
C#		<a href="#">.NET Framework</a> , <a href="#">.NET Core</a> , <a href="#">UWP</a> , <a href="#">Unity</a> , <a href="#">Xamarin</a>
Java		<a href="#">Android</a> , <a href="#">JRE</a>
JavaScript		<a href="#">Browser</a>
Node.js		<a href="#">Windows</a> , <a href="#">Linux</a> , <a href="#">macOS</a>
Objective-C	<a href="#">macOS</a> , <a href="#">iOS</a>	<a href="#">iOS</a> , <a href="#">macOS</a>
Python		<a href="#">Windows</a> , <a href="#">Linux</a> , <a href="#">macOS</a>
Swift	<a href="#">macOS</a> , <a href="#">iOS</a>	<a href="#">iOS</a> , <a href="#">macOS</a>

# Especificación del lenguaje fuente para la conversión de voz en texto

13/01/2020 • 5 minutes to read • [Edit Online](#)

En este artículo, obtendrá información sobre cómo especificar el lenguaje fuente de una entrada de audio que se pasa al SDK de Voz para el reconocimiento de voz. Además, se proporciona código de ejemplo para especificar un modelo de voz personalizado para el reconocimiento mejorado.

## Cómo especificar el lenguaje fuente en C#

El primer paso consiste en crear un `SpeechConfig`:

```
var speechConfig = SpeechConfig.FromSubscription("YourSubscriptionKey", "YourServiceRegion");
```

A continuación, especifique el lenguaje fuente del audio con `SpeechRecognitionLanguage`:

```
speechConfig.SpeechRecognitionLanguage = "de-DE";
```

Si utiliza un modelo personalizado para el reconocimiento, puede especificar el punto de conexión con `EndpointId`:

```
speechConfig.EndpointId = "The Endpoint ID for your custom model.;"
```

## Cómo especificar el lenguaje fuente en C++

En este ejemplo, el lenguaje de origen se proporciona explícitamente como un parámetro mediante el método `FromConfig`.

```
auto recognizer = SpeechRecognizer::FromConfig(speechConfig, "de-DE", audioConfig);
```

En este ejemplo, el lenguaje de origen se proporciona mediante `SourceLanguageConfig`. A continuación, se pasa el método `sourceLanguageConfig` como un parámetro a `FromConfig` al crear el elemento `recognizer`.

```
auto sourceLanguageConfig = SourceLanguageConfig::FromLanguage("de-DE");
auto recognizer = SpeechRecognizer::FromConfig(speechConfig, sourceLanguageConfig, audioConfig);
```

En este ejemplo, el lenguaje de origen y el punto de conexión personalizado se proporcionan mediante `SourceLanguageConfig`. Se pasa el método `sourceLanguageConfig` como un parámetro a `FromConfig` al crear el elemento `recognizer`.

```
auto sourceLanguageConfig = SourceLanguageConfig::FromLanguage("de-DE", "The Endpoint ID for your
custom model.");
auto recognizer = SpeechRecognizer::FromConfig(speechConfig, sourceLanguageConfig, audioConfig);
```

**NOTE**

`SetSpeechRecognitionLanguage` y `SetEndpointId` son métodos en desuso de la clase `SpeechConfig` en C++ y Java. No se recomienda el uso de estos métodos y no deben usarse al construir un `SpeechRecognizer`.

## Cómo especificar el lenguaje fuente en Java

En este ejemplo, el lenguaje de origen se proporciona explícitamente cuando se crea un nuevo elemento `SpeechRecognizer`.

```
SpeechRecognizer recognizer = new SpeechRecognizer(speechConfig, "de-DE", audioConfig);
```

En este ejemplo, el lenguaje de origen se proporciona mediante `SourceLanguageConfig`. A continuación, se pasa el método `sourceLanguageConfig` como parámetro al crear el elemento `SpeechRecognizer`.

```
SourceLanguageConfig sourceLanguageConfig = SourceLanguageConfig.fromLanguage("de-DE");
SpeechRecognizer recognizer = new SpeechRecognizer(speechConfig, sourceLanguageConfig, audioConfig);
```

En este ejemplo, el lenguaje de origen y el punto de conexión personalizado se proporcionan mediante `SourceLanguageConfig`. A continuación, se pasa el método `sourceLanguageConfig` como parámetro al crear el elemento `SpeechRecognizer`.

```
SourceLanguageConfig sourceLanguageConfig = SourceLanguageConfig.fromLanguage("de-DE", "The Endpoint ID
for your custom model.");
SpeechRecognizer recognizer = new SpeechRecognizer(speechConfig, sourceLanguageConfig, audioConfig);
```

**NOTE**

`setSpeechRecognitionLanguage` y `setEndpointId` son métodos en desuso de la clase `SpeechConfig` en C++ y Java. No se recomienda el uso de estos métodos y no deben usarse al construir un `SpeechRecognizer`.

## Cómo especificar el lenguaje fuente en Python

El primer paso consiste en crear un `speech_config`:

```
speech_key, service_region = "YourSubscriptionKey", "YourServiceRegion"
speech_config = speechsdk.SpeechConfig(subscription=speech_key, region=service_region)
```

A continuación, especifique el lenguaje fuente del audio con `speech_recognition_language`:

```
speech_config.speech_recognition_language="de-DE"
```

Si utiliza un modelo personalizado para el reconocimiento, puede especificar el punto de conexión con `endpoint_id`:

```
speech_config.endpoint_id = "The Endpoint ID for your custom model."
```

## Cómo especificar el lenguaje fuente en JavaScript

El primer paso consiste en crear un `SpeechConfig` :

```
var speechConfig = sdk.SpeechConfig.fromSubscription("YourSubscriptionkey", "YourRegion");
```

A continuación, especifique el lenguaje fuente del audio con `speechRecognitionLanguage` :

```
speechConfig.speechRecognitionLanguage = "de-DE";
```

Si utiliza un modelo personalizado para el reconocimiento, puede especificar el punto de conexión con `endpointId` :

```
speechConfig.endpointId = "The Endpoint ID for your custom model.;"
```

## Cómo especificar el lenguaje fuente en Objective-C

El primer paso consiste en crear un `speechConfig` :

```
SPXSpeechConfiguration *speechConfig = [[SPXSpeechConfiguration alloc]  
initWithSubscription:@"YourSubscriptionkey" region:@"YourRegion"];
```

A continuación, especifique el lenguaje fuente del audio con `speechRecognitionLanguage` :

```
speechConfig.speechRecognitionLanguage = @"/de-DE";
```

Si utiliza un modelo personalizado para el reconocimiento, puede especificar el punto de conexión con `endpointId` :

```
speechConfig.endpointId = @"/The Endpoint ID for your custom model."/;
```

## Otras referencias

- Para obtener una lista de idiomas y configuraciones regionales admitidos para la conversión de voz en texto, consulte [compatibilidad de idioma](#).

## Pasos siguientes

- [Documentación de referencia del SDK de voz](#)

# ¿Qué es Custom Speech?

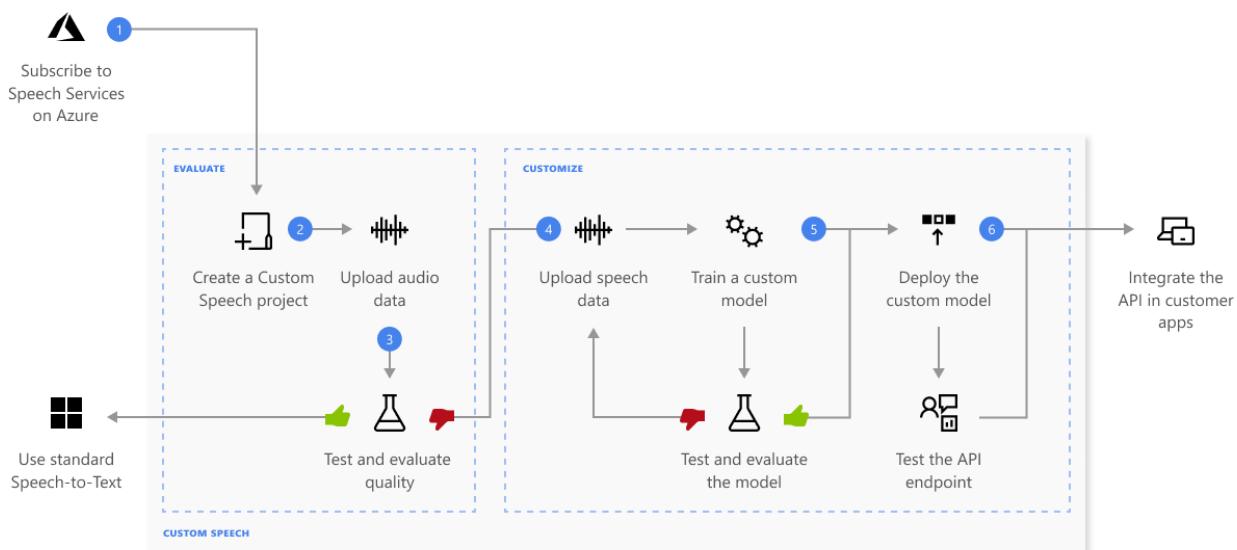
13/01/2020 • 6 minutes to read • [Edit Online](#)

Custom Speech es un conjunto de herramientas en línea que permiten evaluar y mejorar la precisión de la conversión de voz a texto de Microsoft para las aplicaciones, herramientas y productos. Para comenzar solo se necesita una serie de archivos de audio de prueba. Siga los vínculos que se incluyen a continuación para empezar a crear una experiencia personalizada de conversión de voz a texto.

## ¿Qué incluye Custom Speech?

Para utilizar Custom Speech, necesitará una cuenta de Azure y una suscripción al servicio de voz. Cuando ya tenga la cuenta, podrá preparar los datos, entrenar y probar sus modelos, inspeccionar la calidad del reconocimiento, evaluar la precisión y, en última instancia, implementar y utilizar el modelo de conversión de voz a texto personalizado.

Este diagrama resalta las partes que componen el [portal de Custom Speech](#). Use los siguientes vínculos para obtener más información sobre cada paso.



1. **Suscríbese y cree un proyecto:** cree una cuenta de Azure y suscríbase al servicio de voz. Esta suscripción unificada proporciona acceso a la conversión de voz a texto, la conversión de texto a voz, la traducción de voz y el [portal de Custom Speech](#). A continuación, mediante la suscripción al servicio de voz, cree su primer proyecto de Custom Speech.
2. **Cargue datos de prueba:** cargue datos de prueba (archivos de audio) para evaluar la oferta de conversión de voz a texto para sus aplicaciones, herramientas y productos.
3. **Inspeccione la calidad del reconocimiento:** use el [portal de Custom Speech](#) para reproducir el audio cargado e inspeccionar la calidad del reconocimiento de voz de los datos de prueba. Para conocer las medidas cuantitativas, consulte [Inspección de los datos](#).
4. **Evalúe la precisión:** evalúe la precisión del modelo de conversión de voz a texto. El [portal de Custom Speech](#) proporcionará una *tasa de errores de palabras*, que puede utilizarse para determinar si se necesita más entrenamiento. Si está satisfecho con la precisión, puede usar directamente las API del servicio de voz. Si desea mejorar la precisión en una media relativa del 5 al 20 %, use la pestaña **Entrenamiento** del portal para cargar datos de entrenamiento adicionales, como transcripciones con etiqueta humana y texto

relacionado.

5. **Entrene el modelo:** mejore la precisión del modelo de conversión de voz a texto uniendo transcripciones escritas (de 10 a 1000 horas) y texto relacionado (< 200 MB) a datos de prueba de audio. Estos datos ayudan a entrenar el modelo de conversión de voz a texto. Después del entrenamiento, vuelva a probar; si le satisface el resultado, ya puede implementar el modelo.
6. **Implemente el modelo:** cree un punto de conexión personalizado para el modelo de conversión de voz a texto y úselo en sus aplicaciones, herramientas o productos.

## Configuración de la cuenta de Azure

Para poder usar el [portal de Custom Speech](#) y crear un modelo personalizado, se necesita una suscripción al servicio de voz. Siga estas instrucciones para crear una suscripción estándar al servicio de voz: [Creación de una suscripción a Voz](#).

### NOTE

Asegúrese de crear suscripciones estándar (S0); no se admiten las suscripciones de prueba gratuita (F0).

Después de crear la cuenta de Azure y la suscripción al servicio de voz, deberá iniciar sesión en el [portal de Custom Speech](#) y conectarse a su suscripción.

1. Obtenga la clave de la suscripción del servicio de voz en Azure Portal.
2. Inicie sesión en el [portal de Custom Speech](#).
3. Seleccione la suscripción que necesita para trabajar y crear un proyecto de voz.
4. Si desea modificar la suscripción, utilice el ícono **engranaje** situado en el panel de navegación superior.

## Creación de un proyecto

El contenido, como datos, modelos, pruebas y puntos de conexión, se organiza en **proyectos** en el [portal de Custom Speech](#). Cada proyecto es específico de un dominio y un país o idioma. Por ejemplo, puede crear un proyecto para centros de llamadas que usan el inglés en Estados Unidos.

Para crear su primer proyecto, seleccione **Speech-to-text/Custom speech** (Conversión de voz a texto/Conversión de voz personalizada) y, a continuación, haga clic en **New project** (Nuevo proyecto). Siga las instrucciones del asistente para crear el proyecto. Después de crear el proyecto, verá cuatro pestañas: **Datos, Pruebas, Entrenamiento e Implementación**. Use los vínculos incluidos en [Pasos siguientes](#) para aprender a usar cada pestaña.

## Pasos siguientes

- [Preparación y prueba de los datos](#)
- [Inspección de los datos](#)
- [Evaluación de los datos](#)
- [Entrenamiento del modelo](#)
- [Implementación del modelo](#)

# Listas de frases para reconocimiento de voz a texto

13/01/2020 • 2 minutes to read • [Edit Online](#)

Al proporcionar al servicio de voz una lista de frases, puede mejorar la precisión del reconocimiento de voz. Las listas de frases se usan para identificar frases conocidas en datos de audio, como el nombre de una persona o una ubicación específica.

Por ejemplo, si tiene el comando "Mover a" y "Cerca" como posible destino que se puede decir, puede añadir la entrada "Mover a Cerca". Al agregar una frase, aumentará la probabilidad de que, cuando se reconozca el audio, se reconozca "Mover a Cerca" en lugar de "Mover acerca".

A una lista de frases se pueden agregar palabras solas o frases completas. Durante el reconocimiento, se usa una entrada de una lista de frases si el audio incluye una coincidencia exacta de toda una frase como frase independiente. Si no se encuentra una coincidencia exacta con la frase, el reconocimiento no está asistido.

## NOTE

Actualmente, las listas de frases solo admiten el inglés para la conversión de voz en texto.

## Cómo usar las listas de frases

Los ejemplos siguientes muestran cómo crear una lista de frases mediante el objeto `PhraseListGrammar`.

```
PhraseListGrammar phraseList = PhraseListGrammar.FromRecognizer(recognizer);
phraseList.AddPhrase("Move to Ward");
phraseList.AddPhrase("Move to Bill");
phraseList.AddPhrase("Move to Ted");
```

```
auto phraselist = PhraseListGrammar::FromRecognizer(recognizer);
phraselist->AddPhrase("Move to Ward");
phraselist->AddPhrase("Move to Bill");
phraselist->AddPhrase("Move to Ted");
```

```
PhraseListGrammar phraseListGrammar = PhraseListGrammar.fromRecognizer(recognizer);
phraseListGrammar.addPhrase("Move to Ward");
phraseListGrammar.addPhrase("Move to Bill");
phraseListGrammar.addPhrase("Move to Ted");
```

```
phrase_list_grammar = speechsdk.PhraseListGrammar.from_recognizer(reco)
phrase_list_grammar.addPhrase("Move to Ward")
phrase_list_grammar.addPhrase("Move to Bill")
phrase_list_grammar.addPhrase("Move to Ted")
```

```
var phraseListGrammar = SpeechSDK.PhraseListGrammar.fromRecognizer(reco);
phraseListGrammar.addPhrase("Move to Ward");
phraseListGrammar.addPhrase("Move to Bill");
phraseListGrammar.addPhrase("Move to Ted");
```

**NOTE**

El número máximo de listas de frases que usará el servicio de voz para establecer coincidencias con la voz es 1024 frases.

También puede borrar las frases asociadas con `PhraseListGrammar` llamando a `clear()`.

```
phraseList.Clear();
```

```
phraselist->Clear();
```

```
phraseListGrammar.clear();
```

```
phrase_list_grammar.clear()
```

```
phraseListGrammar.clear();
```

**NOTE**

Los cambios realizados en el objeto `PhraseListGrammar` se aplican en el siguiente reconocimiento o tras una reconexión con el servicio de voz.

## Pasos siguientes

- [Documentación de referencia del SDK de voz](#)

# Tutorial: Creación de un modelo de inquilino (versión preliminar)

14/01/2020 • 11 minutes to read • [Edit Online](#)

Tenant Model (Custom Speech con datos de Office 365) es un servicio de participación para clientes empresariales de Office 365 que genera automáticamente un modelo de reconocimiento de voz personalizado a partir de los datos de Office 365 de su organización. El modelo está optimizado para términos técnicos, jerga y nombres de personas, y todo ello de forma segura y compatible.

## IMPORTANT

Si su organización se inscribe mediante el servicio Tenant Model, el servicio de voz puede acceder al modelo de lenguaje de su organización. El modelo se genera a partir de documentos y correos electrónicos de grupos públicos de Office 365, que puede ver cualquier usuario de la organización. El administrador de Office 365 de la organización puede activar o desactivar el uso del modelo de lenguaje en toda la organización desde el portal de administración de Office 365.

En este tutorial, aprenderá a:

- Inscribirse en Tenant Model desde el Centro de administración de Microsoft 365
- Obtener una clave de suscripción del servicio Voz
- Crear un modelo de inquilino
- Implementar un modelo de inquilino
- Usar un modelo de inquilino con el SDK de Voz

## Inscripción en el servicio Tenant Model

Para poder implementar un modelo de inquilino, es preciso estar inscrito en el servicio Tenant Model. La inscripción se completa en el Centro de administración de Microsoft 365 y solo puede realizarla un administrador de Microsoft 365.

1. Inicie sesión en el [Centro de administración de Microsoft 365](#).
2. En el panel izquierdo, seleccione **Configuración, Aplicaciones y Servicios de voz de Azure**.

The screenshot shows the Microsoft 365 Admin Center interface. On the left, there's a navigation sidebar with options like Home, Users, Groups, Roles, Billing, Support, Settings (which is highlighted with a red box), Microsoft Search, Apps (also highlighted with a red box), Security & privacy, Organization profile, and Partner relationships. The main content area is titled 'Contoso' and shows the 'Services & add-ins' section. It lists various services with their icons and descriptions. The 'Azure Speech Services' service is highlighted with a red box. The table columns are Name ↑, Description, and Host Apps. Other listed services include Cortana, Directory Synchronization, Dynamics 365 AI for Sales - Analytics, Dynamics 365 AI for Sales - Connection Graph, Integrated apps, and Microsoft Graph data connect.

Name ↑	Description	Host Apps
Azure multi-factor authentication	Manage multi-factor authentication settings for your users.	
Azure Speech Services	Allow use of your organization's emails and documents to improve speech recognition accuracy.	
Cortana	Turn Cortana access on or off for your entire organization.	
Directory Synchronization	Sync users to the cloud using Azure Active Directory.	
Dynamics 365 AI for Sales - Analytics	Allow Dynamics 365 to generate insights based on user data.	
Dynamics 365 AI for Sales - Connection Graph	Manage and update your Dynamics 365 AI for Sales - Connection Graph settings.	
Integrated apps	Let users decide whether third-party apps can access their Office 365 info.	
Microsoft Graph data connect	Manage and update your Microsoft Graph data connect settings.	

3. Seleccione la casilla **Permitir el modelo de idioma de toda la organización** y, después, **Guardar**

cambios.

The screenshot shows the Microsoft 365 admin center interface. On the left, there's a navigation sidebar with options like Home, Users, Groups, Roles, Billing, Support, Settings, Microsoft Search, Apps, Security & privacy, and Organization profile. The main content area is titled 'Services & add-ins' and lists several services: Azure multi-factor authentication, Azure Speech Services, Cortana, Directory Synchronization, Dynamics 365 AI for Sales - Analytics, Dynamics 365 AI for Sales - Connection Graph, and Integrated apps. The 'Azure Speech Services' row is highlighted. To its right, there's a detailed description of what the service does and a configuration section. In this section, there's a checkbox labeled 'Allow the organization-wide language model' which is checked, and a blue-bordered 'Save changes' button at the bottom.

Para desactivar la instancia del modelo de inquilino:

1. Repita los pasos 1 y 2 anteriores.
2. Desactive la casilla **Permitir el modelo de idioma de toda la organización** y seleccione **Guardar cambios**.

## Obtener una clave de suscripción del servicio Voz

Para usar un modelo de inquilino con el SDK de Voz, necesita un recurso de Voz y su clave de suscripción asociada.

1. Inicie sesión en [Azure Portal](#).
2. Seleccione **Crear un recurso**.
3. En el cuadro **Buscar en Marketplace**, escriba **Speech**.
4. En la lista de resultados, seleccione **Speech** y, después, **Crear**.
5. Siga las instrucciones en pantalla para crear el recurso. Asegúrese de lo siguiente:
  - La **ubicación** se ha establecido en **eastus** o **westus**.
  - El **plan de tarifa** está establecido en **S0**.
6. Seleccione **Crear**.

Después de unos minutos, se crea el recurso. La clave de suscripción está disponible en la sección **Información general** del recurso.

## Creación de un modelo de lenguaje

Una vez que el administrador haya habilitado Tenant Model para su organización, puede crear un modelo de lenguaje basado en los datos de Office 365.

1. Inicie sesión en [Speech Studio](#).
2. En la parte superior derecha, seleccione **Settings** (Configuración) (ícono del engranaje) y, después, **Tenant Model settings** (Configuración de Tenant Model).

The screenshot shows the Speech Studio interface. At the top, there's a header bar with the text 'Cognitive Services | Speech Customization'. Below the header, there's a navigation bar with icons for search, gear, help, and user profile. The main content area has sections for 'Subscription' and 'Tenant Model settings'. A red box highlights the 'Tenant Model settings' link under the 'Subscription' section.

Speech Studio muestra un mensaje que le indica que si es apto para crear un modelo de inquilino.

#### NOTE

Los clientes empresariales de Office 365 en Norteamérica son aptos para crear un modelo de inquilino (inglés). Para los clientes de Caja de seguridad del cliente, Clave de cliente u Office 365 Administración Pública característica no está disponible. Para determinar si es un cliente de Caja de seguridad del cliente o de Clave de cliente, consulte:

- [Caja de seguridad del cliente](#)
- [Clave de cliente](#)
- [Office 365 Administración Pública](#)

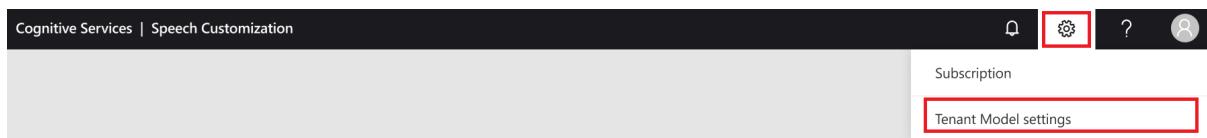
### 3. Seleccione **Habilitar envío**.

Cuando el modelo de inquilino esté listo, recibirá un mensaje de correo electrónico de confirmación con más instrucciones.

## Implementación de un modelo de inquilino

Cuando la instancia del modelo de inquilino esté lista, siga estos pasos para implementarla:

1. En el mensaje de correo electrónico de confirmación, seleccione el botón **View model** (Ver modelo). O bien, inicie sesión en [Speech Studio](#).
2. En la parte superior derecha, seleccione **Settings** (Configuración) (ícono del engranaje) y, después, **Tenant Model settings** (Configuración de Tenant Model).



### 3. Seleccione **Implementar**.

Cuando el modelo se haya implementado, el estado cambiará a *Implementado*.

## Usar un modelo de inquilino con el SDK de Voz

Ahora que ha implementado el modelo, puede usarlo con el SDK de Voz. En esta sección, se usa un código de ejemplo para llamar al servicio de voz mediante la autenticación de Azure Active Directory (Azure AD).

Echemos un vistazo al código que usará para llamar al SDK de Voz en C#. En este ejemplo, el reconocimiento de voz se realiza mediante un modelo de inquilino. En esta guía se da por supuesto que la plataforma ya está configurada. Si necesita ayuda para el programa de configuración, consulte [Inicio rápido: Reconocimiento de voz, C# \(.NET Core\)](#).

Copie este código en el proyecto:

```
namespace PrincetonSROnly.FrontEnd.Samples
{
    using System;
    using System.Collections.Generic;
    using System.IO;
    using System.Net.Http;
    using System.Text;
    using System.Text.RegularExpressions;
    using System.Threading.Tasks;
    using Microsoft.CognitiveServices.Speech;
    using Microsoft.CognitiveServices.Speech.Audio;
    using Microsoft.IdentityModel.Clients.ActiveDirectory;
    using Newtonsoft.Json.Linq;
```

```

// ServiceApplicationId is a fixed value. No need to change it.

public class TenantLMSample
{
    private const string EndpointUriArgName = "EndpointUri";
    private const string SubscriptionKeyArgName = "SubscriptionKey";
    private const string UsernameArgName = "Username";
    private const string PasswordArgName = "Password";
    private const string ClientApplicationId = "f87bc118-1576-4097-93c9-dbf8f45ef0dd";
    private const string ServiceApplicationId = "18301695-f99d-4cae-9618-6901d4bdc7be";

    public static async Task ContinuousRecognitionWithTenantLMAsync(Uri endpointUri, string
subscriptionKey, string audioDirPath, string username, string password)
    {
        var config = SpeechConfig.FromEndpoint(endpointUri, subscriptionKey);

        // Passing client specific information for obtaining LM
        if (string.IsNullOrEmpty(username) || string.IsNullOrEmpty(password))
        {
            config.AuthorizationToken = await
AcquireAuthTokenWithInteractiveLoginAsync().ConfigureAwait(false);
        }
        else
        {
            config.AuthorizationToken = await AcquireAuthTokenWithUsernamePasswordAsync(username,
password).ConfigureAwait(false);
        }

        var stopRecognition = new TaskCompletionSource<int>();

        // Creates a speech recognizer using file as audio input.
        // Replace with your own audio file name.
        using (var audioInput = AudioConfig.FromWavFileInput(audioDirPath))
        {
            using (var recognizer = new SpeechRecognizer(config, audioInput))
            {
                // Subscribes to events
                recognizer.Recognizing += (s, e) =>
                {
                    Console.WriteLine($"RECOGNIZING: Text={e.Result.Text}");
                };

                recognizer.Recognized += (s, e) =>
                {
                    if (e.Result.Reason == ResultReason.RecognizedSpeech)
                    {
                        Console.WriteLine($"RECOGNIZED: Text={e.Result.Text}");
                    }
                    else if (e.Result.Reason == ResultReason.NoMatch)
                    {
                        Console.WriteLine($"NOMATCH: Speech could not be recognized.");
                    }
                };
            };

            recognizer.Canceled += (s, e) =>
            {
                Console.WriteLine($"CANCELED: Reason={e.Reason}");
                if (e.Reason == CancellationReason.Error)
                {
                    Exception exp = new Exception(string.Format("Error Code: {0}\nError Details{1}\nIs
your subscription information updated?", e.ErrorCode, e.ErrorDetails));
                    throw exp;
                }

                stopRecognition.TrySetResult(0);
            };
        }

        recognizer.SessionStarted += (s, e) =>
    }
}

```

```

        Console.WriteLine("\n    Session started event.");
    };

    recognizer.SessionStopped += (s, e) =>
    {
        Console.WriteLine("\n    Session stopped event.");
        Console.WriteLine("\nStop recognition.");
        stopRecognition.TrySetResult(0);
    };
}

// Starts continuous recognition. Uses StopContinuousRecognitionAsync() to stop
recognition.
await recognizer.StartContinuousRecognitionAsync().ConfigureAwait(false);

// Waits for completion.
// Use Task.WaitAny to keep the task rooted.
Task.WaitAny(new[] { stopRecognition.Task });

// Stops recognition.
await recognizer.StopContinuousRecognitionAsync().ConfigureAwait(false);
}

}

public static void Main(string[] args)
{
    var arguments = new Dictionary<string, string>();
    string inputArgNamePattern = "--";
    Regex regex = new Regex(inputArgNamePattern);
    if (args.Length > 0)
    {
        foreach (var arg in args)
        {
            var userArgs = arg.Split("=");
            arguments[regex.Replace(userArgs[0], string.Empty)] = userArgs[1];
        }
    }

    var endpointString = arguments.GetValueOrDefault(EndpointUriArgName,
$"wss://westus.online.princeton.customspeech.ai/msgraphcustomspeech/conversation/v1");
    var endpointUri = new Uri(endpointString);

    if (!arguments.ContainsKey(SubscriptionKeyArgName))
    {
        Exception exp = new Exception("Subscription Key missing! Please pass in a Cognitive services
subscription Key using --SubscriptionKey=\"your_subscription_key\" +
                    "Find more information on creating a Cognitive services resource and accessing your
Subscription key here: https://docs.microsoft.com/azure/cognitive-services/cognitive-services-apis-create-account?tabs=multiservice%2Cwindows");
        throw exp;
    }

    var subscriptionKey = arguments[SubscriptionKeyArgName];
    var username = arguments.GetValueOrDefault(UsernameArgName, null);
    var password = arguments.GetValueOrDefault>PasswordArgName, null);

    var audioDirPath =
Path.Combine(Path.GetDirectoryName(System.Reflection.Assembly.GetExecutingAssembly().Location),
"../../../../AudioSamples/DictationBatman.wav");
    if (!File.Exists(audioDirPath))
    {
        Exception exp = new Exception(string.Format("Audio File does not exist at path: {0}",
audioDirPath));
        throw exp;
    }

    ContinuousRecognitionWithTenantLMAsync(endpointUri, subscriptionKey, audioDirPath, username,
password).GetAwaiter().GetResult();
}

```

```

    }

    private static async Task<string> AcquireAuthTokenWithUsernamePasswordAsync(string username, string
password)
{
    var tokenEndpoint = "https://login.microsoftonline.com/common/oauth2/token";
    var postBody = $"resource={ServiceApplicationId}&client_id=
{ClientApplicationId}&grant_type=password&username={username}&password={password}";
    var stringContent = new StringContent(postBody, Encoding.UTF8, "application/x-www-form-
urlencoded");
    using (HttpClient httpClient = new HttpClient())
    {
        var response = await httpClient.PostAsync(tokenEndpoint, stringContent).ConfigureAwait(false);

        if (response.IsSuccessStatusCode)
        {
            var result = await response.Content.ReadAsStringAsync().ConfigureAwait(false);

            JObject jobject = JObject.Parse(result);
            return jobject["access_token"].Value<string>();
        }
        else
        {
            throw new Exception($"Requesting token from {tokenEndpoint} failed with status code
{response.StatusCode}: {await response.Content.ReadAsStringAsync().ConfigureAwait(false)}");
        }
    }
}

private static async Task<string> AcquireAuthTokenWithInteractiveLoginAsync()
{
    var authContext = new AuthenticationContext("https://login.windows.net/microsoft.onmicrosoft.com");
    var deviceCodeResult = await authContext.AcquireDeviceCodeAsync(ServiceApplicationId,
ClientApplicationId).ConfigureAwait(false);

    Console.WriteLine(deviceCodeResult.Message);

    var authResult = await
authContext.AcquireTokenByDeviceCodeAsync(deviceCodeResult).ConfigureAwait(false);

    return authResult.AccessToken;
}
}
}

```

Después tendrá que volver a compilar y ejecutar el proyecto desde la línea de comandos. Antes de ejecutar el comando, actualice algunos parámetros, para lo que debe seguir estos pasos:

1. Reemplace <Username> y <Password> por los valores de un usuario de inquilino válido.
2. Reemplace <Subscription-Key> por la clave de suscripción del recurso de Voz. Este valor está disponible en la hoja de **información general** del recurso de Voz de [Azure Portal](#).
3. Reemplace <Endpoint-Uri> por el siguiente punto de conexión. Asegúrese de que reemplaza <your region> por la región donde se creó el recurso de Voz. Se admiten estas regiones: westus , westus2 y eastus . La información de la región está disponible en la sección de **información general** del recurso de Voz de [Azure Portal](#).

"wss://<your region>.online.princeton.customspeech.ai/msgraphcustomspeech/conversation/v1".

4. Ejecute el siguiente comando:

```
dotnet TenantLMSample.dll --Username=<Username> --Password=<Password> --SubscriptionKey=<Subscription-  
Key> --EndpointUri=<Endpoint-Uri>
```

En este tutorial, ha aprendido a usar los datos de Office 365 para crear un modelo de reconocimiento de voz personalizado, a implementarlo y a usarlo con el SDK de Voz.

## Pasos siguientes

- [Speech Studio](#)
- [Acerca del SDK de Voz](#)

# Procedimientos para: Selección de un dispositivo de entrada de audio con el SDK de Voz

13/01/2020 • 7 minutes to read • [Edit Online](#)

La versión 1.3.0 del SDK de Voz presenta una API para seleccionar la entrada de audio. En este artículo se describe cómo obtener los identificadores de los dispositivos de audio conectados a un sistema. El SDK de Voz puede usar estos posteriormente configurando el dispositivo de audio mediante el objeto `AudioConfig`:

```
audioConfig = AudioConfig.FromMicrophoneInput("<device id>");
```

```
audioConfig = AudioConfig.FromMicrophoneInput("<device id>");
```

```
audio_config = AudioConfig(device_name=<device id>);
```

```
audioConfig = AudioConfiguration.FromMicrophoneInput("<device id>");
```

```
audioConfig = AudioConfiguration.fromMicrophoneInput("<device id>");
```

```
audioConfig = AudioConfiguration.fromMicrophoneInput("<device id>");
```

## NOTE

El uso del micrófono no está disponible para JavaScript cuando se ejecuta en Node.js.

## Id. de dispositivos de audio en Windows para aplicaciones de escritorio

Las [cadena del identificador de punto de conexión](#) del dispositivo de audio se puede recuperar del objeto `IMMDevice` de Windows para las aplicaciones de escritorio.

En el siguiente ejemplo de código se muestra cómo usarlo para enumerar los dispositivos de audio de C++:

```
#include <cstdio>
#include <mmdeviceapi.h>

#include <FunctionDiscoveryKeys_devpkey.h>

const CLSID CLSID_MMDeviceEnumerator = __uuidof(MMDeviceEnumerator);
const IID IID_IMMDeviceEnumerator = __uuidof(IMMDeviceEnumerator);

constexpr auto REFTIMES_PER_SEC = (10000000 * 25);
constexpr auto REFTIMES_PER_MILLISEC = 10000;

#define EXIT_ON_ERROR(hres) \
    if (FAILED(hres)) { goto Exit; }

#define SAFE_RELEASE(punk) \
    if ((punk) != NULL) \
        punk->Release();
```

```

    { (punk)->Release(); (punk) = NULL; }

void ListEndpoints();

int main()
{
    CoInitializeEx(NULL, COINIT_MULTITHREADED);
    ListEndpoints();
}

//-----
// This function enumerates all active (plugged in) audio
// rendering endpoint devices. It prints the friendly name
// and endpoint ID string of each endpoint device.
//-----
void ListEndpoints()
{
    HRESULT hr = S_OK;
    IMMDeviceEnumerator *pEnumerator = NULL;
    IMMDeviceCollection *pCollection = NULL;
    IMMDevice *pEndpoint = NULL;
    IPropertyStore *pProps = NULL;
    LPWSTR pwszID = NULL;

    hr = CoCreateInstance(CLSID_MMDeviceEnumerator, NULL, CLSCTX_ALL, IID_IMMDeviceEnumerator,
    (void**)&pEnumerator);
    EXIT_ON_ERROR(hr);

    hr = pEnumerator->EnumAudioEndpoints(eCapture, DEVICE_STATE_ACTIVE, &pCollection);
    EXIT_ON_ERROR(hr);

    UINT count;
    hr = pCollection->GetCount(&count);
    EXIT_ON_ERROR(hr);

    if (count == 0)
    {
        printf("No endpoints found.\n");
    }

    // Each iteration prints the name of an endpoint device.
    PROPVARIANT varName;
    for (ULONG i = 0; i < count; i++)
    {
        // Get pointer to endpoint number i.
        hr = pCollection->Item(i, &pEndpoint);
        EXIT_ON_ERROR(hr);

        // Get the endpoint ID string.
        hr = pEndpoint->GetId(&pwszID);
        EXIT_ON_ERROR(hr);

        hr = pEndpoint->OpenPropertyStore(
            STGM_READ, &pProps);
        EXIT_ON_ERROR(hr);

        // Initialize container for property value.
        PropVariantInit(&varName);

        // Get the endpoint's friendly-name property.
        hr = pProps->GetValue(PKEY_Device_FriendlyName, &varName);
        EXIT_ON_ERROR(hr);

        // Print endpoint friendly name and endpoint ID.
        printf("Endpoint %d: \"%S\" (%S)\n", i, varName.pwszVal, pwszID);
    }

    Exit:
    CoTaskMemFree(pwszID);
}

```

```

    pwszID = NULL;
    PropVariantClear(&varName);
    SAFE_RELEASE(pEnumerator);
    SAFE_RELEASE(pCollection);
    SAFE_RELEASE(pEndpoint);
    SAFE_RELEASE(pProps);
}

```

En C#, la biblioteca [NAudio](#) se puede utilizar para acceder a CoreAudio API y enumerar los dispositivos de la siguiente manera:

```

using System;

using NAudio.CoreAudioApi;

namespace ConsoleApp
{
    class Program
    {
        static void Main(string[] args)
        {
            var enumerator = new MMDeviceEnumerator();
            foreach (var endpoint in
                enumerator.EnumerateAudioEndPoints(DataFlow.Capture, DeviceState.Active))
            {
                Console.WriteLine("{0} ({1})", endpoint.FriendlyName, endpoint.ID);
            }
        }
    }
}

```

Este sería un ejemplo de id. de dispositivo: `{0.0.1.00000000}.{5f23ab69-6181-4f4a-81a4-45414013aac8}`.

## Id. de dispositivos de audio en UWP

En la Plataforma universal de Windows (UWP), se pueden obtener dispositivos de entrada de audio mediante la propiedad `Id()` del objeto [DeviceInformation](#) correspondiente.

Los siguientes ejemplos de código muestran cómo hacer esto en C++ y C#:

```

#include <winrt/Windows.Foundation.h>
#include <winrt/Windows.Devices.Enumeration.h>

using namespace winrt::Windows::Devices::Enumeration;

void enumerateDeviceIds()
{
    auto promise = DeviceInformation::FindAllAsync(DeviceClass::AudioCapture);

    promise.Completed(
        []([winrt::Windows::Foundation::IAsyncOperation<DeviceInformationCollection> const& sender,
            winrt::Windows::Foundation::AsyncStatus /* asyncStatus */) {
            auto info = sender.GetResults();
            auto num_devices = info.Size();

            for (const auto &device : info)
            {
                std::wstringstream ss{};
                ss << "looking at device (of " << num_devices << "): " << device.Id().c_str() << "\n";
                OutputDebugString(ss.str().c_str());
            }
        });
}

```

```

using Windows.Devices.Enumeration;
using System.Linq;

namespace helloworld {
    private async void EnumerateDevices()
    {
        var devices = await DeviceInformation.FindAllAsync(DeviceClass.AudioCapture);

        foreach (var device in devices)
        {
            Console.WriteLine($"{device.Name}, {device.Id}\n");
        }
    }
}

```

Este sería un ejemplo de id. de dispositivo:

```
\\?\SWD#MMDEVAPI#{0.0.1.00000000}.{5f23ab69-6181-4f4a-81a4-45414013aac8}#{2eef81be-33fa-4800-9670-1cd474972c3f}
```

## Id. de dispositivos de audio en Linux

Los identificadores de dispositivo se seleccionan mediante identificadores de dispositivo ALSA estándar.

Los identificadores de las entradas conectadas al sistema se encuentran en la salida del comando `arecord -L`. Como alternativa, estas se pueden obtener mediante la [biblioteca C de ALSA](#).

Los identificadores de ejemplo son `hw:1,0` y `hw:CARD=CC,DEV=0`.

## Id. de dispositivos de audio en macOS

La siguiente función que se implementa en Objective-C crea una lista de los nombres e identificadores de los dispositivos de audio conectados a un Mac.

La cadena `deviceUID` se usa para identificar un dispositivo en el SDK de Voz de macOS.

```
#import <Foundation/Foundation.h>
```

```

#import <CoreAudio/CoreAudio.h>

CFArrayRef CreateInputDeviceArray()
{
    AudioObjectPropertyAddress propertyAddress = {
        kAudioHardwarePropertyDevices,
        kAudioObjectPropertyScopeGlobal,
        kAudioObjectPropertyElementMaster
    };

    UInt32 dataSize = 0;
    OSStatus status = AudioObjectGetPropertyDataSize(kAudioObjectSystemObject, &propertyAddress, 0, NULL,
&dataSize);
    if (kAudioHardwareNoError != status) {
        fprintf(stderr, "AudioObjectGetPropertyDataSize (kAudioHardwarePropertyDevices) failed: %i\n",
status);
        return NULL;
    }

    UInt32 deviceCount = (uint32)(dataSize / sizeof(AudioDeviceID));

    AudioDeviceID *audioDevices = (AudioDeviceID *)malloc(dataSize));
    if (NULL == audioDevices) {
        fputs("Unable to allocate memory", stderr);
        return NULL;
    }

    status = AudioObjectGetPropertyData(kAudioObjectSystemObject, &propertyAddress, 0, NULL, &dataSize,
audioDevices);
    if (kAudioHardwareNoError != status) {
        fprintf(stderr, "AudioObjectGetPropertyData (kAudioHardwarePropertyDevices) failed: %i\n", status);
        free(audioDevices);
        audioDevices = NULL;
        return NULL;
    }

    CFMutableArrayRef inputDeviceArray = CFArrayCreateMutable(kCFAllocatorDefault, deviceCount,
&kCFTypeArrayCallBacks);
    if (NULL == inputDeviceArray) {
        fputs("CFArrayCreateMutable failed", stderr);
        free(audioDevices);
        audioDevices = NULL;
        return NULL;
    }

    // Iterate through all the devices and determine which are input-capable
    propertyAddress.mScope = kAudioDevicePropertyScopeInput;
    for (UInt32 i = 0; i < deviceCount; ++i) {
        // Query device UID
        CFStringRef deviceUID = NULL;
        dataSize = sizeof(deviceUID);
        propertyAddress.mSelector = kAudioDevicePropertyDeviceUID;
        status = AudioObjectGetPropertyData(audioDevices[i], &propertyAddress, 0, NULL, &dataSize,
&deviceUID);
        if (kAudioHardwareNoError != status) {
            fprintf(stderr, "AudioObjectGetPropertyData (kAudioDevicePropertyDeviceUID) failed: %i\n",
status);
            continue;
        }

        // Query device name
        CFStringRef deviceName = NULL;
        dataSize = sizeof(deviceName);
        propertyAddress.mSelector = kAudioDevicePropertyNameCFString;
        status = AudioObjectGetPropertyData(audioDevices[i], &propertyAddress, 0, NULL, &dataSize,
&deviceName);
        if (kAudioHardwareNoError != status) {
            fprintf(stderr, "AudioObjectGetPropertyData (kAudioDevicePropertyNameCFString) failed:
%i\n", status);
        }
    }
}

```

```

        continue;
    }

    // Determine if the device is an input device (it is an input device if it has input channels)
    dataSize = 0;
    propertyAddress.mSelector = kAudioDevicePropertyStreamConfiguration;
    status = AudioObjectGetPropertyDataSize(audioDevices[i], &propertyAddress, 0, NULL, &dataSize);
    if (kAudioHardwareNoError != status) {
        fprintf(stderr, "AudioObjectGetPropertyDataSize (kAudioDevicePropertyStreamConfiguration) failed:\n", status);
        continue;
    }

    AudioBufferList *bufferList = (AudioBufferList *)malloc(dataSize));
    if (NULL == bufferList) {
        fputs("Unable to allocate memory", stderr);
        break;
    }

    status = AudioObjectGetPropertyData(audioDevices[i], &propertyAddress, 0, NULL, &dataSize,
    bufferList);
    if (kAudioHardwareNoError != status || 0 == bufferList->mNumberBuffers) {
        if (kAudioHardwareNoError != status)
            fprintf(stderr, "AudioObjectGetPropertyData (kAudioDevicePropertyStreamConfiguration) failed:\n", status);
        free(bufferList);
        bufferList = NULL;
        continue;
    }

    free(bufferList);
    bufferList = NULL;

    // Add a dictionary for this device to the array of input devices
    CFStringRef keys [] = { CFSTR("deviceUID"), CFSTR("deviceName")};
    CFStringRef values [] = { deviceUID, deviceName};

    CFDictionaryRef deviceDictionary = CFDictionaryCreate(kCFAllocatorDefault,
        (const void **)(keys),
        (const void **)(values),
        2,
        &kCFTypeDictionaryKeyCallBacks,
        &kCFTypeDictionaryValueCallBacks);

    CFArrayAppendValue(inputDeviceArray, deviceDictionary);

    CFRelease(deviceDictionary);
    deviceDictionary = NULL;
}

free(audioDevices);
audioDevices = NULL;

// Return a non-mutable copy of the array
CFArrayRef immutableInputDeviceArray = CFArrayCreateCopy(kCFAllocatorDefault, inputDeviceArray);
CFRelease(inputDeviceArray);
inputDeviceArray = NULL;

return immutableInputDeviceArray;
}

```

Por ejemplo, el id. de usuario para el micrófono integrado es `BuiltInMicrophoneDevice`.

## Id. de dispositivos de audio en iOS

La selección de dispositivos de audio con el SDK de Voz no se admite en iOS. No obstante, las aplicaciones que

usan el SDK pueden influir en el enrutamiento de audio mediante la plataforma [AVAudioSession](#).

Por ejemplo, la instrucción

```
[[AVAudioSession sharedInstance] setCategory:AVAudioSessionCategoryRecord  
withOptions:AVAudioSessionCategoryOptionAllowBluetooth error:NULL];
```

permite el uso de un auricular Bluetooth para una aplicación habilitada para voz.

## Identificadores de dispositivos de audio en JavaScript

En JavaScript, el método [MediaDevices.enumerateDevices\(\)](#) puede usarse para enumerar los dispositivos multimedia y buscar un identificador de dispositivo para pasar a [fromMicrophone\(...\)](#).

## Pasos siguientes

[Exploración de ejemplos en GitHub](#)

## Otras referencias

- [Personalización de modelos acústicos](#)
- [Personalización de modelos de lenguaje](#)

# Detección automática de idioma para la conversión de voz en texto

13/01/2020 • 3 minutes to read • [Edit Online](#)

La detección automática de idioma se usa para determinar la coincidencia más probable para el audio que se pasa al SDK de Voz cuando se compara con una lista de idiomas proporcionados. El valor devuelto por la detección automática de idioma se usa para seleccionar el modelo de idioma para la conversión de voz en texto, lo que le proporciona una transcripción más precisa. Para ver los idiomas que están disponibles, consulte la [compatibilidad de idioma](#).

En este artículo, aprenderá a usar `AutoDetectSourceLanguageConfig` para construir un objeto `SpeechRecognizer` y recuperar el idioma detectado.

## IMPORTANT

Esta característica solo está disponible para el SDK de Voz para C++ y el SDK de Voz para Java.

## Detección automática de idioma con el SDK de Voz

La detección automática de idioma actualmente tiene un límite de servicio de dos idiomas por detección. Tenga en cuenta esta limitación al construir el objeto `AutoDetectSourceLanguageConfig`. En los ejemplos siguientes, creará un objeto `AutoDetectSourceLanguageConfig` y lo usará para construir un `SpeechRecognizer`.

## TIP

También puede especificar un modelo personalizado para utilizarlo al realizar la conversión de voz en texto. Para obtener más información, consulte [Uso de un modelo personalizado para la detección automática de idioma](#).

En los fragmentos de código siguientes se muestra cómo usar la detección automática de idioma en las aplicaciones:

```
auto autoDetectSourceLanguageConfig = AutoDetectSourceLanguageConfig::FromLanguages({ "en-US", "de-DE" });
auto recognizer = SpeechRecognizer::FromConfig(speechConfig, autoDetectSourceLanguageConfig, audioConfig);
speechRecognitionResult = recognizer->RecognizeOnceAsync().get();
auto autoDetectSourceLanguageResult = AutoDetectSourceLanguageResult::FromResult(speechRecognitionResult);
auto detectedLanguage = autoDetectSourceLanguageResult->Language;
```

```

AutoDetectSourceLanguageConfig autoDetectSourceLanguageConfig =
AutoDetectSourceLanguageConfig.fromLanguages(Arrays.asList("en-US", "de-DE"));
SpeechRecognizer recognizer = new SpeechRecognizer(speechConfig, autoDetectSourceLanguageConfig, audioConfig);
Future<SpeechRecognitionResult> future = recognizer.recognizeOnceAsync();
SpeechRecognitionResult result = future.get(30, TimeUnit.SECONDS);
AutoDetectSourceLanguageResult autoDetectSourceLanguageResult =
AutoDetectSourceLanguageResult.fromResult(result);
String detectedLanguage = autoDetectSourceLanguageResult.getLanguage();

recognizer.close();
speechConfig.close();
autoDetectSourceLanguageConfig.close();
audioConfig.close();
result.close();

```

## Uso de un modelo personalizado para la detección automática de idioma

Además de la detección de idiomas mediante los modelos del servicio de Voz, puede especificar un modelo personalizado para obtener reconocimiento mejorado. Si no se proporciona un modelo personalizado, el servicio utilizará el modelo de lenguaje predeterminado.

En los fragmentos de código siguientes se muestra cómo especificar un modelo personalizado en la llamada al servicio de Voz. Si el idioma detectado es `en-US`, se usa el modelo predeterminado. Si el idioma detectado es `fr-FR`, se usa el punto de conexión para el modelo personalizado:

```

std::vector<std::shared_ptr<SourceLanguageConfig>> sourceLanguageConfigs;
sourceLanguageConfigs.push_back(SourceLanguageConfig::FromLanguage("en-US"));
sourceLanguageConfigs.push_back(SourceLanguageConfig::FromLanguage("fr-FR", "The Endpoint Id for custom model
of fr-FR"));
auto autoDetectSourceLanguageConfig =
AutoDetectSourceLanguageConfig::FromSourceLanguageConfigs(sourceLanguageConfigs);

```

```

List sourceLanguageConfigs = new ArrayList<SourceLanguageConfig>();
sourceLanguageConfigs.add(SourceLanguageConfig.fromLanguage("en-US"));
sourceLanguageConfigs.add(SourceLanguageConfig.fromLanguage("fr-FR", "The Endpoint Id for custom model of fr-
FR"));
AutoDetectSourceLanguageConfig autoDetectSourceLanguageConfig =
AutoDetectSourceLanguageConfig.fromSourceLanguageConfigs(sourceLanguageConfigs);

```

## Pasos siguientes

- [Documentación de referencia del SDK de voz](#)

# Uso de entradas de audio comprimido con códec con el SDK de voz

13/01/2020 • 2 minutes to read • [Edit Online](#)

La API **Compressed Audio Input Stream** del SDK de voz le proporciona una forma de hacer streaming de audio comprimido al servicio de voz mediante PullStream o PushStream.

## IMPORTANT

Solo se admite la secuencia de entrada de audio comprimido para C++, C# y Java para Linux (Ubuntu 16.04, Ubuntu 18.04, Debian 9). También se admite para [Java en Android](#) y [Objective-C en la plataforma iOS](#). Se requiere el SDK de voz versión 1.7.0 o posterior.

Para wav/PCM consulte la documentación principal de voz. Aparte de wav/PCM, se admiten los siguientes formatos de entrada de audio comprimido con códec:

- MP3
- OPUS/OGG
- FLAC
- ALAW en el contenedor WAV
- MULAW en el contenedor WAV

## Requisitos previos

El control del audio comprimido se implementa mediante [GStreamer](#). Por motivos de licencia, los archivos binarios de Gstreamer no se compilan ni vinculan con el SDK de voz. Por lo tanto, el desarrollador de aplicaciones debe instalar lo siguiente en 18.04, 16.04 y Debian 9 para usar el audio de entrada comprimido.

```
sudo apt install libgstreamer1.0-0 gstreamer1.0-plugins-base gstreamer1.0-plugins-good gstreamer1.0-plugins-bad gstreamer1.0-plugins-ugly
```

## Ejemplo de código que usa una entrada de audio comprimido con códec

Para hacer streaming en un formato de audio comprimido al servicio de voz, cree `PullAudioInputStream` o `PushAudioInputStream`. A continuación, cree un objeto `AudioConfig` a partir de una instancia de la clase de secuencia, especificando el formato de compresión de la secuencia.

Supongamos que tiene una clase de flujo de entrada llamada `myPushStream` y que usa OPUS/OGG. El aspecto del código sería el siguiente:

```
using Microsoft.CognitiveServices.Speech;
using Microsoft.CognitiveServices.Speech.Audio;

var speechConfig = SpeechConfig.FromSubscription("YourSubscriptionKey", "YourServiceRegion");

// Create an audio config specifying the compressed audio format and the instance of your input stream class.
var audioFormat = AudioStreamFormat.GetCompressedFormat(AudioStreamContainerFormat.OGG_OPUS);
var audioConfig = AudioConfig.FromStreamInput(myPushStream, audioFormat);

var recognizer = new SpeechRecognizer(speechConfig, audioConfig);

var result = await recognizer.RecognizeOnceAsync();

var text = result.GetText();
```

## Pasos siguientes

- [Obtenga su suscripción de prueba a Voz](#)
- [See how to recognize speech in Java](#) (Vea cómo funciona el reconocimiento de voz en Java)

# Procedimientos para: Uso de entradas de audio comprimido con códec con SDK de voz en iOS

13/01/2020 • 5 minutes to read • [Edit Online](#)

La API **Compressed Audio Input Stream** del SDK de voz permite transmitir audio comprimido al servicio de voz mediante un flujo de inserción o extracción.

## IMPORTANT

Se requiere Speech SDK versión 1.7.0 o posterior para la secuencia de audio comprimido en iOS. También se admite para C++, C# y Java en Linux (Ubuntu 16.04, Ubuntu 18.04, Debian 9) y Java en Android.

Para wav/PCM consulte la documentación principal de voz. Aparte de wav/PCM, se admiten los siguientes formatos de entrada de audio comprimido con códec:

- MP3
- OPUS/OGG
- FLAC
- ALAW en el contenedor WAV
- MULAW en el contenedor WAV

## Requisitos previos

El control del audio comprimido se implementa mediante [GStreamer](#). Por motivos de concesión de licencias, estas funciones no se pueden enviar con el SDK, pero es necesario que los programadores de aplicaciones compilen una biblioteca de contenedor que contenga estas funciones y que se envíen con las aplicaciones mediante el SDK.

Para compilar esta biblioteca de contenedor, primero descargue e instale el [SDK de GStreamer](#). A continuación, descargue el proyecto de Xcode para la [biblioteca de contenedor](#).

Abra el proyecto en Xcode y compílelo para el destino **Generic iOS Device** (dispositivo iOS genérico); no funcionará si se compila para un destino específico.

El paso de compilación generará un conjunto dinámico de marcos con una biblioteca dinámica para todas las arquitecturas necesarias con el nombre de `GStreamerWrapper.framework`.

Esta plataforma debe estar incluida en todas las aplicaciones que usan secuencias de audio comprimidas con el SDK de voz.

Aplique la siguiente configuración en el proyecto de Xcode para lograr esto:

1. Copie tanto el `GStreamerWrapper.framework` que acaba de integrar como el marco de Speech SDK de Cognitive Services, que puede descargar [aquí](#), en el directorio que contiene el proyecto de ejemplo.
2. Ajuste las rutas de acceso para los marcos en la *Configuración del proyecto*.
  - a. En la pestaña **General**, en el encabezado **Binarios incrustados**, agregue la biblioteca del SDK como un marco de trabajo: **Agregar binarios insertados > Agregar otros...** > vaya al directorio que eligió y seleccione ambos marcos.
  - b. Vaya a la pestaña **Configuración de compilación** y active la configuración **Todos**.
3. Agregue el directorio `$(SRCROOT)/...` a *Rutas de búsqueda de marco* bajo el encabezado **Rutas de búsqueda**.

## Ejemplo de código que usa una entrada de audio comprimido con códec

Para hacer streaming en un formato de audio comprimido al servicio de voz, cree un elemento

`SPXPullAudioInputStream` o `SPXPushAudioInputStream`.

En el siguiente fragmento de código se muestra cómo crear un `SPXAudioConfiguration` a partir de una instancia de `SPXPushAudioInputStream`, especificando MP3 como formato de compresión de la secuencia.

```
// <setup-stream>
SPXAUDIOSTREAMCONTAINERFORMAT compressedStreamFormat = SPXAUDIOSTREAMCONTAINERFORMAT_MP3;
SPXAUDIOFORMAT *audioFormat = [[SPXAUDIOFORMAT alloc]
initUsingCompressedFormat:compressedStreamFormat];
SPXPUSHAUDIOINPUTSTREAM* stream = [[SPXPUSHAUDIOINPUTSTREAM alloc] initWithAudioFormat:audioFormat];

SPXAUDIOCONFIGURATION* audioConfig = [[SPXAUDIOCONFIGURATION alloc] initWithStreamInput:stream];
if (!audioConfig) {
    NSLog(@"Error creating stream!");
    [self updateRecognitionErrorText:(@"Error creating stream!")];
    return;
}
// </setup-stream>
```

El siguiente fragmento de código muestra cómo se pueden leer los datos de audio comprimidos de un archivo y bombearlos en `SPXPushAudioInputStream`.

```

// <push-compressed-stream>
NSInputStream *compressedStream = [[NSInputStream alloc] initWithFileAtPath:weatherFile];
[compressedStream open];
NSLog(@"result of opening stream: %@", compressedStream.streamError, (unsigned
long)compressedStream.streamStatus);
if (nil == compressedStream)
{
    NSLog(@"Error while opening file");
    audioConfig = nil;
    return;
}

// start recognizing
[self updateRecognitionStatusText:(@"Recognizing from push stream...")];
[speechRecognizer startContinuousRecognition];

const NSInteger nBytesToRead = 1000;
// push data to stream;
uint8_t *buffer = malloc(nBytesToRead);
NSInteger nBytesRead = 0;
while (1)
{
    // read data
    nBytesRead = [compressedStream read:buffer maxLength:nBytesToRead];
    if (0 == nBytesRead) {
        NSLog(@"end of stream reached");
        [stream close];
        break;
    }
    else if (0 > nBytesRead) {
        NSLog(@"error reading stream (%ld): %@", nBytesRead, compressedStream.streamError,
compressedStream.streamStatus);
        [stream close];
        break;
    }
    else
    {
        NSLog(@"Read %lu bytes from file", nBytesRead);
        NSData *data = [NSData dataWithBytesNoCopy:buffer length:nBytesRead freeWhenDone:NO];

        [stream write:data];
        NSLog(@"Wrote %lu bytes to stream", [data length]);
    }
}

[NSThread sleepForTimeInterval:0.1f];
}

[speechRecognizer stopContinuousRecognition];
// </push-compressed-stream>

```

## Pasos siguientes

- [Obtenga su suscripción de prueba a Voz](#)
- [See how to recognize speech in Java](#) (Vea cómo funciona el reconocimiento de voz en Java)

# Procedimientos para: Uso de entradas de audio comprimido con códec con el SDK de Voz en Android

13/01/2020 • 4 minutes to read • [Edit Online](#)

La API **Compressed Audio Input Stream** del SDK de voz le proporciona una forma de hacer streaming de audio comprimido al servicio de voz mediante PullStream o PushStream.

## IMPORTANT

Solo se admite la secuencia de entrada de audio comprimido para **C++, C# y Java para Linux (Ubuntu 16.04, Ubuntu 18.04, Debian 9)**. También se admite para Java en Android y **Objective-C en la plataforma iOS**. Se requiere el SDK de voz versión 1.7.0 o posterior.

Para wav/PCM consulte la documentación principal de voz. Aparte de wav/PCM, se admiten los siguientes formatos de entrada de audio comprimido con códec:

- MP3
- OPUS/OGG
- FLAC
- ALAW en el contenedor WAV
- MULAW en el contenedor WAV

## Requisitos previos para el uso de entradas de audio comprimido con códec en Android

El audio comprimido de códec se implementa mediante **GStreamer**. Por motivos de licencias, los archivos binarios de Gstreamer no se compilan con el SDK. Tendrá que usar los archivos binarios compilados previamente para Android. Para descargar las bibliotecas compiladas previamente, consulte [Instalación para Android Development](#).

`libgstreamer_android.so` es obligatorio. Asegúrese de que los complementos de Gstreamer están vinculados en `libgstreamer_android.so`.

```
GSTREAMER_PLUGINS := coreelements app audioconvert mpg123 audioresample audioparsers ogg opusparse opus  
wavparse alaw mulaw flac
```

A continuación se proporciona un ejemplo de archivo `Android.mk` y `Application.mk`. Siga estos pasos para crear el objeto compartido gstreamer: `libgstreamer_android.so`.

```

# Android.mk
LOCAL_PATH := $(call my-dir)

include $(CLEAR_VARS)

LOCAL_MODULE     := dummy
LOCAL_SHARED_LIBRARIES := gstreamer_android
LOCAL_LDLIBS := -llog
include $(BUILD_SHARED_LIBRARY)

ifndef GSTREAMER_ROOT_ANDROID
$(error GSTREAMER_ROOT_ANDROID is not defined!)
endif

ifndef APP_BUILD_SCRIPT
$(error APP_BUILD_SCRIPT is not defined!)
endif

ifndef TARGET_ARCH_ABI
$(error TARGET_ARCH_ABI is not defined!)
endif

ifeq ($(TARGET_ARCH_ABI),armeabi)
GSTREAMER_ROOT      := $(GSTREAMER_ROOT_ANDROID)/arm
else ifeq ($(TARGET_ARCH_ABI),armeabi-v7a)
GSTREAMER_ROOT      := $(GSTREAMER_ROOT_ANDROID)/armv7
else ifeq ($(TARGET_ARCH_ABI),arm64-v8a)
GSTREAMER_ROOT      := $(GSTREAMER_ROOT_ANDROID)/arm64
else ifeq ($(TARGET_ARCH_ABI),x86)
GSTREAMER_ROOT      := $(GSTREAMER_ROOT_ANDROID)/x86
else ifeq ($(TARGET_ARCH_ABI),x86_64)
GSTREAMER_ROOT      := $(GSTREAMER_ROOT_ANDROID)/x86_64
else
$(error Target arch ABI not supported: $(TARGET_ARCH_ABI))
endif

GSTREAMER_NDK_BUILD_PATH  := $(GSTREAMER_ROOT)/share/gst-android/ndk-build/
include $(GSTREAMER_NDK_BUILD_PATH)/plugins.mk
GSTREAMER_PLUGINS      := coreelements app audioconvert mpg123 audioresample audioparsers ogg opusparse
opus wavparse alaw mulaw flac
GSTREAMER_EXTRA_LIBS    := -liconv
include $(GSTREAMER_NDK_BUILD_PATH)/gstreamer-1.0.mk

```

```

# Application.mk
APP_STL = c++_shared
APP_PLATFORM = android-21
APP_BUILD_SCRIPT = Android.mk

```

Puede compilar `libgstreamer_android.so` con el siguiente comando en Ubuntu 16.04 o 18.04. Las siguientes líneas de comandos solo se han probado para la [versión Gstreamer de Android 1.14.4](#) con [Android NDK b16b](#).

```

# assuming wget and unzip already installed on the system
mkdir buildLibGstreamer
cd buildLibGstreamer
wget https://dl.google.com/android/repository/android-ndk-r16b-linux-x86_64.zip
unzip -q -o android-ndk-r16b-linux-x86_64.zip
export PATH=$PATH:$(pwd)/android-ndk-r16b
export NDK_PROJECT_PATH=$(pwd)/android-ndk-r16b
wget https://gstreamer.freedesktop.org/data/pkg/android/1.14.4/gstreamer-1.0-android-universal-1.14.4.tar.bz2
mkdir gstreamer_android
tar -xjf gstreamer-1.0-android-universal-1.14.4.tar.bz2 -C $(pwd)/gstreamer_android/
export GSTREAMER_ROOT_ANDROID=$(pwd)/gstreamer_android

mkdir gstreamer
# Copy the Application.mk and Android.mk from the documentation above and put it inside $(pwd)/gstreamer

# Enable only one of the following at one time to create the shared object for the targeted ABI
echo "building for armeabi-v7a. libgstreamer_android.so will be placed in $(pwd)/armeabi-v7a"
ndk-build -C $(pwd)/gstreamer "NDK_APPLICATION_MK=Application.mk" APP_ABI=armeabi-v7a NDK_LIBS_OUT=$(pwd)

#echo "building for arm64-v8a. libgstreamer_android.so will be placed in $(pwd)/arm64-v8a"
#ndk-build -C $(pwd)/gstreamer "NDK_APPLICATION_MK=Application.mk" APP_ABI=arm64-v8a NDK_LIBS_OUT=$(pwd)

#echo "building for x86_64. libgstreamer_android.so will be placed in $(pwd)/x86_64"
#ndk-build -C $(pwd)/gstreamer "NDK_APPLICATION_MK=Application.mk" APP_ABI=x86_64 NDK_LIBS_OUT=$(pwd)

#echo "building for x86. libgstreamer_android.so will be placed in $(pwd)/x86"
#ndk-build -C $(pwd)/gstreamer "NDK_APPLICATION_MK=Application.mk" APP_ABI=x86 NDK_LIBS_OUT=$(pwd)

```

Una vez que el objeto compartido (libgstreamer\_android.so) esté integrado, el programador de aplicaciones debe colocar el objeto compartido en la aplicación de Android para que el SDK de voz pueda cargarlo.

## Ejemplo de código que usa una entrada de audio comprimido con códec

Para hacer streaming en un formato de audio comprimido al servicio de voz, cree `PullAudioInputStream` o `PushAudioInputStream`. A continuación, cree un objeto `AudioConfig` a partir de una instancia de la clase de secuencia, especificando el formato de compresión de la secuencia.

Supongamos que tiene una clase de flujo de entrada llamada `myPullStream` y que usa OPUS/OGG. El aspecto del código sería el siguiente:

```

import com.microsoft.cognitiveservices.speech.audio.AudioConfig;
import com.microsoft.cognitiveservices.speech.audio.AudioInputStream;
import com.microsoft.cognitiveservices.speech.audio.AudioStreamFormat;
import com.microsoft.cognitiveservices.speech.audio.PullAudioInputStream;
import com.microsoft.cognitiveservices.speech.internal.AudioStreamContainerFormat;

SpeechConfig speechConfig = SpeechConfig.fromSubscription("YourSubscriptionKey", "YourServiceRegion");

// Create an audio config specifying the compressed audio format and the instance of your input stream class.
AudioStreamFormat audioFormat = AudioStreamFormat.getCompressedFormat(AudioStreamContainerFormat.OGG_OPUS);
AudioConfig audioConfig = AudioConfig.fromStreamInput(myPullStream, audioFormat);

SpeechRecognizer recognizer = new SpeechRecognizer(speechConfig, audioConfig);

SpeechRecognitionResult result = recognizer.recognizeOnceAsync().get()

String text = result.getText();

```

## Pasos siguientes

- [Obtenga su suscripción de prueba a Voz](#)
- [See how to recognize speech in Java](#) (Vea cómo funciona el reconocimiento de voz en Java)

# Instalación y ejecución de contenedores del servicio de voz (versión preliminar)

14/01/2020 • 38 minutes to read • [Edit Online](#)

Los contenedores le permiten ejecutar algunas de las API del servicio de voz en su propio entorno. Los contenedores son excelentes para requisitos específicos de control de datos y seguridad. En este artículo, aprenderá a descargar, instalar y ejecutar un contenedor de Voz.

Los contenedores de Voz permiten a los clientes compilar una arquitectura de aplicación de voz optimizada para las sólidas funcionalidades de la nube y la localidad del perímetro. Hay cuatro contenedores distintos disponibles. Los dos contenedores estándar son **conversión de voz a texto** y **conversión de texto a voz**. Los dos contenedores personalizados son **conversión de voz a texto personalizada** y **conversión de texto a voz personalizada**.

## IMPORTANT

Actualmente, todos los contenedores de voz se ofrecen como parte de una [versión preliminar pública "validada"](#). Se hará un anuncio cuando los contenedores de voz pasen a la disponibilidad general (GA).

FUNCIÓN	CARACTERÍSTICAS	MÁS RECIENTE
Voz a texto	Transcribe registros continuos de voz en tiempo real o de audio por lotes a texto con resultados intermedios.	2.0.0
Conversión de voz a texto personalizada	Con un modelo personalizado del <a href="#">portal de Voz personalizada</a> , transcribe las grabaciones continuas de voz en tiempo real o de audio por lotes a texto con resultados inmediatos.	2.0.0
Texto a voz	Convierte texto a voz de sonido natural con entrada de texto sin formato o Lenguaje de marcado de síntesis de voz (SSML).	1.3.0
Conversión de texto a voz personalizada	Con un modelo personalizado del <a href="#">portal de Voz personalizada</a> , convierte texto a voz de sonido natural con entrada de texto sin formato o Lenguaje de marcado de síntesis de voz (SSML).	1.3.0

Si no tiene una suscripción a Azure, cree una [cuenta gratuita](#) antes de empezar.

## Prerequisites

Requisitos previos para poder usar los contenedores de Voz:

OBLIGATORIO	PROPÓSITO
-------------	-----------

OBLIGATORIO	PROPÓSITO
Motor de Docker	<p>Necesita que el motor de Docker esté instalado en un <a href="#">equipo host</a>. Docker dispone de paquetes que configuran el entorno de Docker en <a href="#">macOS</a>, <a href="#">Windows</a> y <a href="#">Linux</a>. Para conocer los principios básicos de Docker y de los contenedores, consulte <a href="#">Introducción a Docker</a>.</p> <p>Docker debe configurarse para permitir que los contenedores se conecten con Azure y envíen datos de facturación a dicho servicio.</p> <p><b>En Windows</b>, Docker también debe estar configurado de forma que admita los contenedores de Linux.</p>
Conocimientos sobre Docker	<p>Debe tener conocimientos básicos sobre los conceptos de Docker, como los registros, los repositorios, los contenedores y las imágenes de contenedor, así como conocer los comandos <code>docker</code> básicos.</p>
Recurso de Voz	<p>Para usar estos contenedores, debe tener:</p> <p>Recurso de Voz de Azure para obtener la clave de API y el URI de punto de conexión asociados. Ambos valores están disponibles en las páginas Introducción y Claves de <b>Voz</b> de Azure Portal. Los dos son necesarios para iniciar el contenedor.</p> <p><b>{API_KEY}</b> : una de las dos claves de recurso disponibles en la página <b>Claves</b></p> <p><b>{ENDPOINT_URI}</b> : el punto de conexión tal como se proporciona en la página de <b>Información general</b>.</p>

## Solicitud de acceso al registro de contenedor

Rellene y envíe el [formulario de solicitud de contenedores de Voz de Cognitive Services](#) para solicitar acceso al contenedor.

El formulario solicita información acerca del usuario y de su empresa, así como del escenario de usuario para el que se va a usar el contenedor. Después de haber enviado el formulario, el equipo de Azure Cognitive Services lo revisa para asegurarse de que cumple los criterios de acceso al registro de contenedor privado.

### IMPORTANT

Debe usar una dirección de correo electrónico que esté asociada con una cuenta de Microsoft (MSA) o de Azure Active Directory (Azure AD) en el formulario.

Si se aprueba la solicitud, recibirá un correo electrónico con instrucciones que describen cómo obtener las credenciales y tener acceso al registro de contenedor privado.

## Uso de la CLI de Docker para autenticar un registro de contenedor privado

Puede autenticarse con el registro de contenedor privado de contenedores de Cognitive Services, pero el método recomendado de la línea de comandos es mediante el uso de la [CLI de Docker](#).

Use el comando `docker login`, como se muestra en el ejemplo siguiente, para iniciar sesión en `containerpreview.azurecr.io`, el registro de contenedor privado de los contenedores de Cognitive Services.

Reemplace `<username>` por el nombre de usuario y `<password>` por la contraseña proporcionada en las credenciales que recibió del equipo de Azure Cognitive Services.

```
docker login containerpreview.azurecr.io -u <username> -p <password>
```

Si ha protegido las credenciales en un archivo de texto, puede concatenar el contenido de dicho archivo de texto, mediante el comando `cat`, al comando `docker login`, tal como se muestra en el ejemplo siguiente. Reemplace `<passwordFile>` por la ruta de acceso y el nombre del archivo de texto que contiene la contraseña, y `<username>` por el nombre de usuario proporcionado en las credenciales.

```
cat <passwordFile> | docker login containerpreview.azurecr.io -u <username> --password-stdin
```

## Recopilación de los parámetros obligatorios

Hay tres parámetros principales para todos los contenedores de Cognitive Services que son necesarios. El contrato de licencia para el usuario final (CLUF) debe estar presente con un valor de `accept`. Además, se necesitan una dirección URL de punto de conexión y una clave de API.

### URI de punto de conexión {ENDPOINT\_URI}

El valor del URI del **punto de conexión** está disponible en la página *Información general* de Azure Portal del recurso de Cognitive Services correspondiente. Vaya a la página *Información general*, mantenga el cursor sobre el punto de conexión y aparecerá un icono `Copy to clipboard`. Cópielo y utilícelo cuando sea necesario.

The screenshot shows the Azure Portal interface for a Cognitive Services resource named "widgets". On the left, there's a sidebar with navigation links like Home, widgets, Search (Ctrl+/,), Activity log, Access control (IAM), Tags, and Diagnose and solve problems. Below that is a RESOURCE MANAGEMENT section with Keys, Quick start, Pricing tier, and Billing By Subscription. The main content area has tabs: Overview (highlighted with a red arrow), Unavailable setting, and Delete. Under Overview, it shows details: Resource group (widgets-resource-group), Status (Active), Location (North Central US), Subscription (widgets-subscription), Subscription ID (xxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxx), and Tags (Click here to add tags). To the right, it shows API type (<API Type>), Pricing tier (Standard), Endpoint (<https://widgets.cognitiveservices.azure.com/api/example-endpoint>), Manage keys, and Show access keys. A red box highlights the "Copy to clipboard" button next to the Endpoint URL.

### Claves {API\_KEY}

Esta clave se usa para iniciar el contenedor y está disponible en la página de claves de Azure Portal del recurso de Cognitive Services correspondiente. Vaya a la página *Claves* y haga clic en el icono `Copy to clipboard`.

The screenshot shows the Azure Cognitive Services - Keys page. On the left, there's a sidebar with options like Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, RESOURCE MANAGEMENT, and Keys (which has a red arrow pointing to it). The main area shows a 'NAME' field with 'widgets'. Below it is a note about subscription keys: 'These subscription keys are used to access your Cognitive Service API. Do not share your keys. Store them securely— for example, using Azure Key Vault. We also recommend regenerating these keys regularly. Only one key is necessary to make an API call. When regenerating the first key, you can use the second key for continued access to the service.' Under 'KEY 1', there's a field with '<key 1 value>' and a red box around the copy icon. Under 'KEY 2', there's a field with '<key 2 value>' and a red box around the copy icon.

## IMPORTANT

Estas claves de suscripción se usan para tener acceso a la API de Cognitive Services. No comparta las claves. Almacénelas de forma segura, por ejemplo, con Azure Key Vault. También se recomienda regenerar estas claves periódicamente. Solo se necesita una clave para realizar una llamada API. Al volver a generar la primera clave, puede usar la segunda clave para seguir teniendo acceso al servicio.

## El equipo host

El host es un equipo basado en x64 que ejecuta el contenedor de Docker. Puede ser un equipo del entorno local o un servicio de hospedaje de Docker incluido en Azure, como:

- [Azure Kubernetes Service](#).
- [Azure Container Instances](#).
- Un clúster de [Kubernetes](#) implementado en [Azure Stack](#). Para obtener más información, consulte [Implementación de Kubernetes en Azure Stack](#).

## Compatibilidad con la extensión avanzada de vector

El **host** es el equipo que ejecuta el contenedor de Docker. El host *debe ser compatible* con las [extensiones avanzadas de vector](#) (AVX2). Puede comprobar la compatibilidad con AVX2 en los hosts de Linux con el comando siguiente:

```
grep -q avx2 /proc/cpuinfo && echo AVX2 supported || echo No AVX2 support detected
```

## WARNING

El equipo host es *necesario* para admitir AVX2. El contenedor *no* funcionará correctamente sin compatibilidad con AVX2.

## Recomendaciones y requisitos del contenedor

En la tabla siguiente se describe la asignación mínima y recomendada de recursos para cada contenedor de Voz.

- [Voz a texto](#)
- [Conversión de voz a texto personalizada](#)
- [Texto a voz](#)
- [Conversión de texto a voz personalizada](#)

CONTENEDOR	MÍNIMA	RECOMENDADO
Voz a texto	2 núcleos, 2 GB de memoria	4 núcleos, 4 GB de memoria

- Cada núcleo debe ser de 2,6 gigahercios (GHz) como mínimo.

El núcleo y la memoria se corresponden con los valores de `--cpus` y `--memory` que se usan como parte del comando `docker run`.

#### NOTE

Los valores mínimos y recomendados se basan en los límites de Docker y *no* en los recursos de la máquina host. Por ejemplo, partes de la asignación de memoria de contenedores de voz a texto de un modelo grande de lenguaje, y se *recomienda* que todo el archivo se ajuste en memoria, que es de 4 a 6 GB adicionales. Además, la primera ejecución de cualquier contenedor puede tardar más, dado que los modelos se van a paginar en la memoria.

## Obtención de la imagen del contenedor con `docker pull`

Las imágenes de contenedor para Voz están disponibles en la instancia de Container Registry siguiente.

- [Voz a texto](#)
- [Conversión de voz a texto personalizada](#)
- [Texto a voz](#)
- [Conversión de texto a voz personalizada](#)

CONTENEDOR	REPOSITORIO
Voz a texto	<code>containerpreview.azurecr.io/microsoft/cognitive-services-speech-to-text:latest</code>

#### TIP

Puede usar el comando `docker images` para enumerar las imágenes de contenedor descargadas. Por ejemplo, el comando siguiente muestra el id., el repositorio y la etiqueta de cada imagen de contenedor descargada, con formato de tabla:

```
docker images --format "table {{.ID}}\t{{.Repository}}\t{{.Tag}}"
IMAGE ID      REPOSITORY          TAG
<image-id>   <repository-path/name> <tag-name>
```

## Docker pull para los contenedores de Voz

- [Voz a texto](#)
- [Conversión de voz a texto personalizada](#)
- [Texto a voz](#)
- [Conversión de texto a voz personalizada](#)

### Docker pull para el contenedor de conversión de voz a texto

Use el comando `docker pull` para descargar una imagen de contenedor desde la versión preliminar del registro de contenedor.

```
docker pull containerpreview.azurecr.io/microsoft/cognitive-services-speech-to-text:latest
```

## IMPORTANT

La etiqueta `latest` extrae la configuración regional `en-US`. Para otras configuraciones regionales, consulte [Configuración regional de conversión de voz a texto](#).

### Configuraciones regionales de voz a texto

Todas las etiquetas, a excepción de `latest` tienen el formato siguiente y distinguen mayúsculas de minúsculas:

```
<major>.<minor>.<patch>-<platform>-<locale>-<prerelease>
```

La etiqueta siguiente es un ejemplo del formato:

```
2.0.0-amd64-en-us-preview
```

Para ver todas las configuraciones regionales admitidas del contenedor de **conversión de voz a texto**, consulte las [etiquetas de imágenes de la conversión de voz a texto](#).

## Uso del contenedor

Una vez que el contenedor esté en el [equipo host](#), utilice el siguiente proceso para trabajar con el contenedor.

1. Ejecute el contenedor con la configuración de facturación requerida. Hay más [ejemplos](#) del comando `docker run` disponibles.
2. Consulta del punto de conexión de predicción del contenedor.

### Ejecute el contenedor con `docker run`.

Utilice el comando `docker run` para ejecutar el contenedor. Consulte [Recopilación de los parámetros obligatorios](#) para más información sobre cómo obtener los valores de `{Endpoint_URI}` y `{API_Key}`. También hay disponibles otros [ejemplos](#) del comando `docker run`.

- [Voz a texto](#)
- [Conversión de voz a texto personalizada](#)
- [Texto a voz](#)
- [Conversión de texto a voz personalizada](#)

Para ejecutar el contenedor *Conversión de voz a texto*, ejecute el comando `docker run` siguiente.

```
docker run --rm -it -p 5000:5000 --memory 4g --cpus 4 \
containerpreview.azurecr.io/microsoft/cognitive-services-speech-to-text \
Eula=accept \
Billing={ENDPOINT_URI} \
ApiKey={API_KEY}
```

Este comando:

- Ejecuta un contenedor *Conversión de voz a texto* desde la imagen de contenedor.
- Asigna 4 núcleos de CPU y 4 gigabytes (GB) de memoria.
- Expone el puerto TCP 5000 y asigna un seudo-TTY para el contenedor.
- Una vez que se produce la salida, quita automáticamente el contenedor. La imagen del contenedor sigue estando disponible en el equipo host.

#### IMPORTANT

Para poder ejecutar el contenedor, las opciones `Eula`, `Billing` y `ApiKey` deben estar especificadas; de lo contrario, el contenedor no se iniciará. Para obtener más información, vea [Facturación](#).

## Consulta del punto de conexión de predicción del contenedor

CONTENEDORES	DIRECCIÓN URL DEL HOST DEL SDK	PROTOCOLO
Conversión de voz en texto y conversión de voz en texto personalizada	<code>ws://localhost:5000</code>	WS
Conversión de texto a voz y conversión de texto a voz personalizada	<code>http://localhost:5000</code>	HTTP

Para más información sobre cómo usar los protocolos WSS y HTTPS, consulte la [seguridad del contenedor](#).

### Conversión de voz a texto o Conversión de voz a texto personalizada

El contenedor proporciona las API de punto de conexión de consulta basadas en WebSocket, a las que se accede mediante el [SDK de Voz](#). De forma predeterminada, el SDK de Voz usa servicios de voz en línea. Para usar el contenedor, deberá cambiar el método de inicialización.

#### TIP

Al usar el SDK de Voz con los contenedores, no es necesario proporcionar la [clave de suscripción del recurso de Voz de Azure](#) o un [token de portador de autenticación](#).

Consulte los ejemplos siguientes.

- [C#](#)
- [Python](#)

Cambie de usar esta llamada de inicialización en la nube de Azure:

```
var config = SpeechConfig.FromSubscription("YourSubscriptionKey", "YourServiceRegion");
```

por esta llamada mediante el [host](#) de contenedor:

```
var config = SpeechConfig.FromHost(  
    new Uri("ws://localhost:5000"));
```

### Conversión de texto a voz o conversión de texto a voz personalizada

El contenedor proporciona [API de punto de conexión basadas en REST](#). Hay muchos [proyectos de código fuente de ejemplo](#) disponibles para las variaciones de lenguaje, marco y plataforma.

Con el contenedor *Conversión de texto a voz estándar*, debe basarse en la configuración regional y en la voz de la etiqueta de imagen que descargó. Por ejemplo, si descargó la etiqueta `latest`, la configuración regional predeterminada es `en-US` y la voz `JessaRUS`. El argumento `{VOICE_NAME}` sería `en-US-JessaRUS`. Vea el SSML de ejemplo siguiente:

```
<speak version="1.0" xml:lang="en-US">
    <voice name="en-US-JessaRUS">
        This text will get converted into synthesized speech.
    </voice>
</speak>
```

Sin embargo, para *Conversión de texto a voz personalizada*, tendrá que obtener el valor de **Voice / model** desde el [portal de Voz personalizada](#). El nombre del modelo personalizado es sinónimo del nombre de la voz. Vaya a la página de **entrenamiento** y copie el valor de **Voice / model** que se va a usar como el argumento `{VOICE_NAME}`.

Vea el SSML de ejemplo siguiente:

```
<speak version="1.0" xml:lang="en-US">
    <voice name="custom-voice-model">
        This text will get converted into synthesized speech.
    </voice>
</speak>
```

Vamos a crear una solicitud HTTP POST, proporcionando algunos encabezados y una carga de datos. Reemplace el marcador de posición `{VOICE_NAME}` por un valor propio.

```
curl -s -v -X POST http://localhost:5000/speech/synthesize/cognitiveservices/v1 \
-H 'Accept: audio/*' \
-H 'Content-Type: application/ssml+xml' \
-H 'X-Microsoft-OutputFormat: riff-16khz-16bit-mono-pcm' \
-d '<speak version="1.0" xml:lang="en-US"><voice name="{VOICE_NAME}">This is a test, only a test.</voice>
</speak>'
```

Este comando:

- Construye una solicitud HTTP POST para el punto de conexión `speech/synthesize/cognitiveservices/v1`.
- Especifica un encabezado `Accept` de `audio/*`
- Especifica un encabezado `Content-Type` de `application/ssml+xml`. Para más información, consulte el [cuerpo de la solicitud](#).
- Especifica un encabezado `X-Microsoft-OutputFormat` de `riff-16khz-16bit-mono-pcm`. Para más opciones, consulte la [salida de audio](#).
- Envía la solicitud [Lenguaje de marcado de síntesis de voz \(SSML\)](#) dado el `{VOICE_NAME}` al punto de conexión.

## Ejecución de varios contenedores en el mismo host

Si tiene pensado ejecutar varios contenedores con puertos expuestos, asegúrese de que ejecuta cada contenedor con un puerto expuesto diferente. Por ejemplo, ejecute el primer contenedor en el puerto 5000 y el segundo en el

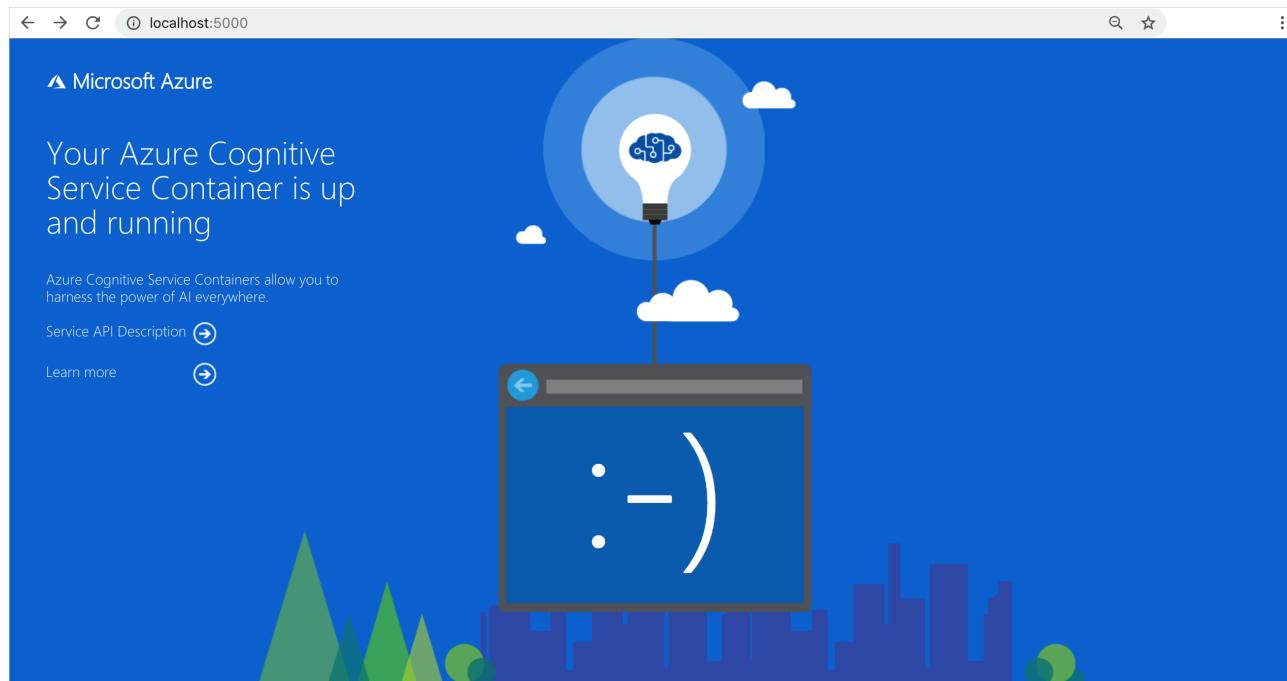
puerto 5001.

Puede tener este contenedor y un contenedor de Azure Cognitive Services diferente en ejecución simultáneamente en el HOST. También puede tener varios contenedores del mismo contenedor de Cognitive Services en ejecución.

## Comprobación de que un contenedor está en ejecución

Hay varias maneras de comprobar que el contenedor está en ejecución. Busque la dirección *IP externa* y el puerto expuesto del contenedor en cuestión y abra el explorador web que prefiera. Use las distintas direcciones URL de solicitud para validar que el contenedor se está ejecutando. Las direcciones URL de solicitud de ejemplo que se enumeran a continuación son `http://localhost:5000`, pero el contenedor específico puede variar. Tenga en cuenta que va a confiar en la *dirección IP externa* y el puerto expuesto del contenedor.

URL DE LA SOLICITUD	PROPÓSITO
<code>http://localhost:5000/</code>	El contenedor ofrece una página principal.
<code>http://localhost:5000/status</code>	Se solicitó con HTTP GET, para comprobar que el contenedor está en ejecución sin causar una consulta al punto de conexión. Esta solicitud se puede usar con los <a href="#">sondeos de ejecución y preparación</a> de Kubernetes.
<code>http://localhost:5000/swagger</code>	El contenedor cuenta con un completo conjunto de documentación sobre los puntos de conexión y una característica de <b>prueba</b> . Esta característica le permite especificar la configuración en un formulario HTML basado en web y realizar la consulta sin necesidad de escribir código. Una vez que la consulta devuelve resultados, se proporciona un ejemplo del comando CURL para mostrar los encabezados HTTP y el formato de cuerpo requeridos.



## Detención del contenedor

Para apagar el contenedor, en el entorno de la línea de comandos donde se ejecuta el contenedor, seleccione Ctrl + C.

# Solución de problemas

Puede encontrar algunos problemas al iniciar o ejecutar el contenedor. Use un [montaje](#) de salida y habilite el registro. Si lo hace, permitirá que el contenedor genere archivos de registro que son útiles para solucionar problemas.

## TIP

Para más información e instrucciones para solución de problemas, vea [Preguntas frecuentes de los contenedores de Cognitive Services](#).

## Facturación

Los contenedores de Voz envían información de facturación a Azure mediante un recurso de *Voz* en la cuenta de Azure.

Las consultas en el contenedor se facturan con el plan de tarifa del recurso de Azure que se usa para <ApiKey>.

Los contenedores de Azure Cognitive Services no tienen licencia para ejecutarse si no están conectados al punto de conexión de facturación para las mediciones. Debe habilitar los contenedores para que comuniquen la información de facturación al punto de conexión de facturación en todo momento. Los contenedores de Cognitive Services no envían datos de los clientes (por ejemplo, la imagen o el texto que se está analizando) a Microsoft.

### Conexión con Azure

El contenedor necesita que se ejecuten los valores del argumento de facturación. Estos valores permiten al contenedor conectarse al punto de conexión de facturación. El contenedor informa sobre el uso cada 10 a 15 minutos. Si el contenedor no se conecta a Azure en la ventana de tiempo permitida, continuará ejecutándose pero no atenderá las consultas hasta que se restaure el punto de conexión de facturación. Se intenta 10 veces la conexión en el mismo intervalo de tiempo de 10 a 15 minutos. Si no se puede conectar con el punto de conexión de facturación en esos 10 intentos, el contenedor deja de ejecutarse.

### Argumentos de facturación

Para que el comando `docker run` inicie el contenedor, se deben especificar las tres opciones siguientes se deben especificar con valores válidos:

OPCIÓN	DESCRIPCIÓN
<code>ApiKey</code>	La clave de API del recurso de Cognitive Services que se usa para realizar un seguimiento de la información de facturación. El valor de esta opción se debe establecer en una clave de API para el recurso aprovisionado que se especifica en <code>Billing</code> .
<code>Billing</code>	El punto de conexión del recurso de Cognitive Services que se usa para realizar el seguimiento de la información de facturación. El valor de esta opción debe establecerse en el URI del punto de conexión de un recurso aprovisionado de Azure.
<code>Eula</code>	Indica que ha aceptado la licencia del contenedor. El valor de esta opción debe establecerse en <code>accept</code> .

Para obtener más información acerca de estas opciones, consulte [Configure containers](#) (Configuración de contenedores).

## Publicaciones de blog

- Ejecución de contenedores de Cognitive Services
- Azure Cognitive Services

## Ejemplos para desarrolladores

Hay ejemplos para desarrolladores disponibles en nuestro [repositorio de GitHub](#).

## Ver seminario web

Únase al [seminario web](#) para más información sobre:

- Implementación de Cognitive Services en cualquier máquina con Docker
- Implementación de Cognitive Services en AKS

## Resumen

En este artículo, ha aprendido los conceptos y el flujo de trabajo para la descarga, instalación y ejecución de contenedores de Voz. En resumen:

- Voz proporciona cuatro contenedores Linux para Docker, que encapsulan varias funcionalidades:
  - *Voz a texto*
  - *Conversión de voz a texto personalizada*
  - *Texto a voz*
  - *Conversión de texto a voz personalizada*
- Las imágenes de contenedor se descargan desde el registro de contenedor de Azure.
- Las imágenes de contenedor se ejecutan en Docker.
- Puede usar la API REST o el SDK para llamar a operaciones en contenedores de Voz mediante la especificación del URI del host del contenedor.
- Debe proporcionar la información de facturación al crear una instancia de un contenedor.

### IMPORTANT

Los contenedores de Cognitive Services no tienen licencia para ejecutarse sin estar conectados a Azure para realizar mediciones. Los clientes tienen que habilitar los contenedores para comunicar la información de facturación con el servicio de medición en todo momento. Los contenedores de Cognitive Services no envían datos de los clientes (por ejemplo, la imagen o el texto que se está analizando) a Microsoft.

## Pasos siguientes

- Revise [Configuración de contenedores](#) para ver las opciones de configuración.
- Aprenda a [usar contenedores del servicio de voz con Kubernetes y Helm](#).
- Uso de otros [contenedores de Cognitive Services](#)

# Notas de la versión

15/01/2020 • 33 minutes to read • [Edit Online](#)

## SDK de Voz 1.8.0: Versión de noviembre de 2019

### Nuevas características

- Se ha agregado una API `FromHost()` para facilitar su uso con contenedores locales y nubes soberanas.
- Se ha agregado la detección automática de idioma de origen para el reconocimiento de voz (en Java y C++)
- Se ha agregado el objeto `SourceLanguageConfig` para el reconocimiento de voz, que se usa para especificar los idiomas de origen esperados (en Java y C++).
- Se ha agregado compatibilidad con `KeywordRecognizer` en Windows (UWP), Android e iOS mediante los paquetes de Nuget y Unity.
- Se ha agregado la API de Java de conversación remota para realizar la transcripción de conversaciones en lotes asíncronos.

### Cambios importantes

- Las funcionalidades de transcripción de conversaciones se han movido al espacio de nombres `Microsoft.CognitiveServices.Speech.Transcription`.
- Parte de los métodos de transcripción de conversaciones se han movido a la nueva clase `Conversation`.
- Compatibilidad eliminada para iOS de 32 bits (ARMv7 y x86)

### Correcciones de errores

- Se ha corregido un bloqueo si se usa `KeywordRecognizer` local sin una clave de suscripción válida al servicio de voz.

### Muestras

- Ejemplo de Xamarin para `KeywordRecognizer`
- Ejemplo de Unity para `KeywordRecognizer`
- Ejemplos de C++ y Java de detección automática de idioma de origen

## SDK de voz 1.7.0: versión de septiembre de 2019

### Nuevas características

- Compatibilidad con la versión beta agregada para Xamarin en la Plataforma universal de Windows (UWP), Android e iOS
- Compatibilidad con iOS agregada para Unity
- Se ha agregado compatibilidad con entradas `Compressed` para ALaw, Mulaw, FLAC en Android, iOS y Linux.
- Se ha agregado `SendMessageAsync` en la clase `Connection` para enviar un mensaje al servicio.
- Se ha agregado `SetMessageProperty` en la clase `Connection` para establecer la propiedad de un mensaje.
- TTS agregó compatibilidad para Java (JRE y Android), Python, Swift y Objective-C
- TTS agregó compatibilidad de reproducción para macOS, iOS y Android
- Se ha agregado información de "límite de palabras" para TTS

### Correcciones de errores

- Se ha corregido un problema de compilación de IL2CPP en Unity 2019 para Android

- Se ha corregido un problema con los encabezados con formato incorrecto en la entrada de archivo WAV que se procesa de forma incorrecta
- Se ha corregido un problema con UUID que no es único en algunas propiedades de conexión
- Se han corregido algunas advertencias sobre los especificadores de nulabilidad en los enlaces SWIFT (puede que se requieran pequeños cambios en el código)
- Se ha corregido un error que provocaba que las conexiones de WebSocket se cerraran de manera incorrecta en la carga de red
- Se ha corregido un problema en Android que a veces provoca que `DialogServiceConnector` use identificadores de impresión duplicados.
- Se han introducido mejoras en la estabilidad de las conexiones entre interacciones multiproceso y la generación de informes de errores (a través de eventos `Canceled`) cuando se producen con `DialogServiceConnector`.
- Los inicios de sesión de `DialogServiceConnector` ahora proporcionarán eventos correctamente, incluso si se llama a `ListenOnceAsync()` durante una operación `StartKeywordRecognitionAsync()` activa.
- Se ha resuelto un bloqueo asociado a la recepción de actividades `DialogServiceConnector`.

## Muestras

- Inicio rápido para Xamarin
- Se ha actualizado el inicio rápido de CPP con información de ARM64 de Linux
- Se ha actualizado el inicio rápido de Unity con información de iOS

# SDK de Voz 1.6.0: versión de junio de 2019

## Muestras

- Ejemplos de inicio rápido para Texto a voz en UWP y Unity
- Ejemplo de inicio rápido para Swift en iOS
- Ejemplos de Unity para Traducción y Reconocimiento de la intención comunicativa y Voz
- Ejemplos de inicios rápidos actualizados para `DialogServiceConnector`

## Mejoras y cambios

- Espacio de nombres de cuadro de diálogo:
  - El nombre de `SpeechBotConnector` ha cambiado a `DialogServiceConnector`
  - El nombre de `BotConfig` ha cambiado a `DialogServiceConfig`
  - `BotConfig::FromChannelSecret()` se ha reasignado a `DialogServiceConfig::FromBotSecret()`
  - Todos los clientes de Voz de Direct Line existentes siguen siendo compatibles después del cambio de nombre
- Actualización del adaptador REST de TTS para admitir una conexión persistente de proxy
- Un mejor mensaje de error cuando se pasa una región no válida
- Swift/Objective-C:
  - Mejores informes de errores: los métodos que pueden generar un error ahora se encuentran en dos versiones: una que expone un objeto `NSError` para el control de errores y una que genera una excepción. La primera se expone a Swift. Este cambio requiere adaptaciones en el código Swift existente.
  - Mejor control de eventos

## Correcciones de errores

- Corrección de TTS: donde el futuro de `SpeakTextAsync` se devolvió sin esperar al fin de la representación del audio
- Corrección para la serialización de las cadenas en C# para permitir la compatibilidad total con idiomas
- Corrección del problema de las aplicaciones centrales de .NET para cargar la biblioteca principal con un marco

de destino net461 en ejemplos

- Corrección de problemas ocasionales para implementar bibliotecas nativas en la carpeta de salida en los ejemplos
- Corrección para cerrar el socket web de manera confiable
- Corrección de un posible bloqueo al abrir una conexión con una carga muy elevada en Linux
- Corrección de metadatos faltantes en el paquete de marcos para macOS
- Corrección de problemas con `pip install --user` en Windows

## Speech SDK 1.5.1

Se trata de una versión de corrección de errores y solo afecta al SDK nativo o administrado. No afecta a la versión de JavaScript del SDK.

### Correcciones de errores

- Corrección de FromSubscription cuando se usa con la transcripción de la conversación.
- Corrección de errores en la detección de palabras clave para los asistentes por voz.

## Speech SDK 1.5.0 Versión de mayo de 2019

### Nuevas características:

- La detección de palabras clave (KWS) ahora está disponible para Windows y Linux. La funcionalidad KWS podría funcionar con cualquier tipo de micrófono; no obstante, la compatibilidad oficial de KWS está limitada actualmente a las matrices de micrófonos que se encuentran en el hardware de Azure Kinect DK o el SDK de dispositivos de voz.
- La funcionalidad de sugerencia de frases está disponible a través del SDK. Para más información, consulte [esta página](#).
- La funcionalidad de transcripción de conversaciones está disponible a través del SDK. Consulte [aquí](#).
- Compatibilidad agregada con los asistentes por voz mediante el canal Direct Line Speech.

### Muestras

- Se han agregado ejemplos para nuevas características o nuevos servicios admitidos por el SDK.

### Mejoras y cambios

- Se han agregado varias propiedades de reconocimiento para ajustar el comportamiento del servicio o los resultados del servicio (por ejemplo, enmascaramiento de palabras soeces etc.).
- Ahora puede configurar el reconocimiento a través de las propiedades de configuración estándar, incluso si ha creado el valor de `FromEndpoint` del reconocedor.
- Objective-C: se agregó la propiedad `OutputFormat` a `SPXSpeechConfiguration`.
- El SDK ahora admite Debian 9 como una distribución de Linux.

### Correcciones de errores

- Se ha corregido un problema donde el recurso de altavoz se destruía demasiado pronto en la conversión de texto a voz.

## Speech SDK 1.4.2

Se trata de una versión de corrección de errores y solo afecta al SDK nativo o administrado. No afecta a la versión de JavaScript del SDK.

## Speech SDK 1.4.1

Esta es una versión solo para JavaScript. No se agregó ninguna característica. Se realizaron las siguientes correcciones:

- Se impide que el paquete web cargue https-proxy-agent.

## Speech SDK 1.4.0 Versión de abril de 2019

### Nuevas características:

- El SDK admite ahora el servicio de conversión de texto a voz en versión beta. Se admite en Windows y Linux Desktop desde C++ y C#. Para más información, consulte la [información general sobre la conversión de texto a voz](#).
- El SDK ahora admite archivos de audio MP3 y Opus/OGG como archivos de entrada de secuencia. Esta característica solo está disponible en Linux desde C++ y C# y está actualmente en versión beta (más detalles [aquí](#)).
- Speech SDK para Java, .NET Core, C++ y Objective-C ha conseguido compatibilidad con macOS. La compatibilidad de Objective-C con macOS está actualmente en versión beta.
- iOS: Speech SDK para iOS (Objective-C) ahora también se publica como una instancia de CocoaPod.
- JavaScript: compatibilidad con micrófono no predeterminada como dispositivo de entrada.
- JavaScript: compatibilidad con servidores proxy para Node.js.

### Muestras

- se han agregado ejemplos para usar Speech SDK con C++ y con Objective-C en macOS.
- Se han agregado ejemplos que muestran el uso del servicio de conversión de texto a voz.

### Mejoras y cambios

- Python: ahora se exponen propiedades adicionales de los resultados del reconocimiento mediante la propiedad `properties`.
- Para la compatibilidad adicional con el desarrollo y la depuración, puede redirigir la información de registro y diagnóstico del SDK a un archivo de registro (más información [aquí](#)).
- JavaScript: mejora del rendimiento del procesamiento de audio.

### Correcciones de errores

- Mac/iOS: se corrigió un error que daba lugar a una larga espera cuando no se podía establecer una conexión con el servicio de voz.
- Python: mejora del control de errores en los argumentos de las devoluciones de llamada de Python.
- JavaScript: se corrigieron los informes de estado erróneos de la voz que finalizaban en RequestSession.

## Speech SDK 1.3.1 Actualización de febrero de 2019

Se trata de una versión de corrección de errores y solo afecta al SDK nativo o administrado. No afecta a la versión de JavaScript del SDK.

### Corrección de error

- Se ha corregido una fuga de memoria cuando se usa la entrada de micrófono. No afecta a la entrada de archivos o basada en secuencias.

## Speech SDK 1.3.0: versión de febrero de 2019

### Nuevas características

- El SDK de voz admite la selección del micrófono de entrada mediante la clase `AudioConfig`. Esto permite transmitir datos de audio al servicio de voz desde un micrófono no predeterminado. Para más información, consulte la documentación en la que se describe cómo [seleccionar un dispositivo de entrada de audio](#). Esta característica aún no está disponible en JavaScript.
- Speech SDK ahora es compatible con Unity en una versión beta. Proporcione sus comentarios en la sección de problemas en el [repositorio de ejemplos de GitHub](#). Esta versión es compatible con Unity en Windows x86 y x64 (aplicaciones de escritorio o de la Plataforma universal de Windows) y Android (ARM32/64, x86). Puede encontrar más información en nuestra [guía de inicio rápido sobre Unity](#).
- El archivo `Microsoft.CognitiveServices.Speech.csharp.bindings.dll` (incluido en versiones anteriores) ya no es necesario. La funcionalidad está ahora integrada en el SDK principal.

## Muestras

El siguiente contenido nuevo está disponible en nuestro [repositorio de ejemplo](#):

- Ejemplos adicionales para `AudioConfig.FromMicrophoneInput`.
- Ejemplos adicionales de Python para traducción y reconocimiento de intenciones.
- Ejemplos adicionales para usar el objeto `Connection` en iOS.
- Ejemplos adicionales de Java para la traducción con la salida de audio.
- Nuevo ejemplo de uso de la [API de REST de transcripción de lotes](#).

## Mejoras y cambios

- Python
  - Mensajes de error y verificación de parámetros mejorada en `SpeechConfig`.
  - Adición de compatibilidad para el objeto `Connection`.
  - Compatibilidad con Python (x86) de 32 bits en Windows.
  - Speech SDK para Python ya no está disponible como beta.
- iOS
  - El SDK ahora se compila en función de la versión 12.1 del SDK de iOS.
  - El SDK ahora es compatible con las versiones 9.2 y posteriores de iOS.
  - Documentación de referencia mejorada y corrección de varios nombres de propiedad.
- JavaScript
  - Adición de compatibilidad para el objeto `Connection`.
  - Archivos de definición de tipos agregados para JavaScript agrupado.
  - Compatibilidad e implementación iniciales para sugerencias de frases.
  - Colección de propiedades devuelta con JSON del servicio para reconocimiento.
- Los archivos DLL de Windows contienen ahora un recurso de versión.
- Si crea un valor de `FromEndpoint` de reconocedor, puede agregar parámetros directamente a la dirección URL del punto de conexión. Con `FromEndpoint` no puede configurar el reconocedor mediante las propiedades de configuración estándar.

## Correcciones de errores

- La contraseña de proxy y el nombre de usuario de proxy vacíos no se administraron correctamente. Con esta versión, si establece el nombre de usuario de proxy y la contraseña de proxy en una cadena vacía, no se enviarán al conectarse al proxy.
- El identificador de sesión creado por el SDK no siempre es realmente aleatorio para algunos lenguajes o entornos. Se ha agregado la inicialización del generador aleatorio para corregir este problema.
- Control mejorado del token de autorización. Si desea usar un token de autorización, especifíquelo en `SpeechConfig` y deje la clave de suscripción vacía. A continuación, cree el reconocedor como de costumbre.

- En algunos casos, el objeto `Connection` no se publicó correctamente. Ahora se ha corregido.
- Se corrigió el ejemplo de JavaScript para admitir la salida de audio para la síntesis de traducción también en Safari.

## Speech SDK 1.2.1

Esta es una versión solo para JavaScript. No se agregó ninguna característica. Se realizaron las siguientes correcciones:

- Activar el final del flujo en `turn.end`, y no en `speech.end`.
- Corregir error de la bomba de audio por el que no se programaba el siguiente envío en caso de error del envío actual.
- Corregir el reconocimiento continuo con el token de autenticación.
- Corrección de errores de diferentes reconocedores y puntos de conexión.
- Mejoras en la documentación.

## Speech SDK 1.2.0: Versión de diciembre de 2018

### Nuevas características

- Python
  - La versión beta de la compatibilidad con Python (3.5 y versiones posteriores) está disponible con esta versión. Para más información, consulte [aquí](#)(quickstart-python.md).
- JavaScript
  - Speech SDK para JavaScript ha sido de código abierto. El código fuente está disponible en [GitHub](#).
  - Ya se admite Node.js; puede encontrar más información [aquí](#).
  - Se quitó la restricción de longitud para las sesiones de audio; la reconexión se realizará automáticamente en la portada.
- Objeto `Connection`
  - Desde el objeto `Recognizer`, puede acceder al objeto `Connection`. Este objeto le permite iniciar la conexión al servicio y suscribirse para conectar y desconectar eventos explícitamente. (Esta característica no está disponible aún ni en JavaScript ni en Python).
- Compatibilidad con Ubuntu 18.04.
- Android
  - Compatibilidad con ProGuard habilitada durante la generación del APK.

### Mejoras

- Mejoras en el uso de subprocessos internos, lo que reduce el número de subprocessos, bloqueos y exclusiones mutuas.
- Se mejoraron los informes de errores y la información. En algunos casos, los mensajes de error no se propagan totalmente.
- Se actualizaron las dependencias de desarrollo en JavaScript para usar los módulos actualizados.

### Correcciones de errores

- Se han corregido las fugas de causadas por un error de coincidencia de tipos en `RecognizeAsync`.
- En algunos casos, se perdieron excepciones.
- Corrección de las fugas de memoria en los argumentos de eventos de traducción.
- Se ha corregido un problema de bloqueo al volver a conectar en sesiones de larga ejecución.
- Se ha corregido un problema que podría dar lugar a que faltase el resultado final para las traducciones con errores.

- C#: Si no se esperaba una operación `async` en el subprocesso principal, es posible que se pudiese desechar el reconocedor antes de completarse la tarea asíncrona.
- Java: Se ha corregido un problema que provocaba un bloqueo de la VM de Java.
- Objective-C: Se ha corregido la asignación de la enumeración; se devolvió `RecognizedIntent` en lugar de `RecognizingIntent`.
- JavaScript: Se ha establecido el formato de salida predeterminado en "simple" en `SpeechConfig`.
- JavaScript: Se ha quitado una incoherencia entre las propiedades del objeto de configuración en JavaScript y otros lenguajes.

## Muestras

- Se han actualizado y corregido varios ejemplos, como las voces de salida para la traducción, etc.
- Se han agregado ejemplos de Node.js en el [repositorio de ejemplo](#).

# Speech SDK 1.1.0

## Nuevas características

- Compatibilidad con Android x86/x64.
- Compatibilidad con proxy: En el objeto `SpeechConfig`, ahora puede llamar a una función para establecer la información del proxy (nombre de host, puerto, nombre de usuario y contraseña). Esta característica no está disponible aún en iOS.
- Mensajes y códigos de error mejorados. Si un reconocimiento devolvió un error, esto ya ha establecido `Reason` (en el evento cancelado) o `CancellationDetails` (en el resultado del reconocimiento) en `Error`. El evento cancelado ahora contiene dos miembros adicionales, `ErrorCode` y `ErrorDetails`. Si el servidor devolvió información de error adicional con el error notificado, ahora estará disponible en los nuevos miembros.

## Mejoras

- Verificación adicional agregada en la configuración del reconocedor y mensaje de error adicional agregado.
- Control mejorado del silencio prolongado en medio de un archivo de audio.
- Paquete NuGet: para proyectos de .NET Framework, evita la compilación con la configuración de AnyCPU.

## Correcciones de errores

- En los reconocedores se han encontrado varias excepciones corregidas. Además, las excepciones se detectan y se convierten en un evento `Canceled`.
- Corrección de una fuga de memoria en la administración de propiedades.
- Se corrigió el error en el que un archivo de entrada de audio podría bloquear el reconocedor.
- Se corrigió un error donde se podrían recibir eventos después de un evento de detención de la sesión.
- Se corrigieron algunas condiciones de subprocessos.
- Se corrigió un problema de compatibilidad de iOS que podría dar lugar a un bloqueo.
- Mejoras de estabilidad para la compatibilidad del micrófono en Android.
- Se corrigió un error donde un reconocedor en JavaScript ignoraría el lenguaje de reconocimiento.
- Se corrigió un error que impedía establecer el valor `EndpointId` (en algunos casos) en JavaScript.
- Se cambió el orden de los parámetros en `AddIntent` en JavaScript y se agregó la firma de JavaScript `AddIntent` que faltaba.

## Muestras

- Se han agregado ejemplos de C++ y C# para el uso de transmisiones de inserción y extracción en el [repositorio de ejemplos](#).

## Speech SDK 1.0.1

Mejoras en la confiabilidad y correcciones de errores:

- Corrección de un potencial error grave debido a una condición de carrera al desechar un reconocedor.
- Corrección de un potencial error grave en el caso de propiedades sin establecer.
- Comprobación adicional de errores y parámetros.
- Objective-C: corrección de posibles errores graves causados por la invalidación de nombres en NSString.
- Objective-C: ajuste de visibilidad en la API.
- JavaScript: corrección con respecto a los eventos y sus cargas.
- Mejoras en la documentación.

Se ha agregado un nuevo ejemplo de Javascript en nuestro [repositorio de ejemplos](#).

## SDK de Voz 1.0.0 de Cognitive Services: Versión de septiembre de 2018

### Nuevas características:

- Compatibilidad con Objective-C en iOS. Consulte la [Guía de inicio rápido de Objective-C para iOS](#).
- Se admite JavaScript en el explorador. Consulte la [Guía de inicio rápido de JavaScript](#).

### Cambios importantes

- Con esta versión se presentan una serie de cambios importantes. Consulte [esta página](#) para más información.

## SDK de Voz 0.6.0 de Cognitive Services: Versión de agosto de 2018

### Nuevas características:

- Ahora, las aplicaciones de UWP creadas con SDK de Voz superan el Kit para la certificación de aplicaciones en Windows (WACK). Consulte la [Guía de inicio rápido de UWP](#).
- Compatibilidad con .NET Standard 2.0 en Linux (Ubuntu 16.04 x64).
- Experimental: compatibilidad con Java 8 en Windows (64 bits) y Linux (Ubuntu 16.04 x 64). Consulte la [Guía de inicio rápido de Java Runtime Environment](#).

### Cambios funcionales

- Se expone más información detallada sobre los errores de conexión.

### Cambios importantes

- En Java (Android), la función `SpeechFactory.configureNativePlatformBindingWithDefaultCertificate` ya no requiere un parámetro de ruta de acceso. Ahora, la ruta de acceso se detecta automáticamente en todas las plataformas compatibles.
- En Java y C#, se ha quitado el descriptor de acceso get- de la propiedad `EndpointUrl`.

### Correcciones de errores

- En Java, se implementa ahora el resultado de la síntesis de audio en el reconocedor de traducción.
- Se ha corregido un error que podía provocar subprocessos inactivos y un mayor número de sockets abiertos y sin usar.
- Se ha corregido un problema por el que un proceso de reconocimiento de larga ejecución podía terminar en mitad de la transmisión.
- Se ha corregido una condición de carrera en el proceso de apagado del reconocedor.

# SDK de Voz 0.5.0 de Cognitive Services: Versión de julio de 2018

## Nuevas características:

- Compatibilidad con la plataforma Android (API 23: Android Marshmallow 6.0 o posterior). Consulte el [inicio rápido de Android](#).
- Compatibilidad con .NET Standard 2.0 en Windows. Consulte el [inicio rápido de .NET Core](#).
- Experimental: compatibilidad con UWP en Windows (versión 1709 o posterior).
  - Consulte la [Guía de inicio rápido de UWP](#).
  - Nota: Las aplicaciones de UWP creadas con el SDK de Voz no pasan aún el Kit para la certificación de aplicaciones en Windows (WACK).
- Compatibilidad con el reconocimiento de ejecución prolongada con reconexión automática.

## Cambios funcionales

- `StartContinuousRecognitionAsync()` admite reconocimiento de ejecución prolongada.
- El resultado del reconocimiento contiene más campos. Tienen un desplazamiento desde el principio del audio y la duración (ambos en tics) del texto reconocido y valores adicionales que representan el estado de reconocimiento, por ejemplo, `InitialSilenceTimeout` e `InitialBabbleTimeout`.
- Compatibilidad con `AuthorizationToken` para la creación de instancias de fábrica.

## Cambios importantes

- Eventos de reconocimiento: el tipo de evento `NoMatch` se combina con el evento `Error`.
- `SpeechOutputFormat` en C# se llama ahora `OutputFormat` para concordar con C++.
- El tipo de valor devuelto de algunos métodos de la interfaz `AudioInputStream` se ha modificado ligeramente:
  - En Java, el método `read` ahora devuelve `long` en lugar de `int`.
  - En C#, el método `Read` ahora devuelve `uint` en lugar de `int`.
  - En C++, los métodos `Read` y `GetFormat` ahora devuelven `size_t` en lugar de `int`.
- C++: las instancias de secuencias de entrada de audio ahora solo se pueden pasar como un valor `shared_ptr`.

## Correcciones de errores

- Se han corregido los valores devueltos incorrectos cuando se agota el tiempo de espera de `RecognizeAsync()`.
- Se ha eliminado la dependencia de las bibliotecas de Media Foundation en Windows. El SDK ahora usa las API de audio básicas.
- Corrección de la documentación: se ha agregado una página de [regiones](#) para describir cuáles son las regiones admitidas.

## Problema conocido

- SDK de Voz para Android no informa de los resultados de la síntesis de voz para la traducción. Este problema se solucionará en la próxima versión.

# SDK de Voz 0.4.0 de Cognitive Services: Versión de junio de 2018

## Cambios funcionales

- `AudioInputStream`

Un reconocedor ahora puede consumir una secuencia como origen de audio. Para más información, consulte la [guía de procedimientos](#) relacionada.

- Formato de salida detallado

Al crear un elemento `SpeechRecognizer`, puede solicitar el formato de salida `Detailed` o `Simple`.

`DetailedSpeechRecognitionResult` contiene una puntuación de confianza, texto reconocido, formato léxico sin formato, formato normalizado y formato normalizado con palabras soeces enmascaradas.

## Cambio importante

- En C# se cambia de `SpeechRecognitionResult.RecognizedText` a `SpeechRecognitionResult.Text`.

## Correcciones de errores

- Se ha corregido un posible problema de devolución de llamada en la capa USP durante el apagado.
- Si un reconocedor usaba un archivo de entrada de audio, mantenía el identificador de archivo más tiempo del necesario.
- Se han eliminado varios interbloqueos entre el suministro de mensajes y el reconocedor.
- Se desencadena un resultado `NoMatch` cuando se agota la respuesta del servicio.
- Las bibliotecas de Media Foundation en Windows son de carga retrasada. Esta biblioteca solo es necesaria para la entrada del micrófono.
- La velocidad de carga de los datos de audio se limita al doble de la velocidad de audio original.
- En Windows, los ensamblados .NET de C# ahora son de nombre seguro.
- Corrección de la documentación: `Region` necesita información para crear un reconocedor.

Se han agregado más ejemplos y se actualizan constantemente. Para obtener el conjunto más reciente de ejemplos, consulte el [repositorio de GitHub de ejemplos de SDK de Voz](#).

## SDK de Voz 0.2.12733 de Cognitive Services: Versión de mayo de 2018

Esta versión es la primera versión preliminar pública de SDK de Voz de Cognitive Services.

# Speech-to-text REST API

13/01/2020 • 19 minutes to read • [Edit Online](#)

Como alternativa al [SDK de Voz](#), el servicio de voz le permite convertir voz a texto mediante una API REST. Cada punto de conexión accesible se asocia con una región. La aplicación requiere una clave de suscripción para el punto de conexión que se va a usar.

Antes de usar speech-to-text API REST debe comprender que:

- Las solicitudes que usan la API REST y transmiten audio directamente solo pueden contener hasta 60 segundos de audio.
- La API REST de voz a texto solo devuelve resultados finales. No se proporcionan resultados parciales.

Si el envío de un audio más grande es necesario para la aplicación, considere la posibilidad de usar el [SDK de Voz](#) o una API REST basada en archivos, como la [transcripción por lotes](#).

## Authentication

Cada solicitud requiere un encabezado de autorización. Esta tabla muestra qué encabezados son compatibles con cada servicio:

ENCABEZADOS DE AUTORIZACIÓN COMPATIBLES	VOZ A TEXTO	TEXTO A VOZ
Ocp-Apim-Subscription-Key	Sí	Sí
Autorización: Portador	Sí	Sí

Cuando se usa el encabezado `Ocp-Apim-Subscription-Key`, solo se le pide que proporcione la clave de suscripción. Por ejemplo:

```
'Ocp-Apim-Subscription-Key': 'YOUR_SUBSCRIPTION_KEY'
```

Cuando se usa el encabezado `Authorization: Bearer`, se le pide que haga una solicitud al punto de conexión `issueToken`. En esta solicitud, va a intercambiar la clave de suscripción de un token de acceso que es válido durante 10 minutos. En las secciones siguientes obtendrá información sobre cómo obtener un token y usar un token.

### Obtención de un token de acceso

Para obtener un token de acceso, tiene que realizar una solicitud al punto de conexión `issueToken` mediante `Ocp-Apim-Subscription-Key` y su clave de suscripción.

Se admiten estas regiones y puntos de conexión:

REGION	PUNTO DE CONEXIÓN DE SERVICIO DE TOKEN
Este de Australia	<a href="https://australiaeast.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://australiaeast.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
Centro de Canadá	<a href="https://canadacentral.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://canadacentral.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
Centro de EE. UU.	<a href="https://centralus.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://centralus.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
Asia oriental	<a href="https://eastasia.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://eastasia.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
Este de EE. UU.	<a href="https://eastus.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://eastus.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
Este de EE. UU. 2	<a href="https://eastus2.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://eastus2.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
Centro de Francia	<a href="https://francecentral.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://francecentral.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
India central	<a href="https://centralindia.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://centralindia.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
Este de Japón	<a href="https://japaneast.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://japaneast.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
Corea Central	<a href="https://koreacentral.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://koreacentral.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
Centro-Norte de EE. UU.	<a href="https://northcentralus.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://northcentralus.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
Europa del Norte	<a href="https://northeurope.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://northeurope.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
Centro-Sur de EE. UU.	<a href="https://southcentralus.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://southcentralus.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>

REGION	PUNTO DE CONEXIÓN DE SERVICIO DE TOKEN
Sudeste asiático	<a href="https://southeastasia.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://southeastasia.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
Sur del Reino Unido 2	<a href="https://uksouth.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://uksouth.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
Europa occidental	<a href="https://westeurope.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://westeurope.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
Oeste de EE. UU.	<a href="https://westus.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://westus.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
Oeste de EE. UU. 2	<a href="https://westus2.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://westus2.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>

Use estos ejemplos para crear la solicitud de token de acceso.

#### Ejemplo de HTTP

Este ejemplo es una solicitud HTTP para obtener un token. Reemplace `YOUR_SUBSCRIPTION_KEY` por la clave de suscripción del servicio Voz. Si la suscripción no está en la región Oeste de EE. UU., reemplace el encabezado `Host` por el nombre de host de la región.

```
POST /sts/v1.0/issueToken HTTP/1.1
Ocp-Apim-Subscription-Key: YOUR_SUBSCRIPTION_KEY
Host: westus.api.cognitive.microsoft.com
Content-type: application/x-www-form-urlencoded
Content-Length: 0
```

El cuerpo de la respuesta contiene el token de acceso en formato JSON Web Token (JWT).

#### Ejemplo de PowerShell

En este ejemplo se muestra un script sencillo de PowerShell para obtener un token de acceso. Reemplace `YOUR_SUBSCRIPTION_KEY` por la clave de suscripción del servicio Voz. Asegúrese de usar el punto de conexión correcto para la región que coincide con su suscripción. En este ejemplo la región es Oeste de EE. UU.

```
$FetchTokenHeader = @{
    'Content-type'='application/x-www-form-urlencoded';
    'Content-Length'='0';
    'Ocp-Apim-Subscription-Key' = 'YOUR_SUBSCRIPTION_KEY'
}

$OAuthToken = Invoke-RestMethod -Method POST -Uri https://westus.api.cognitive.microsoft.com/sts/v1.0/issueToken
-Headers $FetchTokenHeader

# show the token received
$OAuthToken
```

#### Ejemplo de cURL

cURL es una herramienta de la línea de comandos disponible en Linux (y en el subsistema Windows para Linux). Este comando cURL muestra cómo obtener un token de acceso. Reemplace `YOUR_SUBSCRIPTION_KEY` por la clave de suscripción del servicio Voz. Asegúrese de usar el punto de conexión correcto para la región que coincide con su suscripción. En este ejemplo la región es Oeste de EE. UU.

```
curl -v -X POST
"https://westus.api.cognitive.microsoft.com/sts/v1.0/issueToken" \
-H "Content-type: application/x-www-form-urlencoded" \
-H "Content-Length: 0" \
-H "Ocp-Apim-Subscription-Key: YOUR_SUBSCRIPTION_KEY"
```

#### Ejemplo de C#

Esta clase de C# muestra cómo obtener un token de acceso. Pase la clave de suscripción del servicio Voz al crear una instancia de la clase. Si su suscripción no está en la región Oeste de EE. UU., cambie el valor de `FetchTokenUri` para que coincida con la región de su suscripción.

```

public class Authentication
{
    public static readonly string FetchTokenUri =
        "https://westus.api.cognitive.microsoft.com/sts/v1.0/issueToken";
    private string subscriptionKey;
    private string token;

    public Authentication(string subscriptionKey)
    {
        this.subscriptionKey = subscriptionKey;
        this.token = FetchTokenAsync(FetchTokenUri, subscriptionKey).Result;
    }

    public string GetAccessToken()
    {
        return this.token;
    }

    private async Task<string> FetchTokenAsync(string fetchUri, string subscriptionKey)
    {
        using (var client = new HttpClient())
        {
            client.DefaultRequestHeaders.Add("Ocp-Apim-Subscription-Key", subscriptionKey);
            UriBuilder uriBuilder = new UriBuilder(fetchUri);

            var result = await client.PostAsync(uriBuilder.Uri.AbsoluteUri, null);
            Console.WriteLine("Token Uri: {0}", uriBuilder.Uri.AbsoluteUri);
            return await result.Content.ReadAsStringAsync();
        }
    }
}

```

#### Ejemplo de Python

```

# Request module must be installed.
# Run pip install requests if necessary.
import requests

subscription_key = 'REPLACE_WITH_YOUR_KEY'

def get_token(subscription_key):
    fetch_token_url = 'https://westus.api.cognitive.microsoft.com/sts/v1.0/issueToken'
    headers = {
        'Ocp-Apim-Subscription-Key': subscription_key
    }
    response = requests.post(fetch_token_url, headers=headers)
    access_token = str(response.text)
    print(access_token)

```

#### Uso de un token de acceso

Se debe enviar el token de acceso al servicio como encabezado `Authorization: Bearer <TOKEN>`. Cada token de acceso tiene una validez de 10 minutos. Puede obtener un nuevo token en cualquier momento. No obstante, para reducir el tráfico de red y la latencia, se recomienda usar el mismo token durante nueve minutos.

Este es un ejemplo de solicitud HTTP a la API REST de texto a voz:

```

POST /cognitiveservices/v1 HTTP/1.1
Authorization: Bearer YOUR_ACCESS_TOKEN
Host: westus.stt.speech.microsoft.com
Content-type: application/ssml+xml
Content-Length: 199
Connection: Keep-Alive

// Message body here...

```

## Regiones y puntos de conexión

Estas regiones son compatibles con la transcripción de voz a texto mediante la API REST. Asegúrese de que selecciona el punto de conexión que coincide con la región de su suscripción.

REGION	PUNTO DE CONEXIÓN
Este de Australia	<a href="https://australiaeast.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1">https://australiaeast.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1</a>
Centro de Canadá	<a href="https://canadacentral.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1">https://canadacentral.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1</a>
Centro de EE. UU.	<a href="https://centralus.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1">https://centralus.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1</a>

REGION	PUNTO DE CONEXIÓN
Asia oriental	<a href="https://eastasia.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1">https://eastasia.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1</a>
Este de EE. UU	<a href="https://eastus.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1">https://eastus.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1</a>
Este de EE. UU. 2	<a href="https://eastus2.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1">https://eastus2.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1</a>
Centro de Francia	<a href="https://francecentral.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1">https://francecentral.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1</a>
India central	<a href="https://centralindia.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1">https://centralindia.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1</a>
Este de Japón	<a href="https://japaneast.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1">https://japaneast.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1</a>
Corea Central	<a href="https://koreacentral.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1">https://koreacentral.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1</a>
Centro-Norte de EE. UU	<a href="https://northcentralus.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1">https://northcentralus.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1</a>
Europa del Norte	<a href="https://northeurope.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1">https://northeurope.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1</a>
Centro-Sur de EE. UU	<a href="https://southcentralus.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1">https://southcentralus.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1</a>
Sudeste asiático	<a href="https://southeastasia.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1">https://southeastasia.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1</a>
Sur del Reino Unido 2	<a href="https://uksouth.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1">https://uksouth.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1</a>
Europa occidental	<a href="https://westeurope.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1">https://westeurope.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1</a>
Oeste de EE. UU.	<a href="https://westus.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1">https://westus.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1</a>
Oeste de EE. UU. 2	<a href="https://westus2.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1">https://westus2.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1</a>

#### NOTE

El parámetro de idioma debe anexarse a la dirección URL para evitar la recepción de errores HTTP 4xx. Por ejemplo, el idioma definido a inglés de Estados Unidos con el punto de conexión del Oeste de EE. UU. es:

<https://westus.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1?language=en-US>.

## Parámetros de consulta

Estos parámetros podrían incluirse en la cadena de consulta de la solicitud de REST.

PARÁMETRO	DESCRIPCIÓN	OBLIGATORIO U OPCIONAL
<code>language</code>	Identifica el idioma hablado que se está reconociendo. Vea <a href="#">Idiomas admitidos</a> .	Obligatorio
<code>format</code>	Especifica el formato del resultado. Los valores aceptados son: <code>simple</code> y <code>detailed</code> . Los resultados simples incluyen <code>RecognitionStatus</code> , <code>DisplayText</code> , <code>Offset</code> y <code>Duration</code> . Las respuestas detalladas incluyen varios resultados con valores de confianza y cuatro representaciones diferentes. La configuración predeterminada es <code>simple</code> .	Opcional
<code>profanity</code>	Especifica cómo controlar las palabras soeces en los resultados del reconocimiento. Los valores aceptados son <code>masked</code> , que reemplaza las palabras soeces con asteriscos, <code>removed</code> , que quita todas las palabras soeces del resultado o <code>raw</code> que incluye la palabra soez en el resultado. La configuración predeterminada es <code>masked</code> .	Opcional

## Encabezados de solicitud

Esta tabla enumera los encabezados obligatorios y opcionales para las solicitudes de voz a texto.

ENCABEZADO	DESCRIPCIÓN	OBLIGATORIO U OPCIONAL
<code>Ocp-Apim-Subscription-Key</code>	Clave de suscripción del servicio de voz.	Se necesita este encabezado, o bien <code>Authorization</code> .
<code>Authorization</code>	Un token de autorización precedido por la palabra <code>Bearer</code> . Para más información, consulte <a href="#">Autenticación</a> .	Se necesita este encabezado, o bien <code>Ocp-Apim-Subscription-Key</code> .
<code>Content-type</code>	Describe el formato y el códec de los datos de audio proporcionados. Los valores aceptados son: <code>audio/wav; codecs=audio/pcm; samplerate=16000</code> y <code>audio/ogg; codecs=opus</code> .	Obligatorio
<code>Transfer-Encoding</code>	Especifica que se están enviando datos de audio fragmentados en lugar de un único archivo. Use este encabezado solo si hay fragmentación de los datos de audio.	Opcional
<code>Expect</code>	Si usa la transferencia fragmentada, envíe <code>Expect: 100-continue</code> . El servicio de voz confirma la solicitud inicial y espera datos adicionales.	Obligatorio si se envían datos de audio fragmentados.
<code>Accept</code>	Si se proporciona, debe ser <code>application/json</code> . El servicio de voz proporciona resultados en JSON. Algunos marcos de solicitud proporcionan un valor predeterminado no compatible. Se recomienda incluir siempre <code>Accept</code> .	Opcional pero recomendable.

## Formatos de audio

El audio se envía en el cuerpo de la solicitud HTTP `POST`. Debe estar en uno de los formatos de esta tabla:

FORMATO	CÓDEC	BITRATE	VELOCIDAD DE MUESTREO
WAV	PCM	16 bits	16 kHz, mono
OGG	OPUS	16 bits	16 kHz, mono

### NOTE

Se admiten los formatos anteriores a través de la API REST y WebSocket en el servicio de voz. El [SDK de Voz](#) actualmente solo admite el formato WAV con el códec PCM.

## Solicitud de ejemplo

El ejemplo siguiente incluye el nombre de host y los encabezados necesarios. Es importante tener en cuenta que el servicio también espera datos de audio, que no están incluidos en este ejemplo. Como se ha mencionado anteriormente, la fragmentación es recomendable pero no obligatoria.

```
POST speech/recognition/conversation/cognitiveservices/v1?language=en-US&format=detailed HTTP/1.1
Accept: application/json;text/xml
Content-Type: audio/wav; codecs=audio/pcm; samplerate=16000
Ocp-Apim-Subscription-Key: YOUR_SUBSCRIPTION_KEY
Host: westus.stt.speech.microsoft.com
Transfer-Encoding: chunked
Expect: 100-continue
```

## Códigos de estado HTTP

El estado HTTP de cada respuesta indica estados de corrección o error comunes.

CÓDIGO DE ESTADO HTTP	DESCRIPCIÓN	POSIBLE MOTIVO
100	Continuar	Se ha aceptado la solicitud inicial. Continúe con el envío del resto de los datos. (Se usa con la transferencia fragmentada).
200	OK	La solicitud es correcta; el cuerpo de la respuesta es un objeto JSON.
400	Solicitud incorrecta	Código de idioma no proporcionado, idioma no compatible; archivo de audio no válido, etc.

CÓDIGO DE ESTADO HTTP	DESCRIPCIÓN	POSSIBLE MOTIVO
401	No autorizado	Clave de suscripción o token de autorización no válido en la región especificada, o punto de conexión no válido.
403	Prohibido	Falta la clave de suscripción o el token de autorización.

## Transferencia fragmentada

La transferencia fragmentada (`Transfer-Encoding: chunked`) puede ayudar a reducir la latencia de reconocimiento. Permite al servicio de voz empezar a procesar el archivo de audio mientras se transmite. La API de REST no proporciona resultados parciales ni provisionales.

Este ejemplo de código muestra cómo enviar audio en fragmentos. Solo el primer fragmento debe contener el encabezado del archivo de audio.

`request` es un objeto `HttpWebRequest` conectado al punto de conexión de REST adecuado. `audioFile` es la ruta de acceso a un archivo de audio en disco.

```
HttpWebRequest request = null;
request = (HttpWebRequest)HttpWebRequest.Create(requestUri);
request.SendChunked = true;
request.Accept = @"application/json;text/xml";
request.Method = "POST";
request.ProtocolVersion = HttpVersion.Version11;
request.Host = host;
request.ContentType = @"audio/wav; codecs=audio/pcm; samplerate=16000";
request.Headers["Ocp-Apim-Subscription-Key"] = args[1];
request.AllowWriteStreamBuffering = false;

using (fs = new FileStream(audioFile, FileMode.Open, FileAccess.Read))
{
    /*
     * Open a request stream and write 1024 byte chunks in the stream one at a time.
     */
    byte[] buffer = null;
    int bytesRead = 0;
    using (Stream requestStream = request.GetRequestStream())
    {
        /*
         * Read 1024 raw bytes from the input audio file.
         */
        buffer = new Byte[checked((uint)Math.Min(1024, (int)fs.Length))];
        while ((bytesRead = fs.Read(buffer, 0, buffer.Length)) != 0)
        {
            requestStream.Write(buffer, 0, bytesRead);
        }

        // Flush
        requestStream.Flush();
    }
}
```

## Parámetros de respuesta

Los resultados se proporcionan como JSON. El formato `simple` incluye los siguientes campos de nivel superior.

PARÁMETRO	DESCRIPCIÓN
<code>RecognitionStatus</code>	Estado, como <code>Success</code> , para un reconocimiento correcto. Vea la tabla siguiente.
<code>DisplayText</code>	Texto reconocido tras mayúsculas, puntuación, normalización inversa de texto (conversión de texto hablado en formularios más cortos, como 200 para "doscientos" o "Dr. Smith" para "doctor smith") y enmascaramiento de palabras soeces. Solo se presenta en caso de corrección.
<code>Offset</code>	El tiempo (en unidades de 100 nanosegundos) en el que comienza la voz reconocida en la secuencia de audio.
<code>Duration</code>	La duración (en unidades de 100 nanosegundos) de la voz reconocida en la secuencia de audio.

El campo `RecognitionStatus` puede contener estos valores:

STATUS	DESCRIPCIÓN
<code>Success</code>	El reconocimiento es correcto y el campo <code>DisplayText</code> está presente.

STATUS	DESCRIPCIÓN
NoMatch	Se detectó voz en la secuencia de audio, pero no se encontraron coincidencias de palabras en el idioma de destino. Normalmente significa que el idioma de reconocimiento es un idioma distinto al que habla el usuario.
InitialSilenceTimeout	El inicio de la secuencia de audio contiene solo silencio y el servicio ha agotado el tiempo de espera de la voz.
BabbleTimeout	El inicio de la secuencia de audio contiene solo ruido y el servicio ha agotado el tiempo de espera de la voz.
Error	El servicio de reconocimiento ha detectado un error interno y no ha podido continuar. Vuelva a intentarlo si es posible.

#### NOTE

Si el audio consta solo de palabras soeces y el parámetro de consulta `profanity` está establecido en `remove`, el servicio no devuelve ningún resultado de voz.

El formato `detailed` incluye los mismos datos que el formato `simple`, junto con `NBest`, una lista de interpretaciones alternativas del mismo resultado de reconocimiento. Estos resultados se clasifican de más probable a menos probable. La primera entrada es la misma que el resultado de reconocimiento principal. Cuando se usa el formato `detailed`, se proporciona `DisplayText` como `Display` para cada resultado de la lista `NBest`.

Cada objeto de la lista `NBest` incluye:

PARÁMETRO	DESCRIPCIÓN
<code>Confidence</code>	La puntuación de confianza de la entrada de 0,0 (ninguna confianza) a 1,0 (plena confianza)
<code>Lexical</code>	La forma léxica del texto reconocido: palabras reales reconocidas.
<code>ITN</code>	El formato de normalización inversa de texto ("canónica") del texto reconocido, con números de teléfono, números, abreviaturas ("doctor smith" a "dr smith") y otras transformaciones aplicadas.
<code>MaskedITN</code>	Formato ITN con enmascaramiento de palabras soeces aplicado, si se solicita.
<code>Display</code>	Formato de presentación del texto reconocido, con adición de signos de puntuación y mayúsculas. Este parámetro es el mismo que <code>DisplayText</code> que se proporcionó cuando el formato se estableció en <code>simple</code> .

## Respuestas de ejemplo

La siguiente es una respuesta típica de reconocimiento de `simple`:

```
{
  "RecognitionStatus": "Success",
  "DisplayText": "Remind me to buy 5 pencils.",
  "Offset": "1236645672289",
  "Duration": "1236645672289"
}
```

La siguiente es una respuesta típica de reconocimiento de `detailed`:

```
{  
    "RecognitionStatus": "Success",  
    "Offset": "1236645672289",  
    "Duration": "1236645672289",  
    "NBest": [  
        {  
            "Confidence" : "0.87",  
            "Lexical" : "remind me to buy five pencils",  
            "ITN" : "remind me to buy 5 pencils",  
            "MaskedITN" : "remind me to buy 5 pencils",  
            "Display" : "Remind me to buy 5 pencils.",  
        },  
        {  
            "Confidence" : "0.54",  
            "Lexical" : "rewind me to buy five pencils",  
            "ITN" : "rewind me to buy 5 pencils",  
            "MaskedITN" : "rewind me to buy 5 pencils",  
            "Display" : "Rewind me to buy 5 pencils.",  
        }  
    ]  
}
```

## Pasos siguientes

- [Obtenga su suscripción de prueba a Voz](#)
- [Personalización de modelos acústicos](#)
- [Personalización de modelos de lenguaje](#)

# ¿Por qué usar la transcripción de lotes?

13/01/2020 • 15 minutes to read • [Edit Online](#)

La transcripción de lotes es ideal si desea transcribir una gran cantidad de audio en el almacenamiento, como los blobs de Azure. Mediante la API REST dedicada, puede apuntar a archivos de audio con un identificador URI de firma de acceso compartido (SAS) y recibir las transcripciones de forma asíncrona.

## Requisitos previos

### Clave de suscripción

Como sucede con todas las características del servicio Voz, puede crear una clave de suscripción en [Azure Portal](#) siguiendo nuestra [guía de inicio](#). Si va a obtener transcripciones de nuestros modelos de base de referencia, todo lo que necesita hacer es crear una clave.

#### NOTE

Se requiere una suscripción estándar (S0) para el servicio de voz para usar la transcripción de lotes. Las claves de suscripción gratuita (F0) no funcionarán. Para más información, consulte [Precios y límites](#).

### Modelos personalizados

Si tiene previsto personalizar modelos acústicos o de lenguaje, siga los pasos que se indican en [Creación de modelos acústicos](#) y [Personalización de modelos de lenguaje](#). Para usar los modelos creados en la transcripción por lotes, necesita sus identificadores de modelo. Este identificador no es el mismo que el del punto de conexión que se encuentra en la vista de detalles del punto de conexión, sino el identificador del modelo que puede recuperar al seleccionar los detalles de los modelos.

## Transcription API de Azure Batch

Transcription API de Batch ofrece la transcripción asíncrona de voz a texto, además de otras características. Es una API REST que expone métodos para:

1. Crear solicitudes de procesamiento por lotes
2. Consultar el estado
3. Descargar transcripciones

#### NOTE

Esta API resulta muy adecuada para centros de llamadas que suelen acumular miles de horas de audio. Facilita la transcripción de grandes volúmenes de grabaciones de audio.

### Formatos compatibles

Transcription API de Batch admite los siguientes formatos:

FORMATO	CÓDEC	BITRATE	VELOCIDAD DE MUESTREO
WAV	PCM	16 bits	8 o 16 kHz, mono, estéreo
MP3	PCM	16 bits	8 o 16 kHz, mono, estéreo

FORMATO	CÓDEC	BITRATE	VELOCIDAD DE MUESTREO
OGG	OPUS	16 bits	8 o 16 kHz, mono, estéreo

Si se trata de secuencias de audio estéreo, Batch Transcription API divide los canales izquierdo y derecho durante la transcripción. Los dos archivos JSON con el resultado se crean desde un único canal. Las marcas de tiempo por expresión permiten al desarrollador crear una transcripción final ordenada. Esta solicitud de ejemplo incluye las propiedades de filtrado de blasfemias, signos de puntuación y marcas de tiempo de nivel de palabra.

## Configuración

Los parámetros de configuración se proporcionan como JSON:

```
{
  "recordingsUrl": "<URL to the Azure blob to transcribe>",
  "models": [{"Id": "<optional acoustic model ID>"}, {"Id": "<optional language model ID>"}],
  "locale": "<locale to use, for example en-US>",
  "name": "<user defined name of the transcription batch>",
  "description": "<optional description of the transcription>",
  "properties": {
    "ProfanityFilterMode": "Masked",
    "PunctuationMode": "DictatedAndAutomatic",
    "AddWordLevelTimestamps" : "True",
    "AddSentiment" : "True"
  }
}
```

### NOTE

Transcription API de Batch usa un servicio REST para solicitar transcripciones, su estado y los resultados asociados. Puede usar la API con cualquier lenguaje. En la siguiente sección se describe cómo se usa la API.

## Propiedades de configuración

Utilice estas propiedades opcionales para configurar la transcripción:

PARÁMETRO	DESCRIPCIÓN
<code>ProfanityFilterMode</code>	Especifica cómo controlar las palabras soeces en los resultados del reconocimiento. Los valores aceptados son <code>None</code> , que deshabilita el filtrado de palabras soeces; <code>masked</code> , que reemplaza las palabras soeces con asteriscos; <code>removed</code> , que quita todas las palabras soeces del resultado; o <code>tags</code> , que agrega etiquetas de "palabras soeces". La configuración predeterminada es <code>masked</code> .
<code>PunctuationMode</code>	Especifica cómo controlar la puntuación en los resultados del reconocimiento. Los valores aceptados son: <code>None</code> , que deshabilita la puntuación; <code>dictated</code> , que implica puntuación explícita; <code>automatic</code> , que permite que el decodificador se ocupe de la puntuación; o <code>dictatedandautomatic</code> , que implica signos de puntuación dictados o puntuación automática.

PARÁMETRO	DESCRIPCIÓN
AddWordLevelTimestamps	Especifica si las marcas de tiempo de nivel de palabra se deben agregar a la salida. Los valores aceptados son <code>true</code> , que permite las marcas de tiempo de nivel de palabra, y <code>false</code> (el valor predeterminado) que las deshabilita.
AddSentiment	Especifica que la opinión se debe agregar a la expresión. Los valores aceptados son <code>true</code> , que permite la opinión por expresión y <code>false</code> (el valor predeterminado) que la deshabilita.
AddDiarization	Especifica que el análisis de la diarización se debe llevar a cabo en la entrada que se espera sea un canal mono que contenga dos voces. Los valores aceptados son <code>true</code> , que permite la diarización, y <code>false</code> (el valor predeterminado) que la deshabilita. También requiere que <code>AddWordLevelTimestamps</code> se establezca en <code>true</code> .

## Storage

La transcripción de lotes admite [Azure Blob Storage](#) para leer el audio y escribir las transcripciones en el almacenamiento.

## Separación de altavoces (diarización)

La diarización es el proceso de separación de los altavoces en una parte del audio. Nuestra canalización de Batch admite la diarización y es capaz de reconocer dos altavoces en las grabaciones de un canal mono.

Para pedir que la solicitud de transcripción de audio se procese para la diarización, simplemente tiene que agregar el parámetro pertinente en la solicitud HTTP como se muestra a continuación.

```
{
  "recordingsUrl": "<URL to the Azure blob to transcribe>",
  "models": [{"Id": "<optional acoustic model ID>"}, {"Id": "<optional language model ID>"}],
  "locale": "<locale to us, for example en-US>",
  "name": "<user defined name of the transcription batch>",
  "description": "<optional description of the transcription>",
  "properties": {
    "AddWordLevelTimestamps" : "True",
    "AddDiarization" : "True"
  }
}
```

Las marcas de tiempo de nivel de palabra también tendría que estar "activadas" como indican los parámetros de la solicitud anterior.

El audio correspondiente contendrá los altavoces identificados por un número (actualmente se admiten solo dos voces, por lo que los altavoces se identificarán como "Altavoz 1" y "Altavoz 2") seguidos de la salida de la transcripción.

Tenga en cuenta también que la diarización no está disponible en las grabaciones estéreo. Además, toda la salida JSON contendrá la etiqueta Speaker. Si no se usa diarización, mostrará "Speaker: Null" en la salida JSON.

**NOTE**

La diarización está disponible en todas las regiones y para todas las configuraciones regionales!

## Opinión

Opinión es una nueva característica de Transcription API de Batch, que es importante en el dominio del centro de llamadas. Los clientes pueden usar los parámetros `AddSentiment` en sus solicitudes para

1. Obtener información sobre la satisfacción del cliente
2. Obtener información sobre el rendimiento de los agentes (equipo que atiende las llamadas)
3. Identificar el punto exacto en el tiempo de cuándo una llamada giró hacia una dirección negativa
4. Identificar qué salió bien al convertir las llamadas negativas en positivas
5. Identificar lo que les gusta a los clientes y lo que no les gusta de un producto o servicio

Opinión se puntúa por segmento de audio, donde un segmento de audio se define como el lapso de tiempo entre el inicio de la expresión (desplazamiento) y el silencio de detección del fin del flujo de bytes. Todo el texto dentro de ese segmento se utiliza para calcular la opinión. NO se calcula ningún valor agregado de opinión para toda la llamada o la voz completa de cada canal. Estas agregaciones se dejan al propietario del dominio para seguir aplicándolas.

Opinión se aplica en la forma léxica.

Un ejemplo de salida JSON tiene el siguiente aspecto:

```
{  
    "AudioFileResults": [  
        {  
            "AudioFileName": "Channel.0.wav",  
            "AudioFileUrl": null,  
            "SegmentResults": [  
                {  
                    "RecognitionStatus": "Success",  
                    "ChannelNumber": null,  
                    "Offset": 400000,  
                    "Duration": 13300000,  
                    "NBest": [  
                        {  
                            "Confidence": 0.976174,  
                            "Lexical": "what's the weather like",  
                            "ITN": "what's the weather like",  
                            "MaskedITN": "what's the weather like",  
                            "Display": "What's the weather like?",  
                            "Words": null,  
                            "Sentiment": {  
                                "Negative": 0.206194,  
                                "Neutral": 0.793785,  
                                "Positive": 0.0  
                            }  
                        }  
                    ]  
                }  
            ]  
        }  
    ]  
}
```

La característica utiliza un modelo de Opinión, que se encuentra actualmente en versión Beta.

## Código de ejemplo

Hay ejemplos completos disponibles en el [repositorio de ejemplos de GitHub](#) dentro del subdirectorio `samples/batch`.

Si quiere usar un modelo acústico o de lenguaje personalizado, deberá personalizar el código de ejemplo con la información de suscripción, la región del servicio, el URI de SAS que apunta al archivo de audio que se va a transcribir y los identificadores del modelo.

```
// Replace with your subscription key
private const string SubscriptionKey = "YourSubscriptionKey";

// Update with your service region
private const string Region = "YourServiceRegion";
private const int Port = 443;

// recordings and locale
private const string Locale = "en-US";
private const string RecordingsBlobUri = "<SAS URI pointing to an audio file stored in Azure Blob Storage>";

// For usage of baseline models, no acoustic and language model needs to be specified.
private static Guid[] modelList = new Guid[0];

// For use of specific acoustic and language models:
// - comment the previous line
// - uncomment the next lines to create an array containing the guids of your required model(s)
// private static Guid AdaptedAcousticId = new Guid("<id of the custom acoustic model>");
// private static Guid AdaptedLanguageId = new Guid("<id of the custom language model>");
// private static Guid[] modelList = new[] { AdaptedAcousticId, AdaptedLanguageId };

//name and description
private const string Name = "Simple transcription";
private const string Description = "Simple transcription description";
```

El código de ejemplo configurará el cliente y enviará la solicitud de transcripción. A continuación, sondeará la información de estado e imprimirá los detalles sobre el progreso de la transcripción.

```

// get all transcriptions for the user
transcriptions = await client.GetTranscriptionsAsync().ConfigureAwait(false);

completed = 0; running = 0; notStarted = 0;
// for each transcription in the list we check the status
foreach (var transcription in transcriptions)
{
    switch (transcription.Status)
    {
        case "Failed":
        case "Succeeded":
            // we check to see if it was one of the transcriptions we created from this client.
            if (!createdTranscriptions.Contains(transcription.Id))
            {
                // not created from here, continue
                continue;
            }
            completed++;

        // if the transcription was successful, check the results
        if (transcription.Status == "Succeeded")
        {
            var resultsUri0 = transcription.ResultsUrls["channel_0"];

            WebClient webClient = new WebClient();

            var filename = Path.GetTempFileName();
            webClient.DownloadFile(resultsUri0, filename);
            var results0 = File.ReadAllText(filename);
            var resultObject0 = JsonConvert.DeserializeObject<RootObject>(results0);
            Console.WriteLine(results0);

            Console.WriteLine("Transcription succeeded. Results: ");
            Console.WriteLine(results0);
        }
        else
        {
            Console.WriteLine("Transcription failed. Status: {0}", transcription.StatusMessage);
        }
        break;

        case "Running":
            running++;
            break;

        case "NotStarted":
            notStarted++;
            break;
    }
}

```

Para más información sobre las llamadas anteriores, consulte nuestro [documento de Swagger](#). Para ver el ejemplo completo aquí mostrado, vaya a [GitHub](#) en el subdirectorio `samples/batch`.

Tome nota de la configuración asincrónica para publicar audio y recibir el estado de la transcripción. El cliente que crea es un cliente HTTP de .NET. Hay un método `PostTranscriptions` para enviar los detalles del archivo de audio y un método `GetTranscriptions` para recibir los resultados. `PostTranscriptions` devuelve un identificador y `GetTranscriptions` lo usa para crear un identificador y obtener el estado de la transcripción.

El ejemplo de código actual no especifica un modelo personalizado. El servicio usa los modelos base de referencia para transcribir los archivos. Para especificar los modelos, puede pasar los mismos identificadores de modelo para el modelo acústico y de lenguaje.

**NOTE**

En el caso de transcripciones de base de referencia, no es necesario declarar los identificadores de los modelos de base de referencia. Si solo especifica un identificador de modelo de lenguaje (y ningún identificador de modelo acústico), se selecciona automáticamente un modelo acústico correspondiente. Si solo especifica un identificador de modelo acústico, se selecciona automáticamente un modelo de lenguaje correspondiente.

## Descarga del ejemplo

Puede encontrar el ejemplo en el directorio `samples/batch` en el [repositorio de ejemplos de GitHub](#).

**NOTE**

Los trabajos de transcripción por lotes están programados de la mejor manera posible; no hay una estimación de tiempo de cuándo un trabajo cambiará al estado en ejecución. En este estado, la transcripción real se procesa más rápido que el audio en tiempo real.

## Pasos siguientes

- [Obtenga su suscripción de prueba a Voz](#)

# Preguntas más frecuentes sobre el servicio Speech to Text

13/01/2020 • 18 minutes to read • [Edit Online](#)

Si no encuentra respuestas a sus preguntas en estas P+F, consulte otras opciones de soporte técnico.

## General

**P: ¿Qué diferencia hay entre un modelo de línea base y un modelo personalizado de conversión de voz en texto?**

**R.** : Un modelo de línea base se ha entrenado con datos propiedad de Microsoft y ya está implementado en la nube. Puede usar un modelo personalizado con el fin de adaptar un modelo para que se ajuste mejor a un entorno concreto que tenga ruido ambiental o lenguaje específicos. Fábricas, coches o calles ruidosas requerirán un modelo acústico adaptado. Temas como la biología, la física, la radiología, los nombres de productos y los acrónimos personalizados requerirán un modelo de lenguaje adaptado.

**P: ¿Por dónde empiezo si quiero usar un modelo de línea base?**

**R.** : Primero, obtenga una [clave suscripción](#). Si quiere realizar llamadas REST a los modelos de línea de base implementados previamente, vea las [API REST](#). Si quiere usar WebSockets, [descargue el SDK](#).

**P: ¿Siempre es necesario crear un modelo de voz personalizado?**

**R.** : No. Si en la aplicación se usa un lenguaje cotidiano genérico, no es necesario personalizar un modelo. Si la aplicación se usa en un entorno con poco o ningún ruido de fondo, tampoco es necesario personalizar un modelo.

Puede implementar modelos de línea de base y personalizados en el portal, y después ejecutar pruebas de precisión en ellos. Puede usar esta característica para medir la precisión de un modelo de línea de base con respecto a uno personalizado.

**P: ¿Cómo sabré que el procesamiento del conjunto de datos o modelo se ha completado?**

**R.** : Actualmente, el estado del modelo o del conjunto de datos en la tabla es la única manera de saberlo. Cuando se complete el procesamiento, el estado será **Succeeded** (Correcto).

**P: ¿Puedo crear más de un modelo?**

**R.** : No hay límite en cuanto al número de modelos que puede tener en la colección.

**P: Me he dado cuenta de que me he equivocado. ¿Cómo cancelo la importación de datos o la creación del modelo que está en curso?**

**R.** : Actualmente no se puede revertir un proceso de adaptación acústica o de lenguaje. Puede eliminar los modelos y los datos importados cuando estén en un estado terminal.

**P: ¿Qué diferencia existe entre el modelo de búsqueda y dictado y el modelo de conversación?**

**R.** : Puede elegir entre varios modelos de línea base en el servicio Voz. El modelo Conversational es adecuado para el reconocimiento de la voz hablada en un estilo conversacional. Este modelo es ideal para la transcripción de llamadas de teléfono. El modelo Search and Dictation es ideal para aplicaciones desencadenadas por la voz. El modelo Universal es un modelo nuevo con el objetivo de abordar ambos escenarios. El modelo Universal tiene actualmente un nivel de calidad igual o superior al del modelo de conversación en la mayoría de las configuraciones regionales.

**P: ¿Puedo actualizar el modelo existente (apilamiento del modelo)?**

**R.** : Un modelo existente no se puede actualizar. Como solución alternativa, puede combinar el conjunto de datos anterior con el nuevo y readaptarlo.

El conjunto de datos antiguo y el nuevo se deben combinar en un único archivo ZIP (para datos acústicos) o en un archivo .txt (para datos de lenguaje). Cuando termine la adaptación, se debe volver a implementar el nuevo modelo actualizado para obtener un punto de conexión nuevo.

**P: Cuando hay disponible una versión nueva de una línea base, ¿la implementación se actualiza de forma automática?**

**R.** : Las implementaciones NO se actualizan de forma automática.

Si ha adaptado e implementado un modelo con la línea de base V1.0, esa implementación permanecerá tal cual. Los clientes pueden retirar el modelo implementado, volver a adaptar con la versión más reciente de la línea de base y volver a implementar.

**P: ¿Puedo descargar mi modelo y ejecutarlo localmente?**

**R.** : Los modelos no se pueden descargar ni ejecutar localmente.

**P: ¿Se registran mis solicitudes?**

**R.** : Al crear una implementación, tiene una opción para desactivar el seguimiento. Después, no se registrará ningún sonido ni ninguna transcripción. De lo contrario, las solicitudes se registran normalmente en Azure, en almacenamiento seguro.

**P: ¿Están limitadas mis solicitudes?**

**R.** : La API REST limita las solicitudes a 25 cada 5 segundos. Encontrará los detalles en nuestras páginas de [Speech to Text](#).

**P: ¿Cómo se cobra el audio de canal doble?**

**R.** : Si envía cada canal por separado (cada canal en su propio archivo), se le cobrará según la duración de cada archivo. Si envía un solo archivo con cada canal multiplexado juntos, se le cobrará por la duración del archivo individual.

**IMPORTANT**

Si tiene más dudas sobre la privacidad que le impidan usar el servicio Voz personalizado, póngase en contacto con uno de los canales de soporte técnico.

## Aumento de la simultaneidad

**P: ¿Qué sucede si necesito una mayor simultaneidad para el modelo implementado de la que se ofrece en el portal?**

**R.** : Se puede escalar verticalmente el modelo en incrementos de 20 solicitudes simultáneas.

Con la información necesaria, cree una solicitud de soporte técnico en el [portal de soporte técnico de Azure](#). No publique la información en ninguno de los canales públicos (GitHub, Stackoverflow, etc.) que se mencionan en la [página de soporte técnico](#).

Para aumentar la simultaneidad de un **modelo personalizado**, se necesita la siguiente información:

- La región donde se implementa el modelo,
- el identificador del punto de conexión del modelo implementado:

- Inicie sesión en el [portal de Custom Speech](#),
- inicie sesión (si es necesario),
- seleccione el proyecto y la implementación,
- seleccione el punto de conexión para el que necesita el aumento de simultaneidad,
- copie el `Endpoint ID`.

Para aumentar la simultaneidad de un **modelo base**, se necesita la siguiente información:

- La región del servicio

y bien

- un token de acceso para la suscripción (consulte [aquí](#)),

or

- el identificador del recurso de la suscripción:
  - Vaya a [Azure Portal](#),
  - seleccione `Cognitive Services` en el cuadro de búsqueda,
  - en los servicios mostrados, seleccione el servicio de voz para el que desea aumentar la simultaneidad,
  - muestre `Properties` de este servicio,
  - copie el `Resource ID` completo.

## Importación de datos

**P: ¿Cuál es el límite de tamaño de un conjunto de datos, y por qué existe?**

**R. :** El límite actual de un conjunto de datos es de 2 GB. El límite se debe a la restricción del tamaño de un archivo para la carga HTTP.

**P: ¿Puedo comprimir mis archivos de texto para cargar un archivo de texto mayor?**

**R. :** No. Actualmente solo se permiten los archivos de texto no comprimidos.

**P: El informe de datos indica que ha habido expresiones erróneas. ¿Cuál es el problema?**

**R. :** No es un problema que no se pueda cargar el 100 % de las expresiones de un archivo. Si la gran mayoría de las expresiones de un conjunto de datos acústicos o de lenguaje (por ejemplo, más del 95 %) se importan correctamente, el conjunto de datos se podrá usar. Pero se recomienda comprender la causa del error de las expresiones y corregir los problemas. Los problemas más comunes, como los errores de formato, son fáciles de corregir.

## Creación de un modelo acústico

**P: ¿Cuántos datos acústicos necesito?**

**R. :** Se recomienda empezar con 30 minutos a una hora de datos acústicos.

**P: ¿Qué datos debo recopilar?**

**R. :** Recopile datos lo más cercanos posibles al escenario de aplicación y caso de uso. La colección de datos debe coincidir con la aplicación y los usuarios de destino en términos de dispositivo o dispositivos, entornos y tipos de hablante. En general, debe recopilar datos de un intervalo lo más amplio posible de hablantes.

**P: ¿Cómo debo recopilar los datos acústicos?**

**R. :** Puede crear una aplicación de recopilación de datos autónoma o usar software de grabación de audio comercial. También puede crear una versión de la aplicación que registre los datos de audio y después los use.

**P: ¿Debo transcribir los datos de adaptación yo mismo?**

**R.** : Sí. Puede transcribirlos usted mismo o utilizar un servicio de transcripción profesional. Algunos usuarios prefieren usar transcriptores profesionales, mientras que otros usan la colaboración abierta distribuida o realizan las transcripciones ellos mismos.

## Pruebas de precisión

**P: ¿Puedo realizar una prueba sin conexión de mi modelo acústico personalizado con un modelo de lenguaje personalizado?**

**R.** : Sí, basta con seleccionar el modelo de lenguaje personalizado en el menú desplegable al configurar la prueba sin conexión.

**P: ¿Puedo realizar una prueba sin conexión de mi modelo de lenguaje personalizado con un modelo acústico personalizado?**

**R.** : Sí, basta con seleccionar el modelo acústico personalizado en el menú desplegable al configurar la prueba sin conexión.

**P: ¿Qué es Word Error Rate (WER) y cómo se calcula?**

**R.** : WER es la métrica de evaluación para el reconocimiento de voz. WER se cuenta como el número total de errores, lo que incluye las inserciones, eliminaciones y sustituciones, dividido por el número total de palabras en la transcripción de referencia. Para obtener más información, vea [word error rate](#).

**P: ¿Cómo determino si los resultados de una prueba de precisión son correctos?**

**R.** : Los resultados muestran una comparación entre el modelo de línea base y el personalizado. Debe aspirar a superar el modelo de línea de base para que la personalización sea útil.

**P: ¿Cómo puedo determinar el valor WER de un modelo base para ver si se produjo una mejora?**

**R.** : Los resultados de la prueba sin conexión muestran la precisión de línea base del modelo personalizado y la mejora sobre la línea base.

## Creación de un modelo de lenguaje

**P: ¿Cuántos datos de texto tengo que cargar?**

**R.** : Depende de la diferencia que exista entre el vocabulario y las frases que se usan en la aplicación y los modelos de lenguaje iniciales. Para todas las palabras nuevas, es útil proporcionar el máximo de ejemplos posible de utilización de estas palabras. Para las frases comunes que se usan en la aplicación, también es útil incluir frases en los datos de lenguaje, ya que indican al sistema que escuche también estos términos. Es habitual que haya al menos 100, y normalmente varios cientos de expresiones en el conjunto de datos de lenguaje o más. Además, si se espera que algunos tipos de consultas sean más habituales que otros, puede insertar varias copias de las consultas comunes en el conjunto de datos.

**P: ¿Puedo simplemente cargar una lista de palabras?**

**R.** : La carga de una lista de palabras las agregará al vocabulario, pero no enseñará al sistema cómo se usan normalmente. Al proporcionar expresiones completas o parciales (oraciones o frases que es probable que digan los usuarios), el modelo de lenguaje puede aprender las palabras nuevas y cómo se usan. El modelo de lenguaje personalizado es bueno no solo para agregar palabras nuevas al sistema, sino también para ajustar la probabilidad de palabras conocidas para la aplicación. Al proporcionar expresiones completas se ayuda al sistema a aprender mejor.

## Modelo de inquilino (Custom Speech con datos de Office 365)

**P: ¿Qué información se incluye en el modelo de inquilino y cómo se crea?**

**R:** Un modelo de inquilino se crea con documentos y correos electrónicos de [grupos públicos](#) que puede ver cualquier usuario de la organización.

**P: ¿Qué experiencias de voz mejora el modelo de inquilino?**

**R:** Cuando el modelo de inquilino se habilita, se crea y se publica, se usa para mejorar el reconocimiento de las aplicaciones empresariales compiladas con el servicio de voz, que también pasan un token de AAD de usuario que indica la pertenencia a la empresa.

Las experiencias de voz integradas en Office 365, como el Dictado y los subtítulos de PowerPoint, no cambian cuando se crea un modelo de inquilino para las aplicaciones del servicio de voz.

## Pasos siguientes

- [Solución de problemas](#)
- [Notas de la versión](#)

# ¿Qué es el texto a voz?

15/01/2020 • 7 minutes to read • [Edit Online](#)

La característica de texto a voz del servicio de voz permite que sus aplicaciones, herramientas o dispositivos conviertan el texto en una voz sintetizada similar a la humana. Puede elegir entre voces estándar y neuronales, o puede crear una voz personalizada única para su producto o marca. Tiene más de 75 voces estándar disponibles en más de 45 idiomas y configuraciones regionales, y 5 voces neuronales que están disponibles en varios idiomas y configuraciones regionales. Para obtener una lista completa de las voces, los idiomas y las configuraciones regionales compatibles, consulte [Idiomas admitidos](#).

## NOTE

Bing Speech se ha retirado el 15 de octubre de 2019. Si sus aplicaciones, herramientas o productos usan Bing Speech API o Custom Speech, hemos creado guías para que le ayuden a migrar al servicio de voz.

- [Migración de Bing Speech al servicio de voz](#)

## Características principales

- Síntesis de voz: use el [SDK de voz](#) o la [API de REST](#) para convertir texto a voz mediante las voces estándar, neuronal o personalizada.
- Síntesis asincrónica de audio de larga duración: use [Long Audio API](#) para sintetizar asincrónicamente archivos de texto a voz de más de 10 minutos (por ejemplo, audiolibros o audioconferencias). A diferencia de la síntesis realizada mediante el SDK de voz o la API de REST de voz a texto, las respuestas no se devuelven en tiempo real. La expectativa es que las solicitudes se envíen de forma asincrónica, se sondeen las respuestas y el audio sintetizado se descargue cuando esté disponible en el servicio. Solo se admiten las voces neuronales.
- Voces estándar: se crean mediante técnicas de síntesis paramétrica estadística y de síntesis de concatenación. Estas voces son realmente inteligibles y suenan muy naturales. Puede habilitar fácilmente sus aplicaciones para que hablen en más de 45 idiomas, con una amplia gama de opciones de voz. Estas voces proporcionan una alta precisión de pronunciación, admiten abreviaturas, expanden acrónimos, interpretan la fecha y la hora, son polifónicas y ofrecen muchas cosas más. Hay una lista completa de voces estándar en [Idiomas admitidos](#).
- Voces neuronales: las redes neuronales profundas se usan para superar los límites de la síntesis de voz tradicional con respecto al acento y la entonación del lenguaje hablado. La predicción de la prosodia y la síntesis de voz se realizan simultáneamente, lo que resulta en una voz más fluida y natural. Las voces neuronales se pueden usar para que las interacciones con los bots de chat y los asistentes de voz sean más naturales y atractivas, para convertir textos digitales (por ejemplo, los libros electrónicos) en audiolibros y para mejorar los sistemas de navegación de los automóviles. Gracias a su prosodia natural similar a la humana y a la clara articulación de las palabras, las voces neuronales reducen considerablemente la fatiga al escucharlas que suele aparecer cuando los usuarios interactúan con sistemas de inteligencia artificial. Hay una lista completa de voces neuronales en [Idiomas admitidos](#).
- Lenguaje de marcado de síntesis de voz (SSML): lenguaje de marcado basado en XML que se usa para personalizar las salidas de voz a texto. Con SSML, puede ajustar el tono, agregar pausas, mejorar la pronunciación, acelerar o ralentizar la velocidad del habla, subir o bajar el volumen y atribuir varias voces a un solo documento. Consulte [SSML](#).

# Introducción

El servicio de texto a voz está disponible a través del [SDK de voz](#). Existen varios escenarios comunes disponibles como inicios rápidos, en diferentes lenguajes y plataformas:

- [Síntesis de voz en un archivo de audio](#)
- [Síntesis de voz en un altavoz](#)
- [Síntesis asincrónica de audio de formato largo](#)

Si lo prefiere, el servicio de texto a voz es accesible a través de [REST](#).

## Código de ejemplo

El ejemplo de código para texto a voz está disponible en GitHub. Estos ejemplos tratan la conversión de texto a voz en los lenguajes de programación más populares.

- [Ejemplos de conversión de texto a voz \(SDK\)](#)
- [Ejemplos de texto a voz \(REST\)](#)

## Personalización

Además de las voces estándar y neuronales, puede crear y ajustar las voces personalizadas exclusivas del producto o la marca. Todo lo que se necesita para empezar son unos cuantos archivos de audio y las transcripciones asociadas. Para obtener información, consulte [Introducción a voz personalizada](#).

## Nota de precios

Al usar el servicio de texto a voz, se le facturará por cada carácter que se convierte a voz, incluida la puntuación. Si bien el documento SSML en sí no es facturable, los elementos opcionales que se usan para ajustar el modo de convertir el texto a voz, como los fonemas y el tono, se cuentan como caracteres facturables. Aquí tiene una lista de lo que se puede facturar:

- El texto que se ha pasado al servicio de texto a voz en el cuerpo SSML de la solicitud.
- Todas las marcas en el campo de texto del cuerpo de la solicitud que están en formato SSML, excepto las etiquetas `<speak>` y `<voice>`.
- Letras, puntuación, espacios, tabulaciones, marcas y todos los caracteres de espacios en blanco.
- Cada punto de código que se define en Unicode

Para obtener más información, consulte [Precios](#).

### IMPORTANT

Cada carácter en chino, japonés y coreano se cuenta como dos caracteres para la facturación.

## Documentos de referencia

- [Acerca del SDK de Voz](#)
- [API REST: Text-to-speech](#) (API de REST: Texto a voz)

## Pasos siguientes

- [Obtención de una suscripción de gratuita al servicio de voz](#)
- [Obtención del SDK de voz](#)

# Inicio rápido: Síntesis de voz en un archivo de audio

15/01/2020 • 23 minutes to read • [Edit Online](#)

En este inicio rápido, usará el SDK de [Voz](#) para convertir texto en voz sintetizada en un archivo de audio. Una vez que se cumplen los requisitos previos, la síntesis de voz en un archivo solo necesita realizar estos cinco pasos:

- Cree un objeto `SpeechConfig` a partir de la clave y la región de suscripción.
- Cree un objeto de configuración de audio que especifique el nombre del archivo .WAV.
- Cree un objeto `SpeechSynthesizer` con los objetos de configuración anteriores.
- Con el objeto `SpeechSynthesizer`, convierta el texto en voz sintetizada y guárdelo en el archivo de audio especificado.
- Inspeccione el objeto `SpeechSynthesizer` devuelto en caso de errores.

Si prefiere ponerse a trabajar de inmediato, vea o descargue todos los [ejemplos para C# del SDK de Voz](#) en GitHub. Si no, vamos a comenzar.

## Prerequisites

Antes de comenzar, compruebe lo siguiente:

- [Ha creado un recurso de Voz de Azure](#)
- [Ha configurado el entorno de desarrollo](#)
- [Ha creado un proyecto de ejemplo vacío](#)

## Abra el proyecto en Visual Studio.

El primer paso es asegurarse de que tiene el proyecto abierto en Visual Studio.

1. Inicie Visual Studio 2019.
2. Cargue el proyecto y abra `Program.cs`.

## Inicio con código reutilizable

Vamos a agregar código que funcione como el esqueleto del proyecto. Tenga en cuenta que ha creado un método asíncrono llamado `SynthesisToFileAsync()`.

```

using System;
using System.Threading.Tasks;
using Microsoft.CognitiveServices.Speech;

namespace helloworld
{
    class Program
    {
        public static async Task SynthesisToAudioFileAsync()
        {

        }

        static void Main()
        {
            SynthesisToAudioFileAsync().Wait();
        }
    }
}

```

## Creación de una configuración de Voz

Antes de inicializar un objeto `SpeechSynthesizer`, debe crear una configuración que use la clave y la región de suscripción. Inserte este código en el método `SynthesisToAudioFileAsync()`.

```
var config = SpeechConfig.FromSubscription("YourSubscriptionKey", "YourServiceRegion");
```

## Creación de una configuración de audio

Ahora, debe crear un objeto `AudioConfig` que apunte al archivo de audio. Este objeto se crea dentro de una instrucción `using` para garantizar la versión correcta de los recursos no administrados. Inserte este código en el método `SynthesisToAudioFileAsync()`, justo debajo de la configuración de Voz.

```

var fileName = "helloworld.wav";
using (var fileOutput = AudioConfig.FromWavFileOutput(fileName))
{
}

```

## Inicialización de un `SpeechSynthesizer`

Ahora, se creará el objeto `SpeechSynthesizer` con los objetos `SpeechConfig` y `AudioConfig` creados anteriormente. Este objeto se crea también dentro de una instrucción `using` para garantizar la versión correcta de los recursos no administrados. Inserte este código en el método `SynthesisToAudioFileAsync()`, dentro de la instrucción `using` que encapsula el objeto `AudioConfig`.

```

using (var synthesizer = new SpeechSynthesizer(config, fileOutput))
{
}

```

## Síntesis de texto mediante `SpeakTextAsync`

En el objeto `SpeechSynthesizer`, va a llamar al método `SpeakTextAsync()`. Este método envía el texto al servicio de voz, que lo convierte en audio. El `SpeechSynthesizer` utilizará la voz predeterminada si `config.VoiceName` no

se especifica explícitamente.

Dentro de la instrucción using, agregue este código:

```
var text = "Hello world!";
var result = await synthesizer.SpeakTextAsync(text);
```

## Comprobación de errores

Cuando el servicio de voz devuelve el resultado de la síntesis, debe comprobar que el texto se ha sintetizado correctamente.

Dentro de la instrucción using, debajo de `SpeakTextAsync()`, agregue este código:

```
if (result.Reason == ResultReason.SynthesizingAudioCompleted)
{
    Console.WriteLine($"Speech synthesized to [{fileName}] for text [{text}]");
}
else if (result.Reason == ResultReason.Canceled)
{
    var cancellation = SpeechSynthesisCancellationDetails.FromResult(result);
    Console.WriteLine($"CANCELED: Reason={cancellation.Reason}");

    if (cancellation.Reason == CancellationReason.Error)
    {
        Console.WriteLine($"CANCELED: ErrorCode={cancellation.ErrorCode}");
        Console.WriteLine($"CANCELED: ErrorDetails=[{cancellation.ErrorDetails}]");
        Console.WriteLine($"CANCELED: Did you update the subscription info?");
    }
}
```

## Comprobación del código

En este momento, el código debe tener esta apariencia:

```

// Copyright (c) Microsoft. All rights reserved.
// Licensed under the MIT license. See LICENSE.md file in the project root for full license information.
//

using System;
using System.Threading.Tasks;
using Microsoft.CognitiveServices.Speech;

namespace helloworld
{
    class Program
    {
        public static async Task SynthesisToAudioFileAsync()
        {
            var config = SpeechConfig.FromSubscription("YourSubscriptionKey", "YourServiceRegion");

            var fileName = "helloworld.wav";
            using (var fileOutput = AudioConfig.FromWavFileOutput(fileName))
            {
                using (var synthesizer = new SpeechSynthesizer(config, fileOutput))
                {
                    var text = "Hello world!";
                    var result = await synthesizer.SpeakTextAsync(text);

                    if (result.Reason == ResultReason.SynthesizingAudioCompleted)
                    {
                        Console.WriteLine($"Speech synthesized to [{fileName}] for text [{text}]");
                    }
                    else if (result.Reason == ResultReason.Canceled)
                    {
                        var cancellation = SpeechSynthesisCancellationDetails.FromResult(result);
                        Console.WriteLine($"CANCELED: Reason={cancellation.Reason}");

                        if (cancellation.Reason == CancellationReason.Error)
                        {
                            Console.WriteLine($"CANCELED: ErrorCode={cancellation.ErrorCode}");
                            Console.WriteLine($"CANCELED: ErrorDetails=[{cancellation.ErrorDetails}]");
                            Console.WriteLine($"CANCELED: Did you update the subscription info?");
                        }
                    }
                }
            }

            static void Main()
            {
                SynthesisToAudioFileAsync().Wait();
            }
        }
    }
}

```

## Compilación y ejecución de la aplicación

Ya está listo para compilar la aplicación y probar la síntesis de voz con el servicio Voz.

- Compile el código:** en la barra de menús de Visual Studio, elija **Compilar > Compilar solución**.
- Inicie la aplicación:** en la barra de menús, elija **Depurar > Iniciar depuración** o presione **F5**.
- Iniciar síntesis:** el texto se convierte en voz y se guarda en los datos de audio especificados.

Speech synthesized to [helloworld.wav] for text [Hello world!]

# Pasos siguientes

[Exploración de ejemplos de C# en GitHub](#)

## Consulte también

- [Creación de una voz personalizada](#)
- [Grabación de ejemplos de voz personalizada](#)

---

En este inicio rápido, usará el SDK de [Voz](#) para convertir texto en voz sintetizada en un archivo de audio. Una vez que se cumplen los requisitos previos, la síntesis de voz en un archivo solo necesita realizar estos cinco pasos:

- Cree un objeto `SpeechConfig` a partir de la clave y la región de suscripción.
- Cree un objeto de configuración de audio que especifique el nombre del archivo .WAV.
- Cree un objeto `SpeechSynthesizer` con los objetos de configuración anteriores.
- Con el objeto `SpeechSynthesizer`, convierta el texto en voz sintetizada y guárdelo en el archivo de audio especificado.
- Inspeccione el objeto `SpeechSynthesizer` devuelto en caso de errores.

Si prefiere ponerse a trabajar de inmediato, vea o descargue todos los [ejemplos para C++ del SDK de Voz](#) en GitHub. Si no, vamos a comenzar.

## Prerequisites

Antes de comenzar, compruebe lo siguiente:

- [Ha creado un recurso de Voz de Azure](#)
- [Ha configurado el entorno de desarrollo](#)
- [Ha creado un proyecto de ejemplo vacío](#)

## Incorporación de código de ejemplo

1. Abra el archivo de origen **helloworld.cpp**.
2. Reemplace todo el código por el fragmento siguiente:

```

// Creates an instance of a speech config with specified subscription key and service region.
// Replace with your own subscription key and service region (e.g., "westus").
auto config = SpeechConfig::FromSubscription("YourSubscriptionKey", "YourServiceRegion");

// Creates a speech synthesizer using file as audio output.
// Replace with your own audio file name.
auto fileName = "helloworld.wav";
auto fileOutput = AudioConfig::FromWavFileOutput(fileName);
auto synthesizer = SpeechSynthesizer::FromConfig(config, fileOutput);

// Converts the specified text to speech, saving the audio data in the file specified above.
// Replace with your own text.
auto text = "Hello world!";
auto result = synthesizer->SpeakTextAsync(text).get();

// Checks result for successful completion or errors.
if (result->Reason == ResultReason::SynthesizingAudioCompleted)
{
    cout << "Speech synthesized to [" << fileName << "] for text [" << text << "]" << std::endl;
}
else if (result->Reason == ResultReason::Canceled)
{
    auto cancellation = SpeechSynthesisCancellationDetails::FromResult(result);
    cout << "CANCELED: Reason=" << (int)cancellation->Reason << std::endl;

    if (cancellation->Reason == CancellationReason::Error)
    {
        cout << "CANCELED: ErrorCode=" << (int)cancellation->ErrorCode << std::endl;
        cout << "CANCELED: ErrorDetails=[" << cancellation->ErrorDetails << "]" << std::endl;
        cout << "CANCELED: Did you update the subscription info?" << std::endl;
    }
}

```

3. En el mismo archivo, reemplace la cadena `YourSubscriptionKey` por la clave de suscripción.
4. Reemplace la cadena `YourServiceRegion` por la [región](#) asociada a sus suscripción (por ejemplo, `westus` para la suscripción de evaluación gratuita).
5. Reemplace la cadena `helloworld.wav` por su propio nombre de archivo.
6. En la barra de menús, elija **Archivo > Guardar todo**.

## Compilación y ejecución de la aplicación

1. En la barra de menús, seleccione **Compilar > Compilar solución** para compilar la aplicación. El código se debería compilar sin errores ahora.
2. Elija **Depurar > Iniciar depuración** o presione **F5** para iniciar la aplicación **HelloWorld**.
3. El texto se convierte en voz y se guarda en los datos de audio especificados.

Speech synthesized to [helloworld.wav] for text [Hello world!]

## Pasos siguientes

[Exploración de ejemplos de C++ en GitHub](#)

## Consulte también

- [Creación de una voz personalizada](#)
- [Grabación de ejemplos de voz personalizada](#)

En este inicio rápido, usará el SDK de [Voz](#) para convertir texto en voz sintetizada en un archivo de audio. Una vez que se cumplen los requisitos previos, la síntesis de voz en un archivo solo necesita realizar estos cinco pasos:

- Cree un objeto `SpeechConfig` a partir de la clave y la región de suscripción.
- Cree un objeto de configuración de audio que especifique el nombre del archivo .WAV.
- Cree un objeto `SpeechSynthesizer` con los objetos de configuración anteriores.
- Con el objeto `SpeechSynthesizer`, convierta el texto en voz sintetizada y guárdelo en el archivo de audio especificado.
- Inspeccione el objeto `SpeechSynthesizer` devuelto en caso de errores.

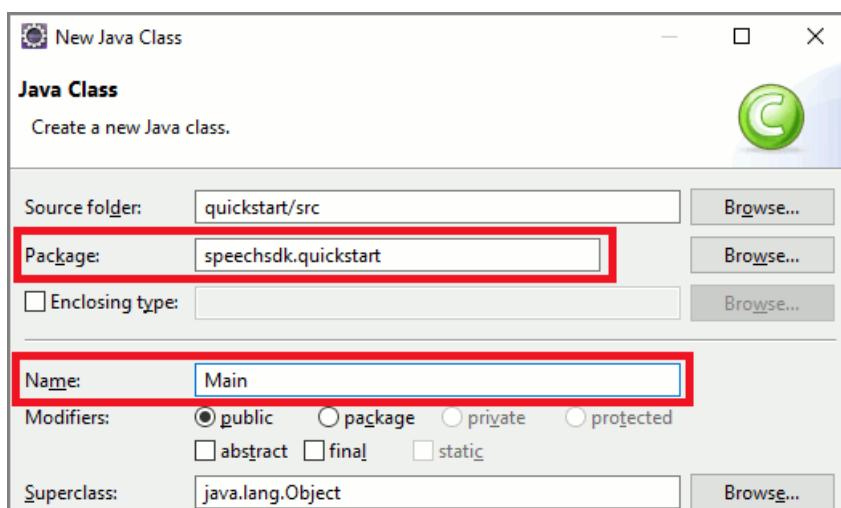
Si prefiere ponerse a trabajar de inmediato, vea o descargue todos los [ejemplos para Java del SDK de Voz](#) en GitHub. Si no, vamos a comenzar.

## Requisitos previos

- [Ha creado un recurso de Voz de Azure](#)
- [Ha configurado el entorno de desarrollo](#)
- [Ha creado un proyecto de ejemplo vacío](#)

## Incorporación de código de ejemplo

1. Para agregar una nueva clase vacía al proyecto de Java, seleccione **File (Archivo) > New (Nuevo) > Class (Clase)**.
2. En la ventana **New Java Class** (Nueva clase de Java) escriba **speechsdk.quickstart** en el campo **Package (Paquete)** y **Main** en el campo **Name (Nombre)**.



3. Reemplace el código en `Main.java` con el siguiente fragmento de código:

```
package speechsdk.quickstart;

import java.util.concurrent.Future;
import com.microsoft.cognitiveservices.speech.*;

/**
 * Quickstart: recognize speech using the Speech SDK for Java.
 */
public class Main {
```

```

/**
 * @param args Arguments are ignored in this sample.
 */
public static void main(String[] args) {
    try {
        // Replace below with your own subscription key
        String speechSubscriptionKey = "YourSubscriptionKey";
        // Replace below with your own service region (e.g., "westus").
        String serviceRegion = "YourServiceRegion";
        // Replace below with your own filename.
        String audioFileName = "helloworld.wav";
        // Replace below with your own filename.
        String text = "Hello world!";

        int exitCode = 1;
        SpeechConfig config = SpeechConfig.fromSubscription(speechSubscriptionKey, serviceRegion);
        assert(config != null);

        AudioConfig audioOutput = AudioConfig.fromWavFileInput(audioFileName);
        assert(audioOutput != null);

        SpeechSynthesizer synth = new SpeechSynthesizer(config, audioOutput);
        assert(synth != null);

        Future<SpeechSynthesisResult> task = synth.SpeakTextAsync(text);
        assert(task != null);

        SpeechSynthesisResult result = task.get();
        assert(result != null);

        if (result.getReason() == ResultReason.SynthesizingAudioCompleted) {
            System.out.println("Speech synthesized to [" + audioFileName + "] for text [" + text +
                "]");
            exitCode = 0;
        }
        else if (result.getReason() == ResultReason.Canceled) {
            SpeechSynthesisCancellationDetails cancellation =
                SpeechSynthesisCancellationDetails.fromResult(result);
            System.out.println("CANCELED: Reason=" + cancellation.getReason());

            if (cancellation.getReason() == CancellationReason.Error) {
                System.out.println("CANCELED: ErrorCode=" + cancellation.getErrorCode());
                System.out.println("CANCELED: ErrorDetails=" + cancellation.getErrorDetails());
                System.out.println("CANCELED: Did you update the subscription info?");
            }
        }

        result.close();
        synth.close();

        System.exit(exitCode);
    } catch (Exception ex) {
        System.out.println("Unexpected exception: " + ex.getMessage());

        assert(false);
        System.exit(1);
    }
}
}

```

4. Reemplace la cadena `YourSubscriptionKey` por la clave de suscripción.
5. Reemplace la cadena `YourServiceRegion` por la [región](#) asociada a sus suscripción (por ejemplo, `westus` para la suscripción de evaluación gratuita).
6. Reemplace la cadena `helloworld.wav` por su propio nombre de archivo.

7. Reemplace la cadena `Hello world!` por su propio texto.

8. Guarde los cambios en el proyecto.

## Compilación y ejecución de la aplicación

Presione F11, o seleccione **Run (Ejecutar) > Debug (Depurar)**. El texto se convierte en voz y se guarda en los datos de audio especificados.

```
Speech synthesized to [helloworld.wav] for text [Hello world!]
```

## Pasos siguientes

[Exploración de ejemplos de Java en GitHub](#)

## Otras referencias

- [Creación de una voz personalizada](#)
- [Grabación de ejemplos de voz personalizada](#)

En este inicio rápido, usará el SDK de [Voz](#) para convertir texto en voz sintetizada en un archivo de audio. Una vez que se cumplen los requisitos previos, la síntesis de voz en un archivo solo necesita realizar estos cinco pasos:

- Cree un objeto `SpeechConfig` a partir de la clave y la región de suscripción.
- Cree un objeto de configuración de audio que especifique el nombre del archivo .WAV.
- Cree un objeto `SpeechSynthesizer` con los objetos de configuración anteriores.
- Con el objeto `SpeechSynthesizer`, convierta el texto en voz sintetizada y guárdelo en el archivo de audio especificado.
- Inspeccione el objeto `SpeechSynthesizer` devuelto en caso de errores.

Si prefiere ponerse a trabajar de inmediato, vea o descargue todos los [ejemplos para Python del SDK de Voz](#) en GitHub. Si no, vamos a comenzar.

## Prerequisites

- Una clave de suscripción de Azure para el servicio Voz. [Obtenga una gratis](#).
- [Python 3.5 o versiones posteriores](#).
- El paquete del SDK de Voz de Python está disponible para estos sistemas operativos:
  - Windows: x64 y x86.
  - Mac: macOS X versión 10.12 o posterior.
  - Linux: Ubuntu 16.04, Ubuntu 18.04, Debian 9 en x64.
- En Linux, ejecute estos comandos para instalar los paquetes necesarios:
  - En Ubuntu:

```
sudo apt-get update
sudo apt-get install build-essential libssl1.0.0 libasound2
```

- En Debian 9:

```
sudo apt-get update  
sudo apt-get install build-essential libssl1.0.2 libasound2
```

- En Windows, necesita [Microsoft Visual C++ Redistributable para Visual Studio 2019](#) para su plataforma.

## Instalación de Speech SDK

### IMPORTANT

Al descargar cualquiera de los componentes del SDK de Voz de Azure Cognitive Services de esta página, acepta su licencia. Consulte los [términos de licencia del software de Microsoft para el SDK de Voz](#).

Este comando instala el paquete de Python desde [PyPI](#) para el SDK de Voz:

```
pip install azure-cognitiveservices-speech
```

## Soporte técnico y actualizaciones

Las actualizaciones del paquete de Python del SDK de Voz se distribuirán mediante PyPI y se anunciarán en la página [Notas de la versión](#). Si hay disponible una nueva versión, puede actualizarse a ella con el comando `pip install --upgrade azure-cognitiveservices-speech`. Para comprobar qué versión está instalada actualmente, inspeccione la variable `azure.cognitiveservices.speech._version_`.

Si tiene un problema o falta una característica, consulte las [opciones de ayuda y soporte técnico](#).

## Creación de una aplicación de Python mediante el SDK de Voz

### Ejecución del ejemplo

Puede copiar el [código de ejemplo](#) de este inicio rápido en un archivo de código fuente `quickstart.py` y ejecutarlo en el IDE o en la consola:

```
python quickstart.py
```

También, puede descargar este tutorial de inicio rápido como un cuaderno de [Jupyter](#) del [repositorio de ejemplos del SDK de Voz](#) y ejecutarlo como un cuaderno.

### Código de ejemplo

```

import azure.cognitiveservices.speech as speechsdk

# Creates an instance of a speech config with specified subscription key and service region.
# Replace with your own subscription key and service region (e.g., "westus").
speech_key, service_region = "YourSubscriptionKey", "YourServiceRegion"
speech_config = speechsdk.SpeechConfig(subscription=speech_key, region=service_region)

# Creates an audio configuration that points to an audio file.
# Replace with your own audio filename.
audio_filename = "helloworld.wav"
audio_output = speechsdk.AudioOutputConfig(filename=audio_filename)

# Creates a synthesizer with the given settings
speech_synthesizer = speechsdk.SpeechSynthesizer(speech_config=speech_config, audio_config=audio_output)

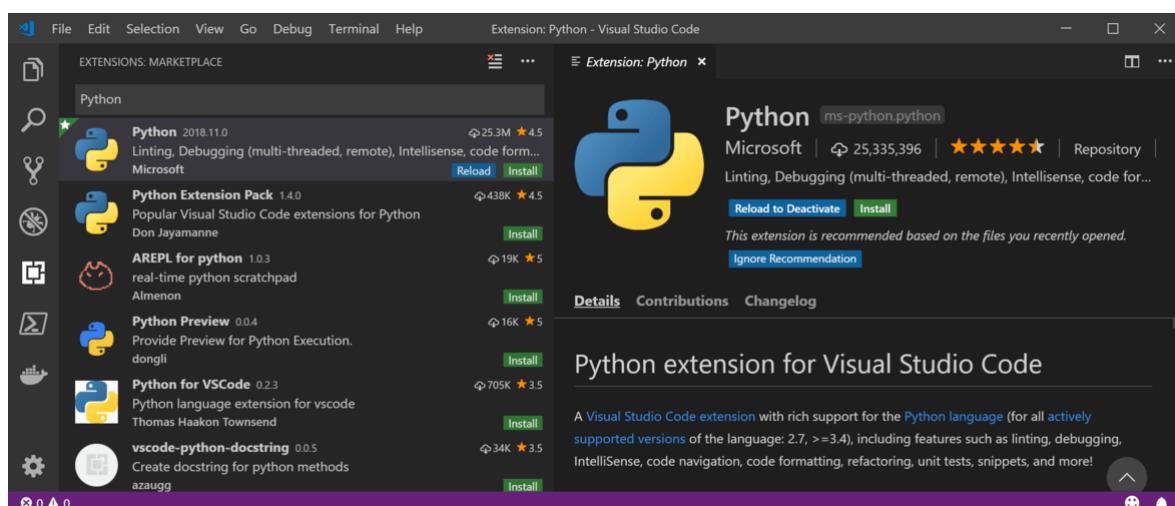
# Synthesizes the text to speech.
# Replace with your own text.
text = "Hello world!"
result = speech_synthesizer.speak_text_async(text).get()

# Checks result.
if result.reason == speechsdk.ResultReason.SynthesizingAudioCompleted:
    print("Speech synthesized to [{}]. for text [{}].".format(audio_filename, text))
elif result.reason == speechsdk.ResultReason.Canceled:
    cancellation_details = result.cancellation_details
    print("Speech synthesis canceled: {}.".format(cancellation_details.reason))
    if cancellation_details.reason == speechsdk.CancellationReason.Error:
        if cancellation_details.error_details:
            print("Error details: {}.".format(cancellation_details.error_details))
print("Did you update the subscription info?")

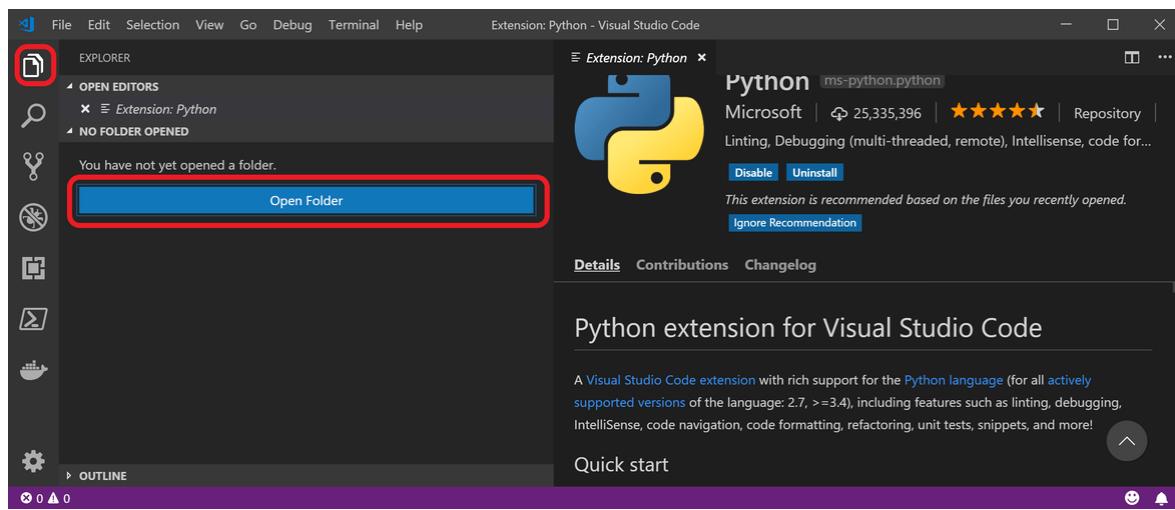
```

## Instalación y uso del SDK de Voz con Visual Studio Code

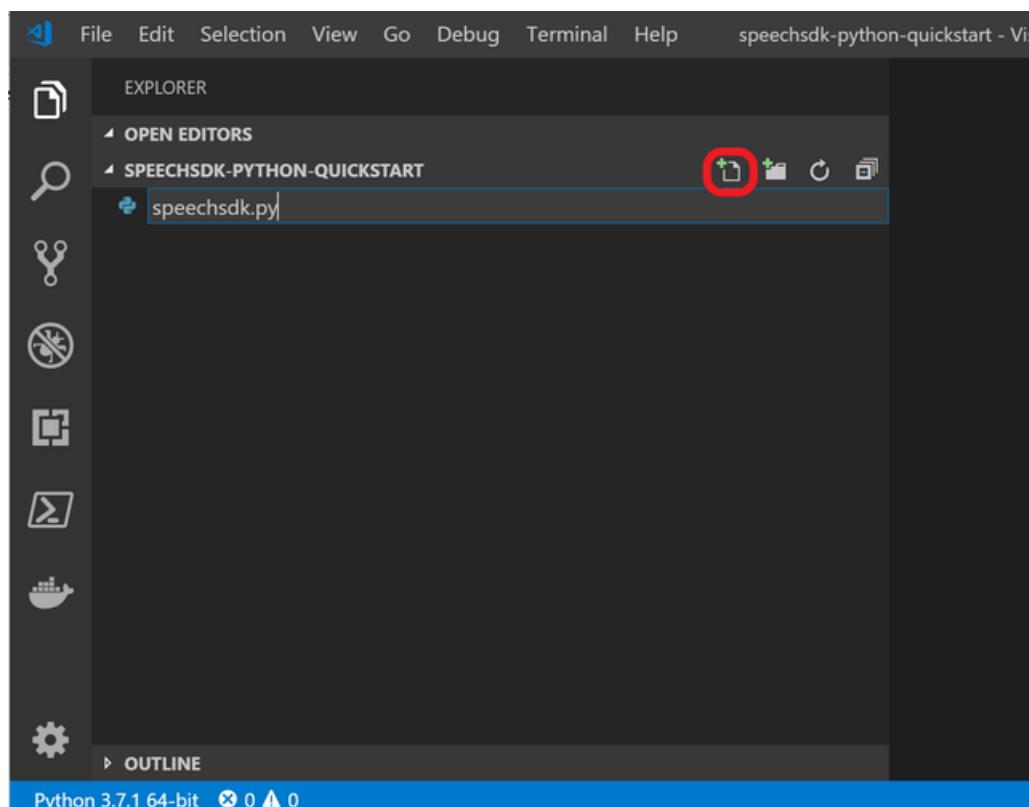
1. Descargue e instale una versión de 64 bits (3.5 o posterior) de [Python](#) en el equipo.
2. Descargue e instale [Visual Studio Code](#).
3. Abra Visual Studio Code e instale la extensión de Python. Seleccione **File > Preferences > Extensions** (Archivo > Preferencias > Extensiones) en el menú. Busque **Python**.



4. Cree una carpeta en la que almacenar el proyecto. Por ejemplo, puede usar para ello el Explorador de Windows.
5. En Visual Studio Code, seleccione el ícono de **File** (Archivo). A continuación, abra la carpeta que creó.



6. Cree un archivo de código fuente de Python `speechsdk.py` mediante la selección del ícono de nuevo archivo.



7. Copie, pegue y guarde el [código de Python](#) en el archivo recién creado.
8. Inserte la información de la suscripción del servicio Voz.
9. Si se selecciona, se muestra un intérprete de Python en el lado izquierdo de la barra de estado en la parte inferior de la ventana. En caso contrario, aparecerá una lista de los intérpretes de Python disponibles. Abra la paleta de comandos (Ctrl+Mayús+P) y escriba **Python: Select Interpreter** (Seleccionar intérprete). Elija un valor apropiado.
10. Puede instalar el paquete de Python del SDK de Voz desde dentro de Visual Studio Code. Hágalo si no está instalado aún para el intérprete de Python seleccionado. Para instalar el paquete del SDK de Voz, abra un terminal. Abra de nuevo la paleta de comandos (Ctrl+Mayús+P) y escriba **Terminal: Create New Integrated Terminal** (Crear terminal integrado). En el terminal que se abre, escriba el comando `python -m pip install azure-cognitiveservices-speech` o el que sea apropiado para su sistema.
11. Para ejecutar el código de ejemplo, haga clic con el botón derecho en algún lugar dentro del editor. Seleccione **Run Python File in Terminal** (Ejecutar archivo de Python en terminal). El texto se convierte

en voz y se guarda en los datos de audio especificados.

```
Speech synthesized to [helloworld.wav] for text [Hello world!]
```

Si tiene problemas para seguir estas instrucciones, consulte el [tutorial de Python para Visual Studio Code](#) con información más amplia.

## Pasos siguientes

[Exploración de los ejemplos de Python en GitHub](#)

## Consulte también

- [Creación de una voz personalizada](#)
- [Grabación de ejemplos de voz personalizada](#)

Vea o descargue todos los [ejemplos del SDK de Voz](#) en GitHub.

## Compatibilidad con plataformas y lenguajes adicionales

Si ha hecho clic en esta pestaña, es probable que no vea un inicio rápido en su lenguaje de programación favorito. No se preocupe, tenemos materiales de inicio rápido y ejemplos de código adicionales disponibles en GitHub. Use la tabla para encontrar el ejemplo correcto para su lenguaje de programación y combinación de plataforma y sistema operativo.

IDIOMA	INICIOS RÁPIDOS ADICIONALES	EJEMPLOS DE CÓDIGO
C++		<a href="#">Windows</a> , <a href="#">Linux</a> , <a href="#">macOS</a>
C#		<a href="#">.NET Framework</a> , <a href="#">.NET Core</a> , <a href="#">UWP</a> , <a href="#">Unity</a> , <a href="#">Xamarin</a>
Java		<a href="#">Android</a> , <a href="#">JRE</a>
JavaScript		<a href="#">Browser</a>
Node.js		<a href="#">Windows</a> , <a href="#">Linux</a> , <a href="#">macOS</a>
Objective-C	<a href="#">macOS</a> , <a href="#">iOS</a>	<a href="#">iOS</a> , <a href="#">macOS</a>
Python		<a href="#">Windows</a> , <a href="#">Linux</a> , <a href="#">macOS</a>
Swift	<a href="#">macOS</a> , <a href="#">iOS</a>	<a href="#">iOS</a> , <a href="#">macOS</a>

# Inicio rápido: Síntesis de voz en un altavoz

16/01/2020 • 47 minutes to read • [Edit Online](#)

En este inicio rápido, usará el [SDK de Voz](#) para convertir texto en voz sintetizada. Una vez que se cumplen los requisitos previos, para la representación de voz sintetizada en los altavoces predeterminados solo son necesarios cuatro pasos:

- Cree un objeto `SpeechConfig` a partir de la clave y la región de suscripción.
- Cree un objeto `SpeechSynthesizer` con el objeto `SpeechConfig` anterior.
- Use el objeto `SpeechSynthesizer` para pronunciar el texto.
- Compruebe el objeto `SpeechSynthesisResult` devuelto en caso de errores.

Si prefiere ponerse a trabajar de inmediato, vea o descargue todos los [ejemplos para C# del SDK de Voz](#) en GitHub. Si no, vamos a comenzar.

## Selección del entorno de destino

- [.NET](#)
- [.NET Core](#)
- [Unity](#)
- [UWP](#)

## Requisitos previos

Antes de comenzar, compruebe lo siguiente:

- [Ha creado un recurso de Voz de Azure](#)
- [Ha configurado el entorno de desarrollo](#)
- [Ha creado un proyecto de ejemplo vacío](#)

## Incorporación de código de ejemplo

1. Abra **Program.cs** y reemplace el código generado automáticamente por el de este ejemplo:

```

using System;
using System.Threading.Tasks;
using Microsoft.CognitiveServices.Speech;

namespace helloworld
{
    class Program
    {
        public static async Task SynthesisToSpeakerAsync()
        {
            // Creates an instance of a speech config with specified subscription key and service
            region.

            // Replace with your own subscription key and service region (e.g., "westus").
            // The default language is "en-us".
            var config = SpeechConfig.FromSubscription("YourSubscriptionKey", "YourServiceRegion");

            // Creates a speech synthesizer using the default speaker as audio output.
            using (var synthesizer = new SpeechSynthesizer(config))
            {
                // Receive a text from console input and synthesize it to speaker.
                Console.WriteLine("Type some text that you want to speak...");
                Console.Write("> ");
                string text = Console.ReadLine();

                using (var result = await synthesizer.SpeakTextAsync(text))
                {
                    if (result.Reason == ResultReason.SynthesizingAudioCompleted)
                    {
                        Console.WriteLine($"Speech synthesized to speaker for text [{text}]");
                    }
                    else if (result.Reason == ResultReason.Canceled)
                    {
                        var cancellation = SpeechSynthesisCancellationDetails.FromResult(result);
                        Console.WriteLine($"CANCELED: Reason={cancellation.Reason}");

                        if (cancellation.Reason == CancellationReason.Error)
                        {
                            Console.WriteLine($"CANCELED: ErrorCode={cancellation.ErrorCode}");
                            Console.WriteLine($"CANCELED: ErrorDetails={[{cancellation.ErrorDetails}]}");
                            Console.WriteLine($"CANCELED: Did you update the subscription info?");
                        }
                    }
                }

                // This is to give some time for the speaker to finish playing back the audio
                Console.WriteLine("Press any key to exit...");
                Console.ReadKey();
            }
        }

        static void Main()
        {
            SynthesisToSpeakerAsync().Wait();
        }
    }
}

```

2. Busque la cadena `YourSubscriptionKey` y reemplácela por su clave de suscripción del servicio de voz.
3. Busque la cadena `YourServiceRegion` y reemplácela por la [región](#) asociada a su suscripción. Por ejemplo, si usa la suscripción de evaluación gratuita, la región es `westus`.
4. En la barra de menús, elija **Archivo > Guardar todo**.

# Compilación y ejecución de la aplicación

1. En la barra de menús, elija **Compilar > Compilar solución** para compilar la aplicación. El código se debería compilar sin errores ahora.
2. Elija **Depurar > Iniciar depuración** (o seleccione **F5**) para iniciar la aplicación **helloworld**.
3. Escriba una oración o frase en inglés. La aplicación transmite el texto al servicio de voz, que envía la voz sintetizada a la aplicación para que se reproduzca en el altavoz.

```
Type some text that you want to speak...
> hello
Speech synthesized to speaker for text [hello]
Press any key to exit...
```

## Pasos siguientes

[Exploración de ejemplos de C# en GitHub](#)

## Otras referencias

- [Creación de una voz personalizada](#)
- [Grabación de ejemplos de voz personalizada](#)

En este inicio rápido, usará el [SDK de Voz](#) para convertir texto en voz sintetizada. Una vez que se cumplen los requisitos previos, para la representación de voz sintetizada en los altavoces predeterminados solo son necesarios cuatro pasos:

- Cree un objeto `SpeechConfig` a partir de la clave y la región de suscripción.
- Cree un objeto `SpeechSynthesizer` con el objeto `SpeechConfig` anterior.
- Use el objeto `SpeechSynthesizer` para pronunciar el texto.
- Compruebe el objeto `SpeechSynthesisResult` devuelto en caso de errores.

Si prefiere ponerse a trabajar de inmediato, vea o descargue todos los [ejemplos para C++ del SDK de Voz](#) en GitHub. Si no, vamos a comenzar.

### Selección del entorno de destino

- [Linux](#)
- [Windows](#)

## Requisitos previos

Antes de comenzar, compruebe lo siguiente:

- [Ha creado un recurso de Voz de Azure](#)
- [Ha configurado el entorno de desarrollo](#)
- [Ha creado un proyecto de ejemplo vacío](#)

## Incorporación de código de ejemplo

1. Cree un archivo de código fuente C++ denominado `helloworld.cpp` y pegue el siguiente código en él.

```

#include <iostream> // cin, cout
#include <speechapi_cxx.h>

using namespace std;
using namespace Microsoft::CognitiveServices::Speech;

void synthesizeSpeech()
{
    // Creates an instance of a speech config with specified subscription key and service region.
    // Replace with your own subscription key and service region (e.g., "westus").
    auto config = SpeechConfig::FromSubscription("YourSubscriptionKey", "YourServiceRegion");

    // Creates a speech synthesizer using the default speaker as audio output. The default spoken
language is "en-us".
    auto synthesizer = SpeechSynthesizer::FromConfig(config);

    // Receive a text from console input and synthesize it to speaker.
    cout << "Type some text that you want to speak..." << std::endl;
    cout << "> ";
    std::string text;
    getline(cin, text);

    auto result = synthesizer->SpeakTextAsync(text).get();

    // Checks result.
    if (result->Reason == ResultReason::SynthesizingAudioCompleted)
    {
        cout << "Speech synthesized to speaker for text [" << text << "]" << std::endl;
    }
    else if (result->Reason == ResultReason::Canceled)
    {
        auto cancellation = SpeechSynthesisCancellationDetails::FromResult(result);
        cout << "CANCELED: Reason=" << (int)cancellation->Reason << std::endl;

        if (cancellation->Reason == CancellationReason::Error)
        {
            cout << "CANCELED: ErrorCode=" << (int)cancellation->ErrorCode << std::endl;
            cout << "CANCELED: ErrorDetails=[" << cancellation->ErrorDetails << "]" << std::endl;
            cout << "CANCELED: Did you update the subscription info?" << std::endl;
        }
    }

    // This is to give some time for the speaker to finish playing back the audio
    cout << "Press enter to exit..." << std::endl;
    cin.get();
}

int main(int argc, char **argv) {
    setlocale(LC_ALL, "");
    synthesizeSpeech();
    return 0;
}

```

2. En este nuevo archivo, reemplace la cadena `YourSubscriptionKey` por su clave de suscripción del servicio Voz.
3. Reemplace la cadena `YourServiceRegion` por la [región](#) asociada a sus suscripción (por ejemplo, `westus` para la suscripción de evaluación gratuita).

## Compilación de la aplicación

## NOTE

Asegúrese de introducir los siguientes comandos como una *única línea de comandos*. La forma más fácil de hacerlo es copiar el comando usando el botón **Copiar** junto a cada comando, y luego pegarlo en el símbolo del shell.

- En un sistema **x64** (64 bits), ejecute el siguiente comando para crear la aplicación.

```
g++ helloworld.cpp -o helloworld -I "$SPEECHSDK_ROOT/include/cxx_api" -I  
"$SPEECHSDK_ROOT/include/c_api" --std=c++14 -lpthread -lMicrosoft.CognitiveServices.Speech.core -L  
"$SPEECHSDK_ROOT/lib/x64" -l:libasound.so.2
```

- En un sistema **x86** (32 bits), ejecute el siguiente comando para crear la aplicación.

```
g++ helloworld.cpp -o helloworld -I "$SPEECHSDK_ROOT/include/cxx_api" -I  
"$SPEECHSDK_ROOT/include/c_api" --std=c++14 -lpthread -lMicrosoft.CognitiveServices.Speech.core -L  
"$SPEECHSDK_ROOT/lib/x86" -l:libasound.so.2
```

- En un sistema **ARM64** (64 bits), ejecute el siguiente comando para crear la aplicación.

```
g++ helloworld.cpp -o helloworld -I "$SPEECHSDK_ROOT/include/cxx_api" -I  
"$SPEECHSDK_ROOT/include/c_api" --std=c++14 -lpthread -lMicrosoft.CognitiveServices.Speech.core -L  
"$SPEECHSDK_ROOT/lib/arm64" -l:libasound.so.2
```

## Ejecución de la aplicación

1. Configuración de la ruta de acceso de la biblioteca del cargador para que apunte a la biblioteca del SDK de Voz.

- En un sistema **x64** (64 bits), escriba el siguiente comando.

```
export LD_LIBRARY_PATH="$LD_LIBRARY_PATH:$SPEECHSDK_ROOT/lib/x64"
```

- En un sistema **x86** (32 bits), escriba el siguiente comando.

```
export LD_LIBRARY_PATH="$LD_LIBRARY_PATH:$SPEECHSDK_ROOT/lib/x86"
```

- En un sistema **ARM64** (64 bits), escriba el siguiente comando.

```
export LD_LIBRARY_PATH="$LD_LIBRARY_PATH:$SPEECHSDK_ROOT/lib/arm64"
```

2. Ejecute la aplicación.

```
./helloworld
```

3. En la ventana de consola aparece un aviso que le pide que escriba texto. Escriba algunas palabras o una frase. El texto que escriba se transmite al servicio de voz y se sintetiza en una voz que se reproduce en el altavoz.

```
Type some text that you want to speak...
> hello
Speech synthesized to speaker for text [hello]
Press enter to exit...
```

## Pasos siguientes

[Exploración de ejemplos de C++ en GitHub](#)

## Otras referencias

- [Creación de una voz personalizada](#)
- [Grabación de ejemplos de voz personalizada](#)

En este inicio rápido, usará el [SDK de Voz](#) para convertir texto en voz sintetizada. Una vez que se cumplen los requisitos previos, para la representación de voz sintetizada en los altavoces predeterminados solo son necesarios cuatro pasos:

- Cree un objeto `SpeechConfig` a partir de la clave y la región de suscripción.
- Cree un objeto `SpeechSynthesizer` con el objeto `SpeechConfig` anterior.
- Use el objeto `SpeechSynthesizer` para pronunciar el texto.
- Compruebe el objeto `SpeechSynthesisResult` devuelto en caso de errores.

Si prefiere ponerse a trabajar de inmediato, vea o descargue todos los [ejemplos para Java del SDK de Voz](#) en GitHub. Si no, vamos a comenzar.

### Selección del entorno de destino

- [Java Runtime](#)
- [Android](#)

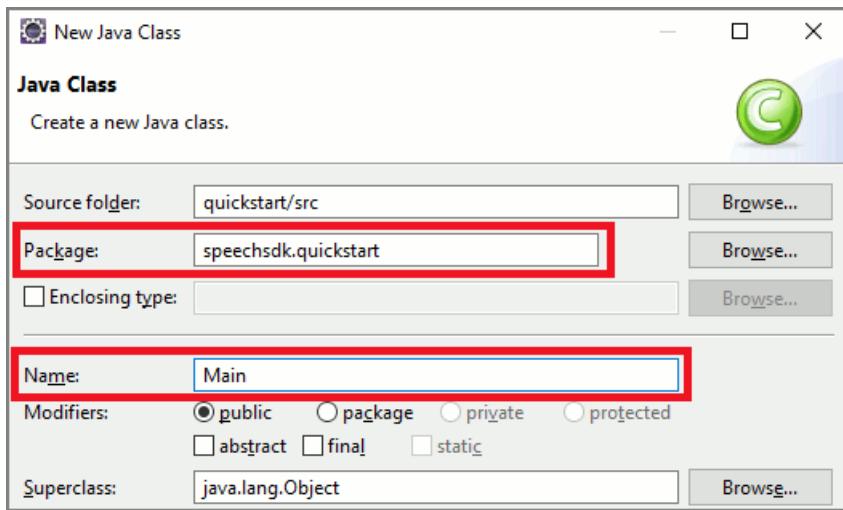
## Requisitos previos

Antes de comenzar, compruebe lo siguiente:

- [Ha creado un recurso de Voz de Azure](#)
- [Ha configurado el entorno de desarrollo](#)
- [Ha creado un proyecto de ejemplo vacío](#)

## Incorporación de código de ejemplo

1. Para agregar una nueva clase vacía al proyecto de Java, seleccione **File (Archivo) > New (Nuevo) > Class (Clase)**.
2. En la ventana **New Java Class** (Nueva clase de Java) escriba **speechsdk.quickstart** en el campo **Package** (Paquete) y **Main** en el campo **Name** (Nombre).



3. Reemplace el código en `Main.java` con el siguiente fragmento de código:

```
package speechsdk.quickstart;

import java.util.Scanner;
import java.util.concurrent.Future;
import com.microsoft.cognitiveservices.speech.*;

/**
 * Quickstart: synthesize speech using the Speech SDK for Java.
 */
public class Main {

    /**
     * @param args Arguments are ignored in this sample.
     */
    public static void main(String[] args) {
        try {
            // Replace below with your own subscription key
            String speechSubscriptionKey = "YourSubscriptionKey";
            // Replace below with your own service region (e.g., "westus").
            String serviceRegion = "YourServiceRegion";

            int exitCode = 1;
            SpeechConfig config = SpeechConfig.fromSubscription(speechSubscriptionKey,
serviceRegion);
            assert(config != null);

            SpeechSynthesizer synth = new SpeechSynthesizer(config);
            assert(synth != null);

            System.out.println("Type some text that you want to speak...");
            System.out.print("> ");
            String text = new Scanner(System.in).nextLine();

            Future<SpeechSynthesisResult> task = synth.SpeakTextAsync(text);
            assert(task != null);

            SpeechSynthesisResult result = task.get();
            assert(result != null);

            if (result.getReason() == ResultReason.SynthesizingAudioCompleted) {
                System.out.println("Speech synthesized to speaker for text [" + text + "]");
                exitCode = 0;
            }
            else if (result.getReason() == ResultReason.Canceled) {
                SpeechSynthesisCancellationDetails cancellation =
SpeechSynthesisCancellationDetails.fromResult(result);
                System.out.println("CANCELED: Reason=" + cancellation.getReason());
            }
        }
    }
}
```

```
        if (cancellation.getReason() == CancellationReason.Error) {
            System.out.println("CANCELED: ErrorCode=" + cancellation.getErrorCode());
            System.out.println("CANCELED: ErrorDetails=" + cancellation.getErrorDetails());
            System.out.println("CANCELED: Did you update the subscription info?");
        }
    }

    result.close();
    synth.close();

    System.exit(exitCode);
} catch (Exception ex) {
    System.out.println("Unexpected exception: " + ex.getMessage());

    assert(false);
    System.exit(1);
}
}
```

4. Reemplace la cadena `YourSubscriptionKey` por la clave de suscripción.
  5. Reemplace la cadena `YourServiceRegion` por la [región](#) asociada con la para la suscripción de evaluación gratuita).
  6. Guarde los cambios en el proyecto.

# Compilación y ejecución de la aplicación

Presione F11, o seleccione **Run (Ejecutar) > Debug (Depurar)**. Escriba un texto cuando se le solicite y oirá el audio sintetizado, que se reproduce en el altavoz predeterminado.

## Pasos siguientes

Exploración de ejemplos de Java en GitHub

## Otras referencias

- Creación de una voz personalizada
  - Grabación de ejemplos de voz personalizada

En este inicio rápido, usará el [SDK de Voz](#) para convertir texto en voz sintetizada. Una vez que se cumplen los requisitos previos, para la representación de voz sintetizada en los altavoces predeterminados solo son necesarios cuatro pasos:

- Cree un objeto `SpeechConfig` a partir de la clave y la región de suscripción.
  - Cree un objeto `SpeechSynthesizer` con el objeto `SpeechConfig` anterior.
  - Use el objeto `SpeechSynthesizer` para pronunciar el texto.
  - Compruebe el objeto `SpeechSynthesisResult` devuelto en caso de errores.

Si prefiere ponerse a trabajar de inmediato, vea o descargue todos los [ejemplos para Python del SDK de Voz](#) en GitHub. Si no, vamos a comenzar.

## Prerequisites

Antes de comenzar, compruebe lo siguiente:

- Ha creado un recurso de Voz de Azure
  - Ha configurado el entorno de desarrollo

- Ha creado un proyecto de ejemplo vacío

## Soporte técnico y actualizaciones

Las actualizaciones del paquete de Python del SDK de Voz se distribuirán mediante PyPI y se anunciarán en la página [Notas de la versión](#). Si hay disponible una nueva versión, puede actualizarse a ella con el comando `pip install --upgrade azure-cognitiveservices-speech`. Para comprobar qué versión está instalada actualmente, inspeccione la variable `azure.cognitiveservices.speech._version_`.

Si tiene un problema o falta una característica, consulte las [opciones de ayuda y soporte técnico](#).

## Creación de una aplicación de Python mediante el SDK de Voz

### Ejecución del ejemplo

Puede copiar el [código de ejemplo](#) de este inicio rápido en un archivo de código fuente `quickstart.py` y ejecutarlo en el IDE o en la consola:

```
python quickstart.py
```

También, puede descargar este tutorial de inicio rápido como un cuaderno de [Jupyter](#) del [repositorio de ejemplos del SDK de Voz](#) y ejecutarlo como un cuaderno.

### Código de ejemplo

```
import azure.cognitiveservices.speech as speechsdk

# Creates an instance of a speech config with specified subscription key and service region.
# Replace with your own subscription key and service region (e.g., "westus").
speech_key, service_region = "YourSubscriptionKey", "YourServiceRegion"
speech_config = speechsdk.SpeechConfig(subscription=speech_key, region=service_region)

# Creates a speech synthesizer using the default speaker as audio output.
speech_synthesizer = speechsdk.SpeechSynthesizer(speech_config=speech_config)

# Receives a text from console input.
print("Type some text that you want to speak...")
text = input()

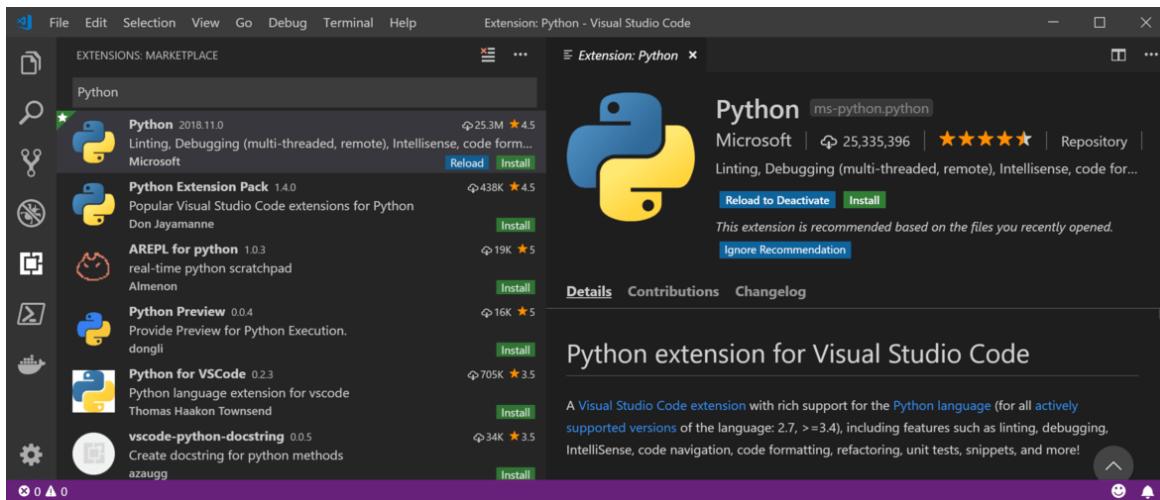
# Synthesizes the received text to speech.
# The synthesized speech is expected to be heard on the speaker with this line executed.
result = speech_synthesizer.speak_text_async(text).get()

# Checks result.
if result.reason == speechsdk.ResultReason.SynthesizingAudioCompleted:
    print("Speech synthesized to speaker for text [{}].format(text))")
elif result.reason == speechsdk.ResultReason.Canceled:
    cancellation_details = result.cancellation_details
    print("Speech synthesis canceled: {}".format(cancellation_details.reason))
    if cancellation_details.reason == speechsdk.CancellationReason.Error:
        if cancellation_details.error_details:
            print("Error details: {}".format(cancellation_details.error_details))
    print("Did you update the subscription info?")
```

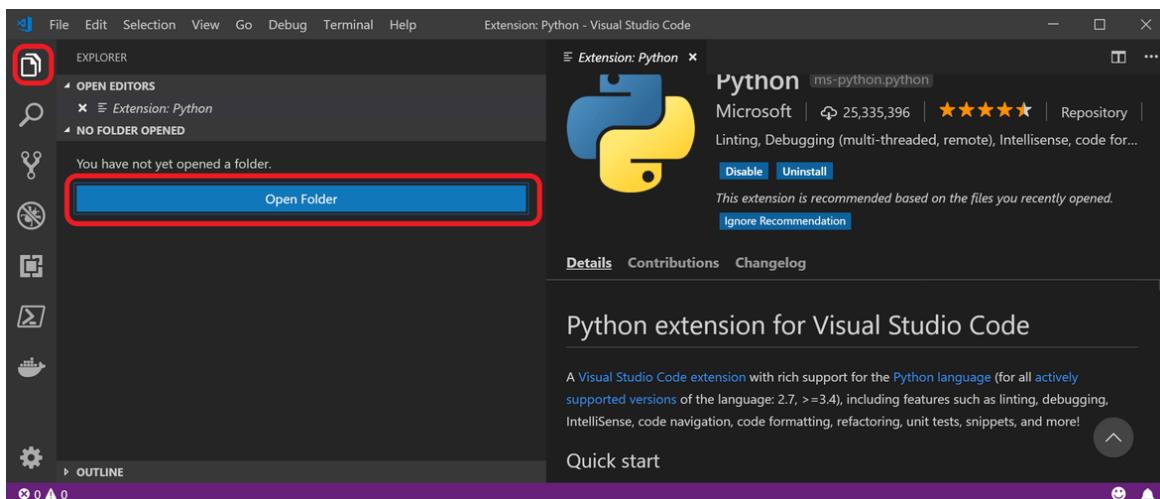
### Instalación y uso del SDK de Voz con Visual Studio Code

1. Descargue e instale una versión de 64 bits (3.5 o posterior) de [Python](#) en el equipo.
2. Descargue e instale [Visual Studio Code](#).
3. Abra Visual Studio Code e instale la extensión de Python. Seleccione **File > Preferences > Extensions**

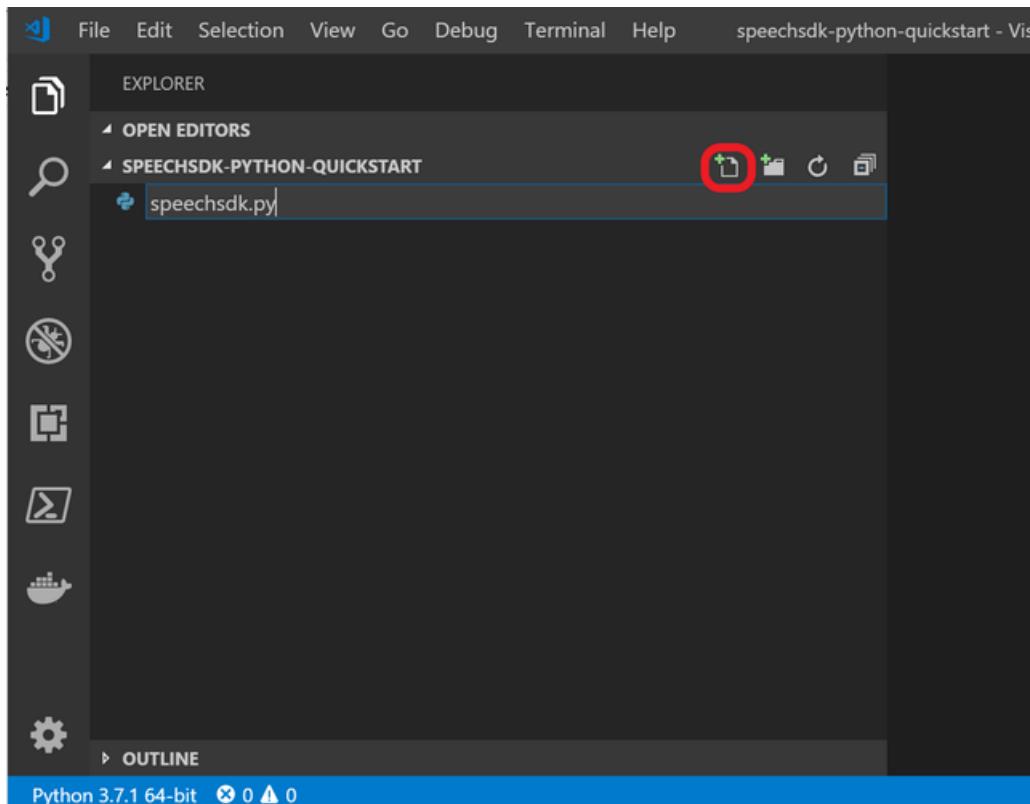
(Archivo > Preferencias > Extensiones) en el menú. Busque **Python**.



4. Cree una carpeta en la que almacenar el proyecto. Por ejemplo, puede usar para ello el Explorador de Windows.
5. En Visual Studio Code, seleccione el ícono de **File** (Archivo). A continuación, abra la carpeta que creó.



6. Cree un archivo de código fuente de Python `speechsdk.py` mediante la selección del ícono de nuevo archivo.



7. Copie, pegue y guarde el [código de Python](#) en el archivo recién creado.
8. Inserte la información de la suscripción del servicio Voz.
9. Si se selecciona, se muestra un intérprete de Python en el lado izquierdo de la barra de estado en la parte inferior de la ventana. En caso contrario, aparecerá una lista de los intérpretes de Python disponibles. Abra la paleta de comandos (Ctrl+Mayús+P) y escriba **Python: Select Interpreter** (Seleccionar intérprete). Elija un valor apropiado.
10. Puede instalar el paquete de Python del SDK de Voz desde dentro de Visual Studio Code. Hágalo si no está instalado aún para el intérprete de Python seleccionado. Para instalar el paquete del SDK de Voz, abra un terminal. Abra de nuevo la paleta de comandos (Ctrl+Mayús+P) y escriba **Terminal: Create New Integrated Terminal** (Crear terminal integrado). En el terminal que se abre, escriba el comando `python -m pip install azure-cognitiveservices-speech` o el que sea apropiado para su sistema.
11. Para ejecutar el código de ejemplo, haga clic con el botón derecho en algún lugar dentro del editor. Seleccione **Run Python File in Terminal** (Ejecutar archivo de Python en terminal). Escriba texto cuando se le solicite. El audio sintetizado se reproduce poco después.

```
File Edit Selection View Go Debug Terminal Help
quickstart.py - speechsdk-python-quickstart - Visual Studio Code
OPEN EDITORS
  × quickstart.py
SPEECHSDK-PYTHON-QUICKSTART
  quickstart.py
quickstart.py
6
7  # Creates an instance of a spe
8  # Replace with your own subscr
9  speech_key , service_region = "
10 speech_config = speechsdk.Spee
11
12  # Creates a speech synthesizer
13 speech_synthesizer = speechsdk
14
15  # Receives a text from console
16 print("Type some text that you
17 text = input()
18
19  # Synthesizes the received tex
20  # The synthesized speech is ex
21 result = speech_synthesizer.sp
22
23  # Checks result.
24 if result.reason == speechsdk.
25     print("Speech synthesized
26 elif result.reason == speechsd
27     cancellation_details = res
28     print("Speech synthesis ca
29     if cancellation_detail
30         if cancellation_detail
31             print("Error detail
32     print("Did you update the
33  # </code>
34
```

Context menu options include: Go to Definition, Peek Definition, Find All References, Peek References, Rename Symbol, Change All Occurrences, Format Document, Format Document With..., Source Action..., Cut, Copy, Paste, Add Word to Folder Dictionary, Add Word to User Dictionary, Ignore Word, Run Current Test File, Run Python File in Terminal (highlighted), Run Selection/Line in Python Terminal, Run Current File in Python Interactive Window, Sort Imports, Command Palette..., and JTF-8 CRLF Python.

Si tiene problemas para seguir estas instrucciones, consulte el [tutorial de Python para Visual Studio Code](#) con información más amplia.

## Pasos siguientes

[Exploración de los ejemplos de Python en GitHub](#)

## Consulte también

- [Creación de una voz personalizada](#)
- [Grabación de ejemplos de voz personalizada](#)

Vea o descargue todos los [ejemplos del SDK de Voz](#) en GitHub.

## Compatibilidad con plataformas y lenguajes adicionales

Si ha hecho clic en esta pestaña, es probable que no vea un inicio rápido en su lenguaje de programación favorito. No se preocupe, tenemos materiales de inicio rápido y ejemplos de código adicionales disponibles en GitHub. Use la tabla para encontrar el ejemplo correcto para su lenguaje de programación y combinación de plataforma y sistema operativo.

IDIOMA	INICIOS RÁPIDOS ADICIONALES	EJEMPLOS DE CÓDIGO
C++		<a href="#">Inicios rápidos</a> , <a href="#">Ejemplos</a>
C#		<a href="#">.NET Framework</a> , <a href="#">.NET Core</a> , <a href="#">UWP</a> , <a href="#">Unity</a> , <a href="#">Xamarin</a>
Java		<a href="#">Android</a> , <a href="#">JRE</a>
JavaScript		<a href="#">Browser</a>
Node.js		<a href="#">Windows</a> , <a href="#">Linux</a> , <a href="#">macOS</a>
Objective-C	macOS, iOS	iOS, macOS

IDIOMA	INICIOS RÁPIDOS ADICIONALES	EJEMPLOS DE CÓDIGO
Python		<a href="#">Windows, Linux, macOS</a>
Swift	<a href="#">macOS, iOS</a>	<a href="#">iOS, macOS</a>

# Inicio rápido: síntesis asincrónica para audio de formato largo en Python (versión preliminar)

13/01/2020 • 11 minutes to read • [Edit Online](#)

En esta guía de inicio rápido, usará Long Audio API para convertir texto a voz de forma asincrónica y recuperar la salida de audio de un URI proporcionado por el servicio. Esta API REST es ideal para los proveedores de contenido que necesitan convertir archivos de texto de más de 10 000 caracteres o 50 párrafos en voz sintetizada. Para obtener más información, consulte [Long Audio API](#).

## NOTE

La síntesis asincrónica de audio de formato largo solo se puede usar con [voces neuronales personalizadas](#).

## Requisitos previos

Esta guía de inicio rápido requiere:

- Python 2.7.x o 3.x.
- [Visual Studio](#), [Visual Studio Code](#) o su editor favorito de código.
- Una suscripción a Azure y una clave de suscripción al servicio de Voz. [Cree una cuenta de Azure](#) y [cree un recurso de voz](#) para obtener la clave. Al crear el recurso de voz, asegúrese de que el plan de tarifa está establecido en **S0** y la ubicación está establecida en una [región compatible](#).

## Creación de un proyecto e importación de los módulos necesarios

Cree un nuevo proyecto de Python con su IDE o editor favorito. A continuación, copie este fragmento de código en un archivo llamado `voice_synthesis_client.py`.

```
import argparse
import json
import ntpath
import urllib3
import requests
import time
from json import dumps, loads, JSONEncoder, JSONDecoder
import pickle

urllib3.disable_warnings(urllib3.exceptions.InsecureRequestWarning)
```

## NOTE

Si no ha usado estos módulos deberá instalarlos antes de ejecutar el programa. Para instalar estos paquetes, ejecute:

```
pip install requests urllib3
```

Estos módulos se usan para analizar argumentos, construir la solicitud HTTP y llamar a la API REST de audios largos para convertir texto a voz.

## Obtención de una lista de voces compatibles

Este código obtiene una lista de las voces disponibles que puede usar para convertir texto a voz. Agregue este código a `voice_synthesis_client.py`:

```
parser = argparse.ArgumentParser(description='Cris client tool to submit voice synthesis requests.')
parser.add_argument('--voices', action="store_true", default=False, help='print voice list')
parser.add_argument('-key', action="store", dest="key", required=True, help='the cris subscription key, like ff1eb62d06d34767bda0207acb1da7d7 ')
parser.add_argument('-region', action="store", dest="region", required=True, help='the region information, could be centralindia, canadacentral or uksouth')
args = parser.parse_args()
baseAddress = 'https://%s.cris.ai/api/texttospeech/v3.0-beta1/' % args.region

def getVoices():
    response=requests.get(baseAddress+"voicesynthesis/voices", headers={"Ocp-Apim-Subscription-Key":args.key}, verify=False)
    voices = json.loads(response.text)
    return voices

if args.voices:
    voices = getVoices()
    print("There are %d voices available:" % len(voices))
    for voice in voices:
        print ("Name: %s, Description: %s, Id: %s, Locale: %s, Gender: %s, PublicVoice: %s, Created: %s" % (voice['name'], voice['description'], voice['id'], voice['locale'], voice['gender'], voice['isPublicVoice'], voice['created']))
```

## Prueba del código

Probemos lo que ha hecho hasta ahora. Ejecute este comando. Para ello, reemplace `<your_key>` por su clave de suscripción de Voz y `<region>` por la región donde se creó el recurso de Voz (por ejemplo: `eastus` o `westus`). Esta información está disponible en la pestaña de **información general** del recurso en [Azure Portal](#).

```
python voice_synthesis_client.py --voices -key <your_key> -region <Region>
```

Debe obtener un resultado con el siguiente aspecto:

```
There are xx voices available:

Name: Microsoft Server Speech Text to Speech Voice (en-US, xxx), Description: xxx , Id: xxx, Locale: en-US,
Gender: Male, PublicVoice: xxx, Created: 2019-07-22T09:38:14Z
Name: Microsoft Server Speech Text to Speech Voice (zh-CN, xxx), Description: xxx , Id: xxx, Locale: zh-CN,
Gender: Female, PublicVoice: xxx, Created: 2019-08-26T04:55:39Z
```

## Conversión de texto en voz

El siguiente paso es preparar un archivo de texto de entrada. Puede ser un texto sin formato o texto SSML, pero debe tener más de 10 000 caracteres o 50 párrafos. Para obtener una lista completa de los requisitos, consulte [Long Audio API](#).

Después de preparar el archivo de texto, el siguiente paso consiste en agregar código para la síntesis de voz al proyecto. Agregue este código a `voice_synthesis_client.py`:

### NOTE

De forma predeterminada, la salida de audio se establece en riff-16khz-16bit-mono-pcm. Para obtener más información sobre las salidas de audio compatibles, consulte [Long audio API](#).

```

parser.add_argument('--submit', action="store_true", default=False, help='submit a synthesis request')
parser.add_argument('--concatenateResult', action="store_true", default=False, help='If concatenate result in
a single wave file')
parser.add_argument('-file', action="store", dest="file", help='the input text script file path')
parser.add_argument('-voiceId', action="store", nargs='+', dest="voiceId", help='the id of the voice which
used to synthesis')
parser.add_argument('-locale', action="store", dest="locale", help='the locale information like zh-CN/en-US')
parser.add_argument('-format', action="store", dest="format", default='riff-16khz-16bit-mono-pcm', help='the
output audio format')

def submitSynthesis():
    modelList = args.voiceId
    data={'name': 'simple test', 'description': 'desc...', 'models': json.dumps(modelList), 'locale':
args.locale, 'outputformat': args.format}
    if args.concatenateResult:
        properties={'ConcatenateResult': 'true'}
        data['properties'] = json.dumps(properties)
    if args.file is not None:
        scriptfilename=ntpath.basename(args.file)
        files = {'script': (scriptfilename, open(args.file, 'rb'), 'text/plain')}
        response = requests.post(baseAddress+"voicesynthesis", data, headers={"Ocp-Apim-Subscription-
Key":args.key}, files=files, verify=False)
        if response.status_code == 202:
            location = response.headers['Operation-Location']
            id = location.split("/")[-1]
            print("Submit synthesis request successful")
            return id
        else:
            print("Submit synthesis request failed")
            print("response.status_code: %d" % response.status_code)
            print("response.text: %s" % response.text)
            return 0

def getSubmittedSynthesis(id):
    response=requests.get(baseAddress+"voicesynthesis/"+id, headers={"Ocp-Apim-Subscription-Key":args.key},
verify=False)
    synthesis = json.loads(response.text)
    return synthesis

if args.submit:
    id = submitSynthesis()
    if (id == 0):
        exit(1)

    while(1):
        print("\r\nChecking status")
        synthesis=getSubmittedSynthesis(id)
        if synthesis['status'] == "Succeeded":
            r = requests.get(synthesis['resultsUrl'])
            filename=id + ".zip"
            with open(filename, 'wb') as f:
                f.write(r.content)
            print("Succeeded... Result file downloaded : " + filename)
            break
        elif synthesis['status'] == "Failed":
            print("Failed...")
            break
        elif synthesis['status'] == "Running":
            print("Running...")
        elif synthesis['status'] == "NotStarted":
            print("NotStarted...")
        time.sleep(10)

```

## Prueba del código

Intentaremos realizar una solicitud para sintetizar el texto; usaremos el archivo de entrada como origen. Deberá

actualizar algunos detalles en la solicitud siguiente:

- Reemplace <your\_key> por la clave de suscripción del servicio Speech. Esta información está disponible en la pestaña de **información general** del recurso en [Azure Portal](#).
- Reemplace <region> por la región donde se creó el recurso de Voz (por ejemplo: `eastus` o `westus`). Esta información está disponible en la pestaña de **información general** del recurso en [Azure Portal](#).
- Reemplace <input> por la ruta de acceso al archivo de texto que quiere convertir de texto a voz.
- Reemplace <locale> por la configuración regional de salida deseada. Para obtener más información, consulte [Compatibilidad de idioma](#).
- Reemplace <voice\_guid> por la voz deseada para la salida de audio. Use una de las voces devueltas por el método de [obtención de una lista de voces compatibles](#) o use la lista de voces neuronales que se proporciona en [compatibilidad de idioma](#).

Convierta texto a voz con este comando:

```
python voice_synthesis_client.py --submit -key <your_key> -region <Region> -file <input> -locale <locale> -voiceId <voice_guid>
```

#### NOTE

"concatenateResult" es un parámetro opcional; si no se proporciona este parámetro, el resultado estará compuesto por varios archivos de onda, uno para cada línea.

Debe obtener un resultado con el siguiente aspecto:

```
Submit synthesis request successful

Checking status
NotStarted...

Checking status
Running...

Checking status
Running...

Checking status
Succeeded... Result file downloaded : xxxx.zip
```

El resultado proporcionado contiene el texto de entrada y los archivos de salida de audio generados por el servicio. Se descargan como un archivo ZIP.

## Eliminación de solicitudes anteriores

Hay un límite de 2000 solicitudes para cada suscripción. Por ello, habrá ocasiones en las que necesite quitar las solicitudes enviadas previamente para poder crear otras nuevas. Si no quita las solicitudes existentes, recibirá un error cuando supere las 2000.

Agregue este código a `voice_synthesis_client.py`:

```

parser.add_argument('--syntheses', action="store_true", default=False, help='print synthesis list')
parser.add_argument('--delete', action="store_true", default=False, help='delete a synthesis request')
parser.add_argument('-synthesisId', action="store", nargs='+', dest="synthesisId", help='the id of the voice
synthesis which need to be deleted')

def getSubmittedSyntheses():
    response=requests.get(baseAddress+"voicesynthesis", headers={"Ocp-Apim-Subscription-Key":args.key},
verify=False)
    syntheses = json.loads(response.text)
    return syntheses

def deleteSynthesis(ids):
    for id in ids:
        print("delete voice synthesis %s " % id)
        response = requests.delete(baseAddress+"voicesynthesis/"+id, headers={"Ocp-Apim-Subscription-
Key":args.key}, verify=False)
        if (response.status_code == 204):
            print("delete successful")
        else:
            print("delete failed, response.status_code: %d, response.text: %s " % (response.status_code,
response.text))

if args.syntheses:
    synthese = getSubmittedSyntheses()
    print("There are %d synthesis requests submitted:" % len(synthese))
    for synthesis in synthese:
        print ("ID : %s , Name : %s, Status : %s " % (synthesis['id'], synthesis['name'],
synthesis['status']))

if args.delete:
    deleteSynthesis(args.synthesisId)

```

## Prueba del código

Ejecute este comando. Para ello, reemplace <your\_key> por su clave de suscripción de Voz y <region> por la región donde se creó el recurso de Voz (por ejemplo: `eastus` o `westus`). Esta información está disponible en la pestaña de **información general** del recurso en [Azure Portal](#).

```
python voice_synthesis_client.py - syntheses -key <your_key> -region <Region>
```

Esto devolverá una lista de síntesis solicitadas. Debe obtener un resultado con el siguiente aspecto:

```

There are <number> synthesis requests submitted:
ID : xxx , Name : xxx, Status : Succeeded
ID : xxx , Name : xxx, Status : Running
ID : xxx , Name : xxx : Succeeded

```

Ahora usaremos algunos de estos valores para quitar o eliminar solicitudes enviadas previamente. Ejecute este comando. Para ello, reemplace <your\_key> por su clave de suscripción de Voz y <region> por la región donde se creó el recurso de Voz (por ejemplo: `eastus` o `westus`). Esta información está disponible en la pestaña de **información general** del recurso en [Azure Portal](#). El <synthesis\_id> debe ser uno de los valores devueltos en la solicitud anterior.

### NOTE

Las solicitudes con el estado "En ejecución" o "En espera" no se pueden quitar ni eliminar.

```
python voice_synthesis_client.py - delete -key <your_key> -region <Region> -synthesisId <synthesis_id>
```

Debe obtener un resultado con el siguiente aspecto:

```
delete voice synthesis xxx
delete successful
```

## Obtención del cliente completo

El `voice_synthesis_client.py` completo está disponible para su descarga en [GitHub](#).

## Pasos siguientes

[Más información sobre Long Audio API](#)

# Lenguaje de marcado de síntesis de voz (SSML)

13/01/2020 • 31 minutes to read • [Edit Online](#)

El lenguaje de marcado de síntesis de voz (SSML) es un lenguaje de marcado basado en XML que permite a los desarrolladores especificar cómo se convierte el texto de entrada en una voz sintetizada mediante el servicio de texto a voz. En comparación con el texto sin formato, SSML permite a los desarrolladores ajustar el tono, la pronunciación, la velocidad del habla, el volumen y muchas cosas más en la salida de texto a voz. La puntuación normal, como hacer una pausa después de un punto o usar la entonación correcta cuando una oración termina con un signo de interrogación, se administra automáticamente.

La implementación del servicios de voz de SSML se basa en la [versión 1.0 del lenguaje de marcado de síntesis de voz](#) del World Wide Web Consortium.

## IMPORTANT

Los caracteres en chino, japonés y coreano se cuentan como dos caracteres para la facturación. Para obtener más información, consulte el apartado [Precios](#).

## Voces neuronales, estándares y personalizadas

Puede elegir entre voces estándar y neuronales, o puede crear su propia voz personalizada única para su producto o marca. Tiene más de 75 voces estándar disponibles en más de 45 idiomas y configuraciones regionales y 5 voces neuronales que están disponibles en 4 idiomas y configuraciones regionales. Para obtener una lista completa de los idiomas compatibles, las configuraciones regionales y las voces (neuronales y estándar), consulte [compatibilidad con idiomas](#).

Para más información sobre las voces estándar, neuronales y personalizadas, consulte la [introducción del texto a voz](#).

## Caracteres especiales

Al usar SSML para convertir el texto en voz sintetizada, tenga en cuenta que, al igual que con XML, los caracteres especiales, como comillas, apóstrofos y corchetes, deben ser de escape. Para más información, consulte [Lenguaje de marcado extensible \(XML\) 1.0: Apéndice D](#).

## Elementos SSML admitidos

Cada documento SSML se crea con los elementos SSML (o etiquetas). Estos elementos se utilizan para ajustar el tono, la prosodia, el volumen y mucho más. En las siguientes secciones se detallan cómo se utiliza cada elemento y cuándo es necesario u opcional.

## IMPORTANT

No se olvide de utilizar comillas dobles alrededor de los valores de atributo. Las normas para un XML válido y bien formado requieren que los valores de los atributos estén rodeados de comillas dobles. Por ejemplo, `<prosody volume="90">` es un elemento válido y bien formado, pero `<prosody volume=90>` no. Es posible que SSML no reconozca valores de atributos que no estén entre comillas.

## Creación de un documento SSML

`speak` es el elemento raíz y se **necesita** para todos los documentos SSML. El elemento `speak` contiene información importante, como la versión, el idioma y la definición del vocabulario de marcado.

## Sintaxis

```
<speak version="1.0" xmlns="https://www.w3.org/2001/10/synthesis" xml:lang="string"></speak>
```

## Atributos

ATRIBUTO	DESCRIPCIÓN	OBLIGATORIO U OPCIONAL
version	Indica la versión de la especificación SSML utilizada para interpretar el marcado del documento. La versión actual es 1.0.	Obligatorio
xml:lang	Especifica el idioma del documento raíz. El valor puede contener un código de idioma en minúsculas y de dos letras (por ejemplo, <b>es</b> ) o el código de idioma y el país o región en mayúscula (por ejemplo, <b>es-US</b> ).	Obligatorio
xmlns	Especifica el URI del documento que define el vocabulario de marcado (los tipos de elementos y nombres de atributos) del documento SSML. El identificador URI actual es <a href="https://www.w3.org/2001/10/synthesis">https://www.w3.org/2001/10/synthesis</a> .	Obligatorio

## Elección de una voz para la conversión de texto a voz

El elemento `voice` es obligatorio. Sirve para especificar la voz que se usa en la conversión de texto a voz.

## Sintaxis

```
<voice name="string">
    This text will get converted into synthesized speech.
</voice>
```

## Atributos

ATRIBUTO	DESCRIPCIÓN	OBLIGATORIO U OPCIONAL
Nombre	Identifica la voz que se usa para la salida de texto a voz. Para ver una lista completa de voces compatibles, consulte <a href="#">Compatibilidad con idiomas</a> .	Obligatorio

## Ejemplo

### NOTE

En este ejemplo se usa la voz `en-US-Jessa24kRUS`. Para ver una lista completa de voces compatibles, consulte [Compatibilidad con idiomas](#).

```
<speak version="1.0" xmlns="https://www.w3.org/2001/10/synthesis" xml:lang="en-US">
    <voice name="en-US-Jessa24kRUS">
        This is the text that is spoken.
    </voice>
</speak>
```

## Uso de varias voces

Dentro del elemento `speak`, puede especificar varias voces para la salida de texto a voz. Estas voces pueden estar en diferentes idiomas. Para cada voz, el texto se debe encapsular en un elemento `voice`.

### Atributos

ATRIBUTO	DESCRIPCIÓN	OBLIGATORIO U OPCIONAL
Nombre	Identifica la voz que se usa para la salida de texto a voz. Para ver una lista completa de voces compatibles, consulte <a href="#">Compatibilidad con idiomas</a> .	Obligatorio

### Ejemplo

```
<speak version="1.0" xmlns="https://www.w3.org/2001/10/synthesis" xml:lang="en-US">
    <voice name="en-US-Jessa24kRUS">
        Good morning!
    </voice>
    <voice name="en-US-Guy24kRUS">
        Good morning to you too Jessa!
    </voice>
</speak>
```

## Ajuste de los estilos de habla

### IMPORTANT

Esta característica solo funciona con las voces neuronales.

De forma predeterminada, el servicio de texto a voz sintetiza el texto mediante un estilo de habla neutro tanto para voces estándar como neuronales. Con las voces neuronales, puede ajustar el estilo de habla para expresar alegría, empatía o sentimiento con el elemento `<mstts:express-as>`. Se trata de un elemento opcional único para el servicio de voz.

Actualmente, los ajustes de estilo de habla son compatibles con estas voces neuronales:

- `en-US-JessaNeutral`
- `zh-CN-XiaoxiaoNeutral`

Los cambios se aplican en el nivel de la oración y el estilo varía según la voz. Si no se admite un estilo, el servicio devolverá la voz con el estilo de habla neutro predeterminado.

### Sintaxis

```
<mstts:express-as type="string"></mstts:express-as>
```

## Atributos

ATRIBUTO	DESCRIPCIÓN	OBLIGATORIO U OPCIONAL
Tipo	Especifica el estilo de habla. Actualmente, los estilos de habla son específicos de la voz.	Se necesita si se ajusta el estilo de habla para una voz neuronal. Si se usa <code>mstts:express-as</code> , se debe proporcionar el tipo. Si se proporciona un valor no válido, se omitirá este elemento.

Utilice esta tabla para determinar qué estilos de habla son compatibles para cada voz neuronal.

VOZ	TIPO	DESCRIPCIÓN
en-US-JessaNeural	tipo = <code>cheerful</code>	Expresa una emoción que es positiva y feliz.
	tipo = <code>empathy</code>	Expresa un sentimiento de cuidado y comprensión.
	tipo = <code>chat</code>	Harlar en un tono informal y relajado
zh-CN-XiaoxiaoNeural	tipo = <code>newscast</code>	Expresa un tono formal, similar a las retransmisiones de noticias.
	tipo = <code>sentiment</code>	Transmite un mensaje conmovedor o una historia.

## Ejemplo

Este fragmento de SSML ilustra cómo se utiliza el elemento `<mstts:express-as>` para cambiar el estilo de habla a `cheerful`.

```
<speak version="1.0" xmlns="https://www.w3.org/2001/10/synthesis"
      xmlns:mstts="https://www.w3.org/2001/mstts" xml:lang="en-US">
  <voice name="en-US-JessaNeural">
    <mstts:express-as type="cheerful">
      That'd be just amazing!
    </mstts:express-as>
  </voice>
</speak>
```

## Adición o supresión de una pausa

Utilice el elemento `break` para insertar las pausas entre palabras, o para evitar pausas agregadas automáticamente por el servicio de texto a voz.

### NOTE

Utilice este elemento para invalidar el comportamiento predeterminado de texto a voz de una palabra o frase si el habla sintetizada de esa palabra o frase no suena natural. Establezca `strength` en `none` para evitar una pausa prosódica, que inserta automáticamente el servicio de texto a voz.

## Sintaxis

```
<break strength="string" />
<break time="string" />
```

## Atributos

ATRIBUTO	DESCRIPCIÓN	OBLIGATORIO U OPCIONAL
intensidad	Especifica la duración relativa de una pausa mediante uno de los valores siguientes: <ul style="list-style-type: none"> <li>• None</li> <li>• x-weak</li> <li>• weak</li> <li>• medium (default)</li> <li>• strong</li> <li>• x-strong</li> </ul>	Opcional
time	Especifica la duración absoluta de una pausa en segundos o milisegundos. Ejemplos de los valores válidos son 2 s y 500	Opcional

INTENSIDAD	DESCRIPCIÓN
Ninguno, o si no se ha proporcionado ningún valor	0 ms
x-weak	250 ms
weak	500 ms
medio	750 ms
strong	1000 ms
x-strong	1250 ms

## Ejemplo

```
<speak version="1.0" xmlns="https://www.w3.org/2001/10/synthesis" xml:lang="en-US">
    <voice name="en-US-Jessa24kRUS">
        Welcome to Microsoft Cognitive Services <break time="100ms" /> Text-to-Speech API.
    </voice>
</speak>
```

## Especificación de párrafos y oraciones

Los elementos `p` y `s` se utilizan para indicar párrafos y oraciones, respectivamente. En ausencia de estos elementos, el servicio de texto a voz determina automáticamente la estructura del documento SSML.

El elemento `p` puede contener texto y los siguientes elementos: `audio`, `break`, `phoneme`, `prosody`, `say-as`, `sub`, `mstts:express-as` y `s`.

El elemento `s` puede contener texto y los siguientes elementos: `audio`, `break`, `phoneme`, `prosody`, `say-as`, `mstts:express-as` y `sub`.

## Sintaxis

```
<p></p>
<s></s>
```

## Ejemplo

```
<speak version="1.0" xmlns="https://www.w3.org/2001/10/synthesis" xml:lang="en-US">
  <voice name="en-US-Jessa24kRUS">
    <p>
      <s>Introducing the sentence element.</s>
      <s>Used to mark individual sentences.</s>
    </p>
    <p>
      Another simple paragraph.
      Sentence structure in this paragraph is not explicitly marked.
    </p>
  </voice>
</speak>
```

## Uso de fonemas para mejorar la pronunciación

El elemento `ph` se utiliza para la pronunciación fonética en los documentos SSML. El elemento `ph` solo puede contener texto, no otros elementos. Proporcione siempre una voz natural como reserva.

Los alfabetos fonéticos se componen de segmentos acústicos, que se componen de letras, números o caracteres, a veces en combinación. Cada segmento acústico describe un sonido de voz único. Esto contrasta con el alfabeto latino, donde cualquier letra puede representar múltiples sonidos hablados. Tenga en cuenta las pronunciasiones diferentes de la letra "c" en las palabras "casa" y "cese" o, en inglés, las distintas pronunciasiones de la combinación de letras "th" en las palabras "thing" y "those".

## Sintaxis

```
<phoneme alphabet="string" ph="string"></phoneme>
```

## Atributos

ATRIBUTO	DESCRIPCIÓN	OBLIGATORIO U OPCIONAL
alfabeto	<p>Especifica el alfabeto fonético que se utilizará al sintetizar la pronunciación de la cadena en el atributo <code>ph</code>. La cadena que especifica el alfabeto debe especificarse en minúsculas. Estos son los posibles alfabetos que puede especificar.</p> <ul style="list-style-type: none"><li>ipa: alfabeto fonético internacional</li><li>sapi: conjunto de segmentos acústicos de Speech API</li><li>ups: conjunto de segmentos acústicos universal</li></ul> <p>El alfabeto solo se aplica al fonema del elemento. Para más información, consulte <a href="#">Phonetic Alphabet Reference</a> (Referencia del alfabeto fonético).</p>	Opcional

ATRIBUTO	DESCRIPCIÓN	OBLIGATORIO U OPCIONAL
ph	Una cadena que contiene los segmentos acústicos que especifican la pronunciación de la palabra en el elemento <code>phoneme</code> . Si la cadena especificada contiene segmentos acústicos no reconocidos, el servicio de texto a voz rechaza todo el documento SSML y no produce ninguna de las salidas de voz especificadas en el documento.	Obligatorio si usa fonemas.

## Ejemplos

```
<speak version="1.0" xmlns="https://www.w3.org/2001/10/synthesis" xml:lang="en-US">
    <voice name="en-US-Jessa24kRUS">
        <s>His name is Mike <phoneme alphabet="ups" ph="JH AU"> Zhou </phoneme></s>
    </voice>
</speak>
```

```
<speak version="1.0" xmlns="https://www.w3.org/2001/10/synthesis" xml:lang="en-US">
    <voice name="en-US-Jessa24kRUS">
        <phoneme alphabet="ipa" ph="təmei̥ɾou̥"> tomato </phoneme>
    </voice>
</speak>
```

## Ajuste de la prosodia

El elemento `prosody` se utiliza para especificar los cambios en el tono, la curva melódica, el rango, la velocidad, la duración y el volumen de la salida de texto a voz. El elemento `prosody` puede contener texto y los siguientes elementos: `audio`, `break`, `p`, `phoneme`, `prosody`, `say-as`, `sub` y `s`.

Dado que los valores de los atributos prosódicos pueden variar en un amplio rango, el reconocedor de voz interpreta los valores asignados como una sugerencia de cuáles deben ser los valores prosódicos reales de la voz seleccionada. El servicio de texto a voz limita o sustituye valores que no son compatibles. Ejemplos de valores no compatibles son un tono de 1 MHz o un volumen de 120.

### Sintaxis

```
<prosody pitch="value" contour="value" range="value" rate="value" duration="value" volume="value">
</prosody>
```

## Atributos

ATRIBUTO	DESCRIPCIÓN	OBLIGATORIO U OPCIONAL
----------	-------------	------------------------

ATRIBUTO	DESCRIPCIÓN	OBLIGATORIO U OPCIONAL
pitch	<p>Indica el tono de la línea de referencia del texto. Puede expresar el tono como:</p> <ul style="list-style-type: none"> <li>• Un valor absoluto, expresado como un número seguido de "Hz" (Hercios). Por ejemplo, 600 Hz.</li> <li>• Un valor relativo, expresado como un número precedido por "+" o "-" y seguido por "Hz" o "st", que especifica una cantidad para cambiar el tono. Por ejemplo: +80 Hz o -2st. La "st" indica que la unidad de cambio es un semitono, que es la mitad de un tono (medio paso) en la escala diatónica estándar.</li> <li>• Un valor constante: <ul style="list-style-type: none"> <li>◦ x-low</li> <li>◦ bajo</li> <li>◦ medio</li> <li>◦ alta</li> <li>◦ x-high</li> <li>◦ default</li> </ul> </li> </ul>	Opcional
contour	<p>La curva melódica no es compatible con las voces neuronales. La curva melódica representa los cambios en el tono del contenido de la voz como una matriz de objetivos en posiciones de tiempo específicas en la salida de voz. Cada destino se define por conjuntos de pares de parámetros. Por ejemplo:</p> <pre>&lt;prosody contour="(0%,+20Hz) (10%, -2st) (40%,+10Hz)"&gt;</pre> <p>El primer valor de cada conjunto de parámetros especifica la ubicación del cambio de tono como porcentaje de la duración del texto. El segundo valor especifica la cantidad para subir o bajar el tono, mediante un valor relativo o un valor de enumeración para el tono (consulte <code>pitch</code>).</p>	Opcional
range	<p>Un valor que representa el rango del tono para el texto. Puede expresar <code>range</code> mediante los mismos valores absolutos, valores relativos o valores de enumeración usados para describir <code>pitch</code>.</p>	Opcional

ATRIBUTO	DESCRIPCIÓN	OBLIGATORIO U OPCIONAL
rate	<p>Indica la velocidad de habla del texto. Puede expresar <code>rate</code> como:</p> <ul style="list-style-type: none"> <li>• Un valor relativo, expresado como un número que actúa como multiplicador del valor predeterminado. Por ejemplo, un valor de <code>1</code> no da como resultado ningún cambio en la velocidad. Un valor de <code>.5</code> da como resultado la mitad de la velocidad. Un valor de <code>3</code> da como resultado el triple de la velocidad.</li> <li>• Un valor constante: <ul style="list-style-type: none"> <li>◦ <code>x-slow</code></li> <li>◦ <code>lento</code></li> <li>◦ <code>medio</code></li> <li>◦ <code>fast</code></li> <li>◦ <code>x-fast</code></li> <li>◦ <code>default</code></li> </ul> </li> </ul>	Opcional
duration	El período de tiempo que debe transcurrir mientras el servicio de síntesis de voz lee el texto, en segundos o milisegundos. Por ejemplo, <code>2 s</code> o <code>1800 ms</code> .	Opcional
volumen	<p>Indica el nivel de volumen de la voz. Puede expresar el volumen como:</p> <ul style="list-style-type: none"> <li>• Un valor absoluto, expresado como un número en el rango de <code>0,0</code> a <code>100,0</code>, de <i>más silencioso</i> a <i>más alto</i>. Por ejemplo, <code>75</code>. El valor predeterminado es <code>100,0</code>.</li> <li>• Un valor relativo, expresado como un número precedido por <code>+</code> o <code>-</code> que especifica una cantidad para cambiar el volumen. Por ejemplo, <code>+10</code> o <code>-5,5</code>.</li> <li>• Un valor constante: <ul style="list-style-type: none"> <li>◦ <code>silent</code></li> <li>◦ <code>x-soft</code></li> <li>◦ <code>soft</code></li> <li>◦ <code>medio</code></li> <li>◦ <code>loud</code></li> <li>◦ <code>x-loud</code></li> <li>◦ <code>default</code></li> </ul> </li> </ul>	Opcional

### Cambio de la velocidad de habla

La velocidad de habla puede aplicarse a voces estándares en el nivel de palabra o frase. Sin embargo, solo puede aplicarse a voces neuronales en el nivel de oración.

### Ejemplo

```
<speak version="1.0" xmlns="https://www.w3.org/2001/10/synthesis" xml:lang="en-US">
  <voice name="en-US-Guy24kRUS">
    <prosody rate="+30.00%">
      Welcome to Microsoft Cognitive Services Text-to-Speech API.
    </prosody>
  </voice>
</speak>
```

## Cambio de volumen

Los cambios de volumen pueden aplicarse a voces estándar en el nivel de palabra o frase. Sin embargo, solo pueden aplicarse a voces neuronales en el nivel de frase.

### Ejemplo

```
<speak version="1.0" xmlns="https://www.w3.org/2001/10/synthesis" xml:lang="en-US">
  <voice name="en-US-Jessa24kRUS">
    <prosody volume="+20.00%">
      Welcome to Microsoft Cognitive Services Text-to-Speech API.
    </prosody>
  </voice>
</speak>
```

## Cambio de tono

Los cambios de tono pueden aplicarse a voces estándar en el nivel de palabra o frase. Sin embargo, solo pueden aplicarse a voces neuronales en el nivel de frase.

### Ejemplo

```
<speak version="1.0" xmlns="https://www.w3.org/2001/10/synthesis" xml:lang="en-US">
  <voice name="en-US-Guy24kRUS">
    Welcome to <prosody pitch="high">Microsoft Cognitive Services Text-to-Speech API.</prosody>
  </voice>
</speak>
```

## Cambio de curva melódica

### IMPORTANT

Los cambios de curva melódica no se admiten con voces neuronales.

### Ejemplo

```
<speak version="1.0" xmlns="https://www.w3.org/2001/10/synthesis" xml:lang="en-US">
  <voice name="en-US-Jessa24kRUS">
    <prosody contour="(80%,+20%) (90%,+30%)">
      Good morning.
    </prosody>
  </voice>
</speak>
```

## Elemento Say-as

`say-as` es un elemento opcional que indica el tipo de contenido (por ejemplo, el número o la fecha) del texto del elemento. De esta forma se proporcionan instrucciones al motor de síntesis de voz sobre cómo pronunciar el texto.

## Sintaxis

```
<say-as interpret-as="string" format="digit string" detail="string"> <say-as>
```

## Atributos

ATRIBUTO	DESCRIPCIÓN	OBLIGATORIO U OPCIONAL
interpret-as	Indica el tipo de contenido del texto del elemento. Para ver una lista de tipos, consulte la tabla siguiente.	Obligatorio
format	Proporciona información adicional sobre el formato preciso del texto del elemento para los tipos de contenido que pueden tener formatos ambiguos. SSML define formatos para los tipos de contenido que los usan (vea la tabla siguiente).	Opcional
detalles	Indica el nivel de detalle con que se va a hablar. Por ejemplo, este atributo puede solicitar que el motor de síntesis de voz pronuncie signos de puntuación. No hay valores estándar definidos para <code>detail</code> .	Opcional

A continuación se muestran los tipos de contenido admitidos para los atributos `interpret-as` y `format`.

Incluya el atributo `format` solo si `interpret-as` está establecido en la fecha y hora.

INTERPRET-AS	FORMAT	INTERPRETACIÓN
address		<p>El texto se pronuncia como una dirección. El motor de síntesis de voz pronuncia:</p> <div style="border: 1px solid black; padding: 5px;"><code>I'm at &lt;say-as interpret-as="address"&gt;150th CT NE, Redmond, WA&lt;/say-as&gt;</code></div> <p>Como "Estoy en el número 150 de Court North East Redmond, en Washington".</p>
cardinal, número		<p>El texto se pronuncia como un número cardinal. El motor de síntesis de voz pronuncia:</p> <div style="border: 1px solid black; padding: 5px;"><code>There are &lt;say-as interpret-as="cardinal"&gt;3&lt;/say-as&gt; alternatives</code></div> <p>Como "Hay tres alternativas".</p>

INTERPRET-AS	FORMAT	INTERPRETACIÓN
caracteres, ortografía		<p>El texto se pronuncia como letras individuales (deletreadas). El motor de síntesis de voz pronuncia:</p> <pre>&lt;say-as interpret-as="characters"&gt;test&lt;/say-as&gt;</pre> <p>Como "T E S T".</p>
date	dmy, mdy, ymd, ydm, ym, my, md, dm, d, m, y	<p>El texto se pronuncia como una fecha. El atributo <code>format</code> especifica el formato de la fecha (<math>d=\text{día}</math>, <math>m=\text{mes}</math>, <math>y=\text{año}</math>). El motor de síntesis de voz pronuncia:</p> <pre>Today is &lt;say-as interpret-as="date" format="mdy"&gt;10-19-2016&lt;/say-as&gt;</pre> <p>Como "Hoy es el diecinueve de octubre de 2016".</p>
digits, number_digit		<p>El texto se pronuncia como una secuencia de dígitos individuales. El motor de síntesis de voz pronuncia:</p> <pre>&lt;say-as interpret-as="number_digit"&gt;123456789&lt;/say-as&gt;</pre> <p>Como "1 2 3 4 5 6 7 8 9".</p>
fraction		<p>El texto se pronuncia como un número fraccionario. El motor de síntesis de voz pronuncia:</p> <pre>&lt;say-as interpret-as="fraction"&gt;3/8&lt;/say-as&gt; of an inch</pre> <p>Como "tres octavos de pulgada".</p>
ordinal		<p>El texto se pronuncia como un número ordinal. El motor de síntesis de voz pronuncia:</p> <pre>Select the &lt;say-as interpret-as="ordinal"&gt;3rd&lt;/say-as&gt; option</pre> <p>Como "Seleccionar la tercera opción".</p>

INTERPRET-AS	FORMAT	INTERPRETACIÓN
telephone		<p>El texto se pronuncia como un número de teléfono. El atributo <code>format</code> puede contener dígitos que representan un código de país. Por ejemplo, "1" para Estados Unidos o "39" para Italia. El motor de síntesis de voz puede utilizar esta información para orientar la pronunciación de un número de teléfono. El número de teléfono también puede incluir el código de país y, si es así, tiene prioridad sobre el código de país de <code>format</code>. El motor de síntesis de voz pronuncia:</p> <pre>The number is &lt;say-as interpret-as="telephone" format="1"&gt;(888) 555-1212&lt;/say-as&gt;</pre> <p>Como "Mi número es el código de área ocho ocho ocho cinco cinco cinco uno dos uno dos".</p>
time	hms12, hms24	<p>El texto se pronuncia como una hora. El atributo <code>format</code> especifica si la hora se especifica mediante un reloj de 12 horas (hms12) o de 24 horas (hms24). Use un signo de dos puntos para separar los números que representan las horas, los minutos y los segundos. Los siguientes son ejemplos de horas válidos: 12:35, 1:14:32, 08:15 y 02:50:45. El motor de síntesis de voz pronuncia:</p> <pre>The train departs at &lt;say-as interpret-as="time" format="hms12"&gt;4:00am&lt;/say-as&gt;</pre> <p>Como "El tren sale a las cuatro A. M."</p>

## Uso

El elemento `say-as` puede contener solo texto.

## Ejemplo

El motor de síntesis de voz pronuncia el ejemplo siguiente como "Su primera solicitud fue de una habitación el diecinueve de octubre a las diez y veinte con llegada a las cinco treinta y cinco P. M.".

```
<speak version="1.0" xmlns="https://www.w3.org/2001/10/synthesis" xml:lang="en-US">
  <voice name="en-US-Jessa24kRUS">
    <p>
      Your <say-as interpret-as="ordinal"> 1st </say-as> request was for <say-as interpret-as="cardinal"> 1
      </say-as> room
      on <say-as interpret-as="date" format="mdy"> 10/19/2010 </say-as>, with early arrival at <say-as
      interpret-as="time" format="hms12"> 12:35pm </say-as>.
    </p>
  </speak>
```

## Adición del audio grabado

`audio` es un elemento opcional que le permite insertar audio MP3 en un documento SSML. El cuerpo del elemento de audio puede contener texto sin formato o marcado SSML que se lee en voz alta si el archivo de audio no está disponible o no se puede reproducir. Además, el elemento `audio` puede contener texto y los siguientes elementos: `audio`, `break`, `p`, `s`, `phoneme`, `prosody`, `say-as` y `sub`.

Cualquier audio incluido en el documento SSML debe cumplir estos requisitos:

- El archivo MP3 debe estar hospedado en un punto de conexión HTTPS accesible desde Internet. Se debe usar HTTPS y el dominio que hospeda el archivo MP3 debe presentar un certificado SSL válido y de confianza.
- El archivo MP3 debe ser válido (MPEG V2).
- La velocidad de bits debe ser de 48 kbps.
- La frecuencia de muestreo debe ser de 16 000 Hz.
- El tiempo total combinado de todos los archivos de texto y audio de una sola respuesta no puede superar los noventa (90) segundos.
- El archivo MP3 no debe contener ninguna información específica del cliente ni otra que sea confidencial.

### Sintaxis

```
<audio src="string"/></audio>
```

### Atributos

ATRIBUTO	DESCRIPCIÓN	OBLIGATORIO U OPCIONAL
src	Especifica la ubicación o la URL del archivo de audio.	Es obligatorio si se usa el elemento de audio en el documento SSML.

### Ejemplo

```
<speak version="1.0" xmlns="https://www.w3.org/2001/10/synthesis" xml:lang="en-US">
  <p>
    <audio src="https://contoso.com/opinionprompt.wav"/>
    Thanks for offering your opinion. Please begin speaking after the beep.
    <audio src="https://contoso.com/beep.wav">
      Could not play the beep, please voice your opinion now. </audio>
    </p>
  </speak>
```

## Adición de audio de fondo

El elemento `mstts:backgroundaudio` permite agregar audio de fondo a los documentos SSML (o mezclar un archivo de audio con el servicio de texto a voz). Con `mstts:backgroundaudio`, puede reproducir en bucle un archivo de audio de fondo, hacer que aparezca gradualmente al principio de la conversión de texto a voz y quitarlo gradualmente al final de la conversión de texto a voz.

Si el audio de fondo proporcionado dura menos que la conversión de texto a voz o el fundido de salida, se reproducirá en bucle. Si dura más que la conversión de texto a voz, se detendrá cuando finalice el fundido de salida.

Solo se permite un archivo de audio de fondo por cada documento SSML. Sin embargo, puede intercalar etiquetas `audio` dentro del elemento `voice` para agregar audio adicional al documento SSML.

## Sintaxis

```
<mstts:backgroundaudio src="string" volume="string" fadein="string" fadeout="string"/>
```

## Atributos

ATRIBUTO	DESCRIPCIÓN	OBLIGATORIO U OPCIONAL
src	Especifica la ubicación o la URL del archivo de audio de fondo.	Es obligatorio si se usa el audio de fondo en el documento SSML.
volumen	Especifica el volumen del archivo de audio de fondo. <b>Valores aceptados:</b> de <code>0</code> a <code>100</code> , ambos incluidos. El valor predeterminado es <code>1</code> .	Opcional
fadein	Especifica la duración del fundido de entrada del audio de fondo en milisegundos. El valor predeterminado es <code>0</code> , que equivale a ningún fundido de entrada. <b>Valores aceptados:</b> de <code>0</code> a <code>10000</code> , ambos incluidos.	Opcional
fadeout	Especifica la duración del fundido de salida del audio de fondo en milisegundos. El valor predeterminado es <code>0</code> , que equivale a ningún fundido de salida. <b>Valores aceptados:</b> de <code>0</code> a <code>10000</code> , ambos incluidos.	Opcional

## Ejemplo

```
<speak version="1.0" xml:lang="en-US" xmlns:mstts="http://www.w3.org/2001/mstts">
    <mstts:backgroundaudio src="https://contoso.com/sample.wav" volume="0.7" fadein="3000"
    fadeout="4000"/>
    <voice name="Microsoft Server Speech Text to Speech Voice (en-US, Jessa24kRUS)">
        The text provided in this document will be spoken over the background audio.
    </voice>
</speak>
```

## Pasos siguientes

- [Compatibilidad de idiomas: voces, configuraciones regionales e idiomas](#)

# Introducción a voz personalizada

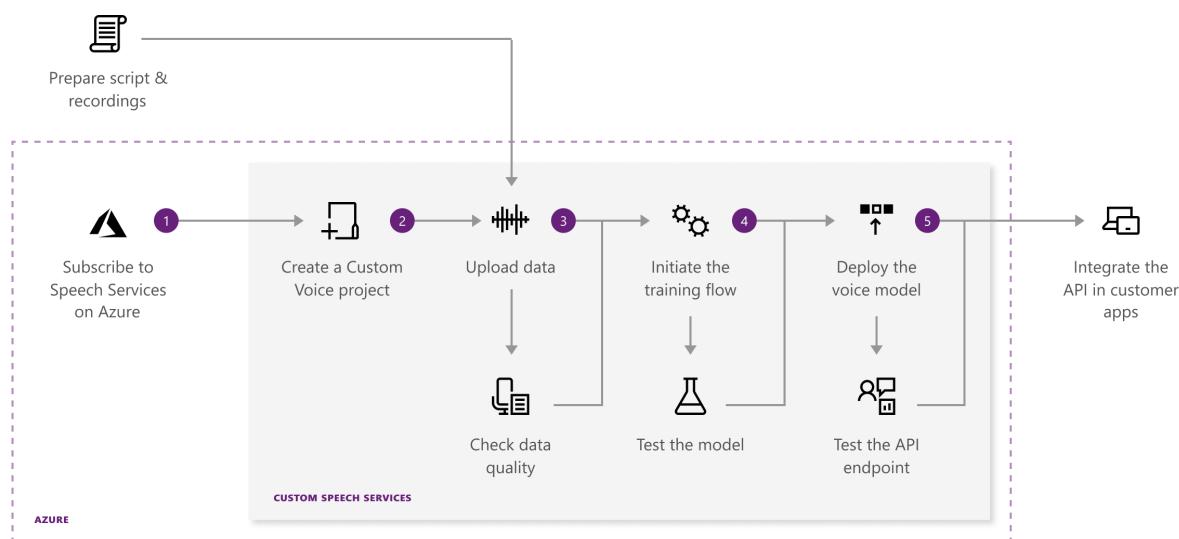
13/01/2020 • 6 minutes to read • [Edit Online](#)

**Custom Voice** es un conjunto de herramientas en línea que permiten crear una voz única y reconocible para su marca. Todo lo que se necesita para empezar son unos cuantos archivos de audio y las transcripciones asociadas. Siga los vínculos que se incluyen a continuación para empezar a crear una experiencia personalizada de conversión de texto a voz.

## ¿Qué incluye Custom Voice?

Antes de comenzar con Custom Voice, necesitará una cuenta de Azure y una suscripción del servicio de voz. Cuando haya creado una cuenta, podrá preparar los datos, entrenar y probar los modelos, evaluar la calidad de la voz y, finalmente, implementar el modelo de voz personalizado.

En el diagrama siguiente se resaltan los pasos necesarios para crear un modelo de voz personalizado desde el [portal de Custom Voice](#). Siga los vínculos para obtener más información.



1. **Suscríbase y cree un proyecto:** cree una cuenta de Azure y una suscripción del servicio de voz. Esta suscripción unificada proporciona acceso a la conversión de voz a texto, la conversión de texto a voz, la traducción de voz y el portal de Custom Voice. A continuación, mediante la suscripción al servicio de voz, cree su primer proyecto de Voz personalizada.
2. **Cargar datos:** carga de datos (audio y texto) mediante el portal de Custom Voice o una API de voz personalizada. Desde el portal, puede investigar y evaluar las puntuaciones de pronunciación y las relaciones de señal a ruido. Para más información, consulte el artículo sobre la [preparación de datos para Custom Voice](#).
3. **Entrenar el modelo:** utilice los datos para crear un modelo de voz de conversión de texto a voz personalizada. Puede entrenar un modelo en diferentes idiomas. Después del entrenamiento, pruebe el modelo y, si le satisface el resultado, ya puede implementar el modelo.
4. **Implementar el modelo:** cree un punto de conexión personalizado para el modelo de voz de conversión de texto a voz y utilícelo para la síntesis de voz en sus productos, herramientas y aplicaciones.

## Voces neuronales personalizadas

La funcionalidad de personalización de voz neuronal se encuentra actualmente en versión preliminar, limitada a clientes seleccionados. Rellene este [formulario de aplicación](#) para empezar.

#### NOTE

Como parte del compromiso de Microsoft de diseñar inteligencia artificial responsable, nuestra intención es proteger los derechos de los individuos y la sociedad, y fomentar unas interacciones transparentes de personas y equipos. Por esta razón, Voz neuronal personalizada no está disponible con carácter general para todos los clientes. Es posible que obtenga acceso a la tecnología solo tras la revisión de sus aplicaciones y después de que se haya comprometido a usarla en consonancia con nuestros principios éticos. Obtenga más información sobre nuestro [proceso de acceso según la aplicación](#).

## Configuración de la cuenta de Azure

Para poder usar el portal de Custom Speech y crear un modelo personalizado, se necesita una suscripción al servicio de voz. Siga estas instrucciones para crear una suscripción al servicio de voz en Azure. Si no tiene una cuenta de Azure, puede registrarse para obtener una nueva.

Después de crear una cuenta de Azure y la suscripción al servicio de voz, deberá iniciar sesión en el portal de Voz personalizada y conectarse a su suscripción.

1. Obtenga la clave de la suscripción del servicio de voz en Azure Portal.
2. Inicie sesión en el [portal de Custom Voice](#).
3. Seleccione la suscripción y cree un proyecto de voz.
4. Si desea cambiar a otra suscripción de voz, utilice el icono de engranaje situado en el panel de navegación superior.

#### NOTE

El servicio Custom Voice no admite la clave de evaluación gratuita de 30 días. Debe tener una clave F0 o S0 creada en Azure para poder usar el servicio.

## Creación de un proyecto

El contenido como datos, modelos, pruebas y puntos finales se organizan en **proyectos** en el portal de Custom Voice. Cada proyecto es específico de un idioma o país y del género de la voz que desea crear. Por ejemplo, puede crear un proyecto de voz femenina para los bots de chat del centro de llamadas que utilizan el inglés de Estados Unidos (en-US).

Para crear su primer proyecto, seleccione la pestaña **Text-to-Speech/Custom Voice** (Conversión de texto a voz/Conversión de voz personalizada) y, después, haga clic en **New Project** (Nuevo proyecto). Siga las instrucciones del asistente para crear el proyecto. Después de crear el proyecto, verá cuatro pestañas: **Data** (Datos), **Training** (Entrenamiento), **Testing** (Pruebas) y **Deployment** (Implementación). Use los vínculos incluidos en [Pasos siguientes](#) para aprender a usar cada pestaña.

## Pasos siguientes

- [Preparación de datos de voz personalizado](#)
- [Creación de una voz personalizada](#)
- [Guía: Grabación de muestras de voz](#)

# Long Audio API (versión preliminar)

13/01/2020 • 3 minutes to read • [Edit Online](#)

Long Audio API está diseñada para la síntesis asincrónica de texto de formato largo a voz (por ejemplo: audiolibros). Esta API no devuelve audio sintetizado en tiempo real, sino que, en su lugar, se espera que se sondeen las respuestas y se consuman las salidas a medida que estén disponibles desde el servicio. A diferencia de Text To Speech API, que usa el SDK de Voz, Long Audio API puede crear audio sintetizado de más de diez minutos, lo que resulta idóneo para los editores y las plataformas de contenido de audio.

Ventajas adicionales de Long Audio API:

- La voz sintetizada devuelta por el servicio utiliza voces neuronales, lo que garantiza salidas de audio de alta fidelidad.
- Dado que no se admiten las respuestas en tiempo real, no es necesario implementar un punto de conexión de voz.

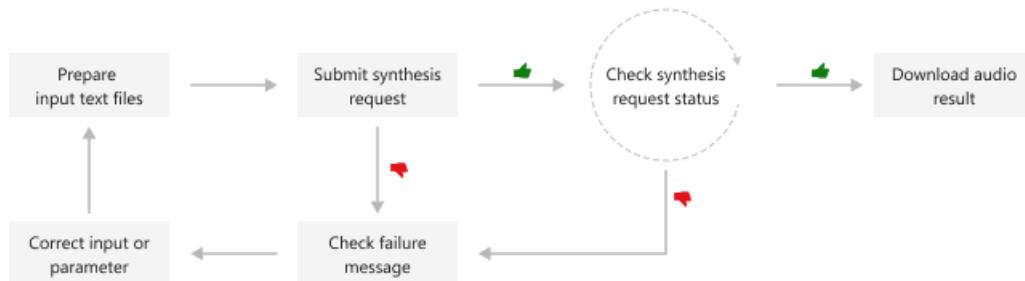
## NOTE

Long Audio API ahora solo admite [Voz neuronal personalizada](#).

## Flujo de trabajo

Normalmente, cuando use Long Audio API, enviará un archivo de texto o archivos que se sintetizarán, sondeará el estado y, a continuación, si el estado es correcto, podrá descargar la salida de audio.

El diagrama a continuación proporciona una introducción general del flujo de trabajo.



## Preparación del contenido para la síntesis

Cuando准备 el archivo de texto, asegúrese de lo siguiente:

- Se trata de texto sin formato (.txt) o texto SSML (.txt).
  - En el caso de texto sin formato, cada párrafo se separa al presionar **Entrar**: consulte un [ejemplo de entrada de texto sin formato](#).
  - En el caso de texto SSML, cada fragmento de SSML se considera un párrafo. Los fragmentos de SSML se deben separar con párrafos distintos: consulte un [ejemplo de entrada de texto SSML](#). Para el código de idioma, consulte [Lenguaje de marcado de síntesis de voz \(SSML\)](#).
- Está codificado como [UTF-8 con marca BOM](#).
- Contiene más de 10 000 caracteres o más de 50 párrafos

- Es un solo archivo, no un archivo ZIP.

## Formatos de salida de audio

Long Audio API admite los siguientes formatos de salida de audio:

### NOTE

El formato de audio predeterminado es riff-16khz-16bit-mono-pcm.

- riff-8khz-16bit-mono-pcm
- riff-16khz-16bit-mono-pcm
- riff-24khz-16bit-mono-pcm
- riff-48khz-16bit-mono-pcm
- audio-16khz-32kbitrate-mono-mp3
- audio-16khz-64kbitrate-mono-mp3
- audio-16khz-128kbitrate-mono-mp3
- audio-24khz-48kbitrate-mono-mp3
- audio-24khz-96kbitrate-mono-mp3
- audio-24khz-160kbitrate-mono-mp3

## Guías de inicio rápido

Se ofrecen guías de inicio rápido diseñadas para ayudarle a ejecutar Long Audio API correctamente. Esta tabla incluye una lista de inicios rápidos de Long Audio API en función del idioma.

- [Inicio rápido: Python](#)

## Código de ejemplo

Hay un ejemplo de código para Long Audio API disponible en GitHub.

- [Código de ejemplo: Python](#)
- [Código de ejemplo: C#](#)
- [Código de ejemplo: Java](#)

# Instalación y ejecución de contenedores del servicio de voz (versión preliminar)

14/01/2020 • 38 minutes to read • [Edit Online](#)

Los contenedores le permiten ejecutar algunas de las API del servicio de voz en su propio entorno. Los contenedores son excelentes para requisitos específicos de control de datos y seguridad. En este artículo, aprenderá a descargar, instalar y ejecutar un contenedor de Voz.

Los contenedores de Voz permiten a los clientes compilar una arquitectura de aplicación de voz optimizada para las sólidas funcionalidades de la nube y la localidad del perímetro. Hay cuatro contenedores distintos disponibles. Los dos contenedores estándar son **conversión de voz a texto** y **conversión de texto a voz**. Los dos contenedores personalizados son **conversión de voz a texto personalizada** y **conversión de texto a voz personalizada**.

## IMPORTANT

Actualmente, todos los contenedores de voz se ofrecen como parte de una [versión preliminar pública "validada"](#). Se hará un anuncio cuando los contenedores de voz pasen a la disponibilidad general (GA).

FUNCIÓN	CARACTERÍSTICAS	MÁS RECIENTE
Voz a texto	Transcribe registros continuos de voz en tiempo real o de audio por lotes a texto con resultados intermedios.	2.0.0
Conversión de voz a texto personalizada	Con un modelo personalizado del <a href="#">portal de Voz personalizada</a> , transcribe las grabaciones continuas de voz en tiempo real o de audio por lotes a texto con resultados inmediatos.	2.0.0
Texto a voz	Convierte texto a voz de sonido natural con entrada de texto sin formato o Lenguaje de marcado de síntesis de voz (SSML).	1.3.0
Conversión de texto a voz personalizada	Con un modelo personalizado del <a href="#">portal de Voz personalizada</a> , convierte texto a voz de sonido natural con entrada de texto sin formato o Lenguaje de marcado de síntesis de voz (SSML).	1.3.0

Si no tiene una suscripción a Azure, cree una [cuenta gratuita](#) antes de empezar.

## Prerequisites

Requisitos previos para poder usar los contenedores de Voz:

OBLIGATORIO	PROPÓSITO
-------------	-----------

OBLIGATORIO	PROPÓSITO
Motor de Docker	<p>Necesita que el motor de Docker esté instalado en un <a href="#">equipo host</a>. Docker dispone de paquetes que configuran el entorno de Docker en <a href="#">macOS</a>, <a href="#">Windows</a> y <a href="#">Linux</a>. Para conocer los principios básicos de Docker y de los contenedores, consulte <a href="#">Introducción a Docker</a>.</p> <p>Docker debe configurarse para permitir que los contenedores se conecten con Azure y envíen datos de facturación a dicho servicio.</p> <p><b>En Windows</b>, Docker también debe estar configurado de forma que admita los contenedores de Linux.</p>
Conocimientos sobre Docker	<p>Debe tener conocimientos básicos sobre los conceptos de Docker, como los registros, los repositorios, los contenedores y las imágenes de contenedor, así como conocer los comandos <code>docker</code> básicos.</p>
Recurso de Voz	<p>Para usar estos contenedores, debe tener:</p> <p>Recurso de Voz de Azure para obtener la clave de API y el URI de punto de conexión asociados. Ambos valores están disponibles en las páginas Introducción y Claves de <b>Voz</b> de Azure Portal. Los dos son necesarios para iniciar el contenedor.</p> <p><b>{API_KEY}</b> : una de las dos claves de recurso disponibles en la página <b>Claves</b></p> <p><b>{ENDPOINT_URI}</b> : el punto de conexión tal como se proporciona en la página de <b>Información general</b>.</p>

## Solicitud de acceso al registro de contenedor

Rellene y envíe el [formulario de solicitud de contenedores de Voz de Cognitive Services](#) para solicitar acceso al contenedor.

El formulario solicita información acerca del usuario y de su empresa, así como del escenario de usuario para el que se va a usar el contenedor. Después de haber enviado el formulario, el equipo de Azure Cognitive Services lo revisa para asegurarse de que cumple los criterios de acceso al registro de contenedor privado.

### IMPORTANT

Debe usar una dirección de correo electrónico que esté asociada con una cuenta de Microsoft (MSA) o de Azure Active Directory (Azure AD) en el formulario.

Si se aprueba la solicitud, recibirá un correo electrónico con instrucciones que describen cómo obtener las credenciales y tener acceso al registro de contenedor privado.

## Uso de la CLI de Docker para autenticar un registro de contenedor privado

Puede autenticarse con el registro de contenedor privado de contenedores de Cognitive Services, pero el método recomendado de la línea de comandos es mediante el uso de la [CLI de Docker](#).

Use el comando `docker login`, como se muestra en el ejemplo siguiente, para iniciar sesión en `containerpreview.azurecr.io`, el registro de contenedor privado de los contenedores de Cognitive Services.

Reemplace `<username>` por el nombre de usuario y `<password>` por la contraseña proporcionada en las credenciales que recibió del equipo de Azure Cognitive Services.

```
docker login containerpreview.azurecr.io -u <username> -p <password>
```

Si ha protegido las credenciales en un archivo de texto, puede concatenar el contenido de dicho archivo de texto, mediante el comando `cat`, al comando `docker login`, tal como se muestra en el ejemplo siguiente. Reemplace `<passwordFile>` por la ruta de acceso y el nombre del archivo de texto que contiene la contraseña, y `<username>` por el nombre de usuario proporcionado en las credenciales.

```
cat <passwordFile> | docker login containerpreview.azurecr.io -u <username> --password-stdin
```

## Recopilación de los parámetros obligatorios

Hay tres parámetros principales para todos los contenedores de Cognitive Services que son necesarios. El contrato de licencia para el usuario final (CLUF) debe estar presente con un valor de `accept`. Además, se necesitan una dirección URL de punto de conexión y una clave de API.

### URI de punto de conexión {ENDPOINT\_URI}

El valor del URI del **punto de conexión** está disponible en la página *Información general* de Azure Portal del recurso de Cognitive Services correspondiente. Vaya a la página *Información general*, mantenga el cursor sobre el punto de conexión y aparecerá un icono `Copy to clipboard`. Cópielo y utilícelo cuando sea necesario.

The screenshot shows the Azure Portal interface for a Cognitive Services resource named "widgets". On the left, there's a sidebar with navigation links like Home, widgets, Search (Ctrl+/,), Activity log, Access control (IAM), Tags, and Diagnose and solve problems. Below that is a RESOURCE MANAGEMENT section with Keys, Quick start, Pricing tier, and Billing By Subscription. The main content area is titled "Overview" (with a red arrow pointing to it). It displays resource details: Resource group (widgets-resource-group), Status (Active), Location (North Central US), Subscription (widgets-subscription), Subscription ID (xxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxx), and Tags (Click here to add tags). To the right, there are sections for API type (<API Type>), Pricing tier (Standard), and Endpoint (https://widgets.cognitiveservices.azure.com/api/example-endpoint). A "Copy to clipboard" button is highlighted with a red box next to the Endpoint URL.

### Claves {API\_KEY}

Esta clave se usa para iniciar el contenedor y está disponible en la página de claves de Azure Portal del recurso de Cognitive Services correspondiente. Vaya a la página *Claves* y haga clic en el icono `Copy to clipboard`.

The screenshot shows the Azure Cognitive Services - Keys page. On the left, there's a sidebar with options like Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, RESOURCE MANAGEMENT, and Keys (which has a red arrow pointing to it). The main area shows a 'NAME' field with 'widgets'. Below it is a note about subscription keys: 'These subscription keys are used to access your Cognitive Service API. Do not share your keys. Store them securely— for example, using Azure Key Vault. We also recommend regenerating these keys regularly. Only one key is necessary to make an API call. When regenerating the first key, you can use the second key for continued access to the service.' Under 'KEY 1', there's a field with '<key 1 value>' and a red box around the copy icon. Under 'KEY 2', there's a field with '<key 2 value>' and a red box around the copy icon.

## IMPORTANT

Estas claves de suscripción se usan para tener acceso a la API de Cognitive Services. No comparta las claves. Almacénelas de forma segura, por ejemplo, con Azure Key Vault. También se recomienda regenerar estas claves periódicamente. Solo se necesita una clave para realizar una llamada API. Al volver a generar la primera clave, puede usar la segunda clave para seguir teniendo acceso al servicio.

## El equipo host

El host es un equipo basado en x64 que ejecuta el contenedor de Docker. Puede ser un equipo del entorno local o un servicio de hospedaje de Docker incluido en Azure, como:

- [Azure Kubernetes Service](#).
- [Azure Container Instances](#).
- Un clúster de [Kubernetes](#) implementado en [Azure Stack](#). Para obtener más información, consulte [Implementación de Kubernetes en Azure Stack](#).

## Compatibilidad con la extensión avanzada de vector

El **host** es el equipo que ejecuta el contenedor de Docker. El host *debe ser compatible* con las [extensiones avanzadas de vector](#) (AVX2). Puede comprobar la compatibilidad con AVX2 en los hosts de Linux con el comando siguiente:

```
grep -q avx2 /proc/cpuinfo && echo AVX2 supported || echo No AVX2 support detected
```

## WARNING

El equipo host es *necesario* para admitir AVX2. El contenedor *no* funcionará correctamente sin compatibilidad con AVX2.

## Recomendaciones y requisitos del contenedor

En la tabla siguiente se describe la asignación mínima y recomendada de recursos para cada contenedor de Voz.

- [Voz a texto](#)
- [Conversión de voz a texto personalizada](#)
- [Texto a voz](#)
- [Conversión de texto a voz personalizada](#)

CONTENEDOR	MÍNIMA	RECOMENDADO
Voz a texto	2 núcleos, 2 GB de memoria	4 núcleos, 4 GB de memoria

- Cada núcleo debe ser de 2,6 gigahercios (GHz) como mínimo.

El núcleo y la memoria se corresponden con los valores de `--cpus` y `--memory` que se usan como parte del comando `docker run`.

#### NOTE

Los valores mínimos y recomendados se basan en los límites de Docker y *no* en los recursos de la máquina host. Por ejemplo, partes de la asignación de memoria de contenedores de voz a texto de un modelo grande de lenguaje, y se *recomienda* que todo el archivo se ajuste en memoria, que es de 4 a 6 GB adicionales. Además, la primera ejecución de cualquier contenedor puede tardar más, dado que los modelos se van a paginar en la memoria.

## Obtención de la imagen del contenedor con `docker pull`

Las imágenes de contenedor para Voz están disponibles en la instancia de Container Registry siguiente.

- [Voz a texto](#)
- [Conversión de voz a texto personalizada](#)
- [Texto a voz](#)
- [Conversión de texto a voz personalizada](#)

CONTENEDOR	REPOSITORIO
Voz a texto	<code>containerpreview.azurecr.io/microsoft/cognitive-services-speech-to-text:latest</code>

#### TIP

Puede usar el comando `docker images` para enumerar las imágenes de contenedor descargadas. Por ejemplo, el comando siguiente muestra el id., el repositorio y la etiqueta de cada imagen de contenedor descargada, con formato de tabla:

```
docker images --format "table {{.ID}}\t{{.Repository}}\t{{.Tag}}"
IMAGE ID      REPOSITORY          TAG
<image-id>   <repository-path/name>  <tag-name>
```

## Docker pull para los contenedores de Voz

- [Voz a texto](#)
- [Conversión de voz a texto personalizada](#)
- [Texto a voz](#)
- [Conversión de texto a voz personalizada](#)

### Docker pull para el contenedor de conversión de voz a texto

Use el comando `docker pull` para descargar una imagen de contenedor desde la versión preliminar del registro de contenedor.

```
docker pull containerpreview.azurecr.io/microsoft/cognitive-services-speech-to-text:latest
```

## IMPORTANT

La etiqueta `latest` extrae la configuración regional `en-US`. Para otras configuraciones regionales, consulte [Configuración regional de conversión de voz a texto](#).

### Configuraciones regionales de voz a texto

Todas las etiquetas, a excepción de `latest` tienen el formato siguiente y distinguen mayúsculas de minúsculas:

```
<major>.<minor>.<patch>-<platform>-<locale>-<prerelease>
```

La etiqueta siguiente es un ejemplo del formato:

```
2.0.0-amd64-en-us-preview
```

Para ver todas las configuraciones regionales admitidas del contenedor de **conversión de voz a texto**, consulte las [etiquetas de imágenes de la conversión de voz a texto](#).

## Uso del contenedor

Una vez que el contenedor esté en el [equipo host](#), utilice el siguiente proceso para trabajar con el contenedor.

1. Ejecute el contenedor con la configuración de facturación requerida. Hay más [ejemplos](#) del comando `docker run` disponibles.
2. Consulta del punto de conexión de predicción del contenedor.

### Ejecute el contenedor con `docker run`.

Utilice el comando `docker run` para ejecutar el contenedor. Consulte [Recopilación de los parámetros obligatorios](#) para más información sobre cómo obtener los valores de `{Endpoint_URI}` y `{API_Key}`. También hay disponibles otros [ejemplos](#) del comando `docker run`.

- [Voz a texto](#)
- [Conversión de voz a texto personalizada](#)
- [Texto a voz](#)
- [Conversión de texto a voz personalizada](#)

Para ejecutar el contenedor *Conversión de voz a texto*, ejecute el comando `docker run` siguiente.

```
docker run --rm -it -p 5000:5000 --memory 4g --cpus 4 \
containerpreview.azurecr.io/microsoft/cognitive-services-speech-to-text \
Eula=accept \
Billing={ENDPOINT_URI} \
ApiKey={API_KEY}
```

Este comando:

- Ejecuta un contenedor *Conversión de voz a texto* desde la imagen de contenedor.
- Asigna 4 núcleos de CPU y 4 gigabytes (GB) de memoria.
- Expone el puerto TCP 5000 y asigna un seudo-TTY para el contenedor.
- Una vez que se produce la salida, quita automáticamente el contenedor. La imagen del contenedor sigue estando disponible en el equipo host.

#### IMPORTANT

Para poder ejecutar el contenedor, las opciones `Eula`, `Billing` y `ApiKey` deben estar especificadas; de lo contrario, el contenedor no se iniciará. Para obtener más información, vea [Facturación](#).

## Consulta del punto de conexión de predicción del contenedor

CONTENEDORES	DIRECCIÓN URL DEL HOST DEL SDK	PROTOCOLO
Conversión de voz en texto y conversión de voz en texto personalizada	<code>ws://localhost:5000</code>	WS
Conversión de texto a voz y conversión de texto a voz personalizada	<code>http://localhost:5000</code>	HTTP

Para más información sobre cómo usar los protocolos WSS y HTTPS, consulte la [seguridad del contenedor](#).

### Conversión de voz a texto o Conversión de voz a texto personalizada

El contenedor proporciona las API de punto de conexión de consulta basadas en WebSocket, a las que se accede mediante el [SDK de Voz](#). De forma predeterminada, el SDK de Voz usa servicios de voz en línea. Para usar el contenedor, deberá cambiar el método de inicialización.

#### TIP

Al usar el SDK de Voz con los contenedores, no es necesario proporcionar la [clave de suscripción del recurso de Voz de Azure](#) o un [token de portador de autenticación](#).

Consulte los ejemplos siguientes.

- [C#](#)
- [Python](#)

Cambie de usar esta llamada de inicialización en la nube de Azure:

```
var config = SpeechConfig.FromSubscription("YourSubscriptionKey", "YourServiceRegion");
```

por esta llamada mediante el [host](#) de contenedor:

```
var config = SpeechConfig.FromHost(  
    new Uri("ws://localhost:5000"));
```

### Conversión de texto a voz o conversión de texto a voz personalizada

El contenedor proporciona [API de punto de conexión basadas en REST](#). Hay muchos [proyectos de código fuente de ejemplo](#) disponibles para las variaciones de lenguaje, marco y plataforma.

Con el contenedor *Conversión de texto a voz estándar*, debe basarse en la configuración regional y en la voz de la etiqueta de imagen que descargó. Por ejemplo, si descargó la etiqueta `latest`, la configuración regional predeterminada es `en-US` y la voz `JessaRUS`. El argumento `{VOICE_NAME}` sería `en-US-JessaRUS`. Vea el SSML de ejemplo siguiente:

```
<speak version="1.0" xml:lang="en-US">
    <voice name="en-US-JessaRUS">
        This text will get converted into synthesized speech.
    </voice>
</speak>
```

Sin embargo, para *Conversión de texto a voz personalizada*, tendrá que obtener el valor de **Voice / model** desde el [portal de Voz personalizada](#). El nombre del modelo personalizado es sinónimo del nombre de la voz. Vaya a la página de **entrenamiento** y copie el valor de **Voice / model** que se va a usar como el argumento `{VOICE_NAME}`.

Vea el SSML de ejemplo siguiente:

```
<speak version="1.0" xml:lang="en-US">
    <voice name="custom-voice-model">
        This text will get converted into synthesized speech.
    </voice>
</speak>
```

Vamos a crear una solicitud HTTP POST, proporcionando algunos encabezados y una carga de datos. Reemplace el marcador de posición `{VOICE_NAME}` por un valor propio.

```
curl -s -v -X POST http://localhost:5000/speech/synthesize/cognitiveservices/v1 \
-H 'Accept: audio/*' \
-H 'Content-Type: application/ssml+xml' \
-H 'X-Microsoft-OutputFormat: riff-16khz-16bit-mono-pcm' \
-d '<speak version="1.0" xml:lang="en-US"><voice name="{VOICE_NAME}">This is a test, only a test.</voice>
</speak>'
```

Este comando:

- Construye una solicitud HTTP POST para el punto de conexión `speech/synthesize/cognitiveservices/v1`.
- Especifica un encabezado `Accept` de `audio/*`
- Especifica un encabezado `Content-Type` de `application/ssml+xml`. Para más información, consulte el [cuerpo de la solicitud](#).
- Especifica un encabezado `X-Microsoft-OutputFormat` de `riff-16khz-16bit-mono-pcm`. Para más opciones, consulte la [salida de audio](#).
- Envía la solicitud [Lenguaje de marcado de síntesis de voz \(SSML\)](#) dado el `{VOICE_NAME}` al punto de conexión.

## Ejecución de varios contenedores en el mismo host

Si tiene pensado ejecutar varios contenedores con puertos expuestos, asegúrese de que ejecuta cada contenedor con un puerto expuesto diferente. Por ejemplo, ejecute el primer contenedor en el puerto 5000 y el segundo en el

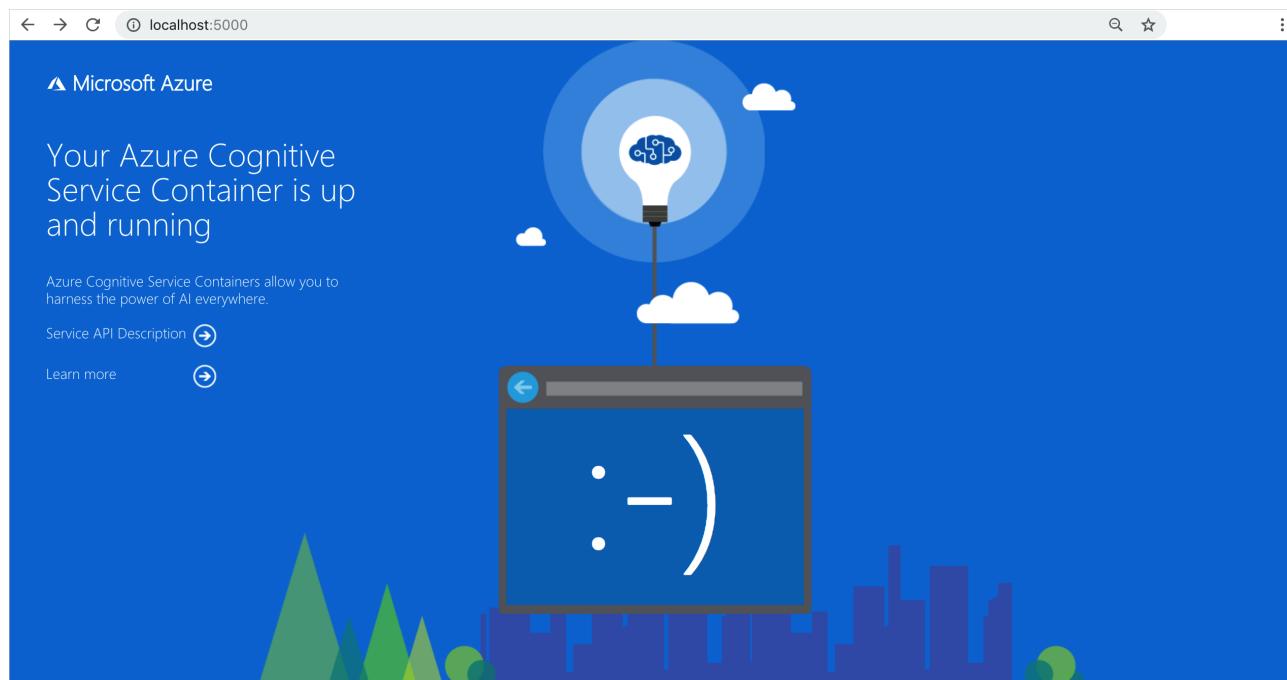
puerto 5001.

Puede tener este contenedor y un contenedor de Azure Cognitive Services diferente en ejecución simultáneamente en el HOST. También puede tener varios contenedores del mismo contenedor de Cognitive Services en ejecución.

## Comprobación de que un contenedor está en ejecución

Hay varias maneras de comprobar que el contenedor está en ejecución. Busque la dirección *IP externa* y el puerto expuesto del contenedor en cuestión y abra el explorador web que prefiera. Use las distintas direcciones URL de solicitud para validar que el contenedor se está ejecutando. Las direcciones URL de solicitud de ejemplo que se enumeran a continuación son `http://localhost:5000`, pero el contenedor específico puede variar. Tenga en cuenta que va a confiar en la *dirección IP externa* y el puerto expuesto del contenedor.

URL DE LA SOLICITUD	PROPOSITO
<code>http://localhost:5000/</code>	El contenedor ofrece una página principal.
<code>http://localhost:5000/status</code>	Se solicitó con HTTP GET, para comprobar que el contenedor está en ejecución sin causar una consulta al punto de conexión. Esta solicitud se puede usar con los <a href="#">sondeos de ejecución y preparación</a> de Kubernetes.
<code>http://localhost:5000/swagger</code>	El contenedor cuenta con un completo conjunto de documentación sobre los puntos de conexión y una característica de <b>prueba</b> . Esta característica le permite especificar la configuración en un formulario HTML basado en web y realizar la consulta sin necesidad de escribir código. Una vez que la consulta devuelve resultados, se proporciona un ejemplo del comando CURL para mostrar los encabezados HTTP y el formato de cuerpo requeridos.



## Detención del contenedor

Para apagar el contenedor, en el entorno de la línea de comandos donde se ejecuta el contenedor, seleccione Ctrl + C.

# Solución de problemas

Puede encontrar algunos problemas al iniciar o ejecutar el contenedor. Use un [montaje](#) de salida y habilite el registro. Si lo hace, permitirá que el contenedor genere archivos de registro que son útiles para solucionar problemas.

## TIP

Para más información e instrucciones para solución de problemas, vea [Preguntas frecuentes de los contenedores de Cognitive Services](#).

## Facturación

Los contenedores de Voz envían información de facturación a Azure mediante un recurso de *Voz* en la cuenta de Azure.

Las consultas en el contenedor se facturan con el plan de tarifa del recurso de Azure que se usa para <ApiKey>.

Los contenedores de Azure Cognitive Services no tienen licencia para ejecutarse si no están conectados al punto de conexión de facturación para las mediciones. Debe habilitar los contenedores para que comuniquen la información de facturación al punto de conexión de facturación en todo momento. Los contenedores de Cognitive Services no envían datos de los clientes (por ejemplo, la imagen o el texto que se está analizando) a Microsoft.

### Conexión con Azure

El contenedor necesita que se ejecuten los valores del argumento de facturación. Estos valores permiten al contenedor conectarse al punto de conexión de facturación. El contenedor informa sobre el uso cada 10 a 15 minutos. Si el contenedor no se conecta a Azure en la ventana de tiempo permitida, continuará ejecutándose pero no atenderá las consultas hasta que se restaure el punto de conexión de facturación. Se intenta 10 veces la conexión en el mismo intervalo de tiempo de 10 a 15 minutos. Si no se puede conectar con el punto de conexión de facturación en esos 10 intentos, el contenedor deja de ejecutarse.

### Argumentos de facturación

Para que el comando `docker run` inicie el contenedor, se deben especificar las tres opciones siguientes se deben especificar con valores válidos:

OPCIÓN	DESCRIPCIÓN
<code>ApiKey</code>	La clave de API del recurso de Cognitive Services que se usa para realizar un seguimiento de la información de facturación. El valor de esta opción se debe establecer en una clave de API para el recurso aprovisionado que se especifica en <code>Billing</code> .
<code>Billing</code>	El punto de conexión del recurso de Cognitive Services que se usa para realizar el seguimiento de la información de facturación. El valor de esta opción debe establecerse en el URI del punto de conexión de un recurso aprovisionado de Azure.
<code>Eula</code>	Indica que ha aceptado la licencia del contenedor. El valor de esta opción debe establecerse en <code>accept</code> .

Para obtener más información acerca de estas opciones, consulte [Configure containers](#) (Configuración de contenedores).

## Publicaciones de blog

- Ejecución de contenedores de Cognitive Services
- Azure Cognitive Services

## Ejemplos para desarrolladores

Hay ejemplos para desarrolladores disponibles en nuestro [repositorio de GitHub](#).

## Ver seminario web

Únase al [seminario web](#) para más información sobre:

- Implementación de Cognitive Services en cualquier máquina con Docker
- Implementación de Cognitive Services en AKS

## Resumen

En este artículo, ha aprendido los conceptos y el flujo de trabajo para la descarga, instalación y ejecución de contenedores de Voz. En resumen:

- Voz proporciona cuatro contenedores Linux para Docker, que encapsulan varias funcionalidades:
  - *Voz a texto*
  - *Conversión de voz a texto personalizada*
  - *Texto a voz*
  - *Conversión de texto a voz personalizada*
- Las imágenes de contenedor se descargan desde el registro de contenedor de Azure.
- Las imágenes de contenedor se ejecutan en Docker.
- Puede usar la API REST o el SDK para llamar a operaciones en contenedores de Voz mediante la especificación del URI del host del contenedor.
- Debe proporcionar la información de facturación al crear una instancia de un contenedor.

### IMPORTANT

Los contenedores de Cognitive Services no tienen licencia para ejecutarse sin estar conectados a Azure para realizar mediciones. Los clientes tienen que habilitar los contenedores para comunicar la información de facturación con el servicio de medición en todo momento. Los contenedores de Cognitive Services no envían datos de los clientes (por ejemplo, la imagen o el texto que se está analizando) a Microsoft.

## Pasos siguientes

- Revise [Configuración de contenedores](#) para ver las opciones de configuración.
- Aprenda a [usar contenedores del servicio de voz con Kubernetes y Helm](#).
- Uso de otros [contenedores de Cognitive Services](#)

# Notas de la versión

15/01/2020 • 33 minutes to read • [Edit Online](#)

## SDK de Voz 1.8.0: Versión de noviembre de 2019

### Nuevas características

- Se ha agregado una API `FromHost()` para facilitar su uso con contenedores locales y nubes soberanas.
- Se ha agregado la detección automática de idioma de origen para el reconocimiento de voz (en Java y C++)
- Se ha agregado el objeto `SourceLanguageConfig` para el reconocimiento de voz, que se usa para especificar los idiomas de origen esperados (en Java y C++).
- Se ha agregado compatibilidad con `KeywordRecognizer` en Windows (UWP), Android e iOS mediante los paquetes de Nuget y Unity.
- Se ha agregado la API de Java de conversación remota para realizar la transcripción de conversaciones en lotes asíncronos.

### Cambios importantes

- Las funcionalidades de transcripción de conversaciones se han movido al espacio de nombres `Microsoft.CognitiveServices.Speech.Transcription`.
- Parte de los métodos de transcripción de conversaciones se han movido a la nueva clase `Conversation`.
- Compatibilidad eliminada para iOS de 32 bits (ARMv7 y x86)

### Correcciones de errores

- Se ha corregido un bloqueo si se usa `KeywordRecognizer` local sin una clave de suscripción válida al servicio de voz.

### Muestras

- Ejemplo de Xamarin para `KeywordRecognizer`
- Ejemplo de Unity para `KeywordRecognizer`
- Ejemplos de C++ y Java de detección automática de idioma de origen

## SDK de voz 1.7.0: versión de septiembre de 2019

### Nuevas características

- Compatibilidad con la versión beta agregada para Xamarin en la Plataforma universal de Windows (UWP), Android e iOS
- Compatibilidad con iOS agregada para Unity
- Se ha agregado compatibilidad con entradas `Compressed` para ALaw, Mulaw, FLAC en Android, iOS y Linux.
- Se ha agregado `SendMessageAsync` en la clase `Connection` para enviar un mensaje al servicio.
- Se ha agregado `SetMessageProperty` en la clase `Connection` para establecer la propiedad de un mensaje.
- TTS agregó compatibilidad para Java (JRE y Android), Python, Swift y Objective-C
- TTS agregó compatibilidad de reproducción para macOS, iOS y Android
- Se ha agregado información de "límite de palabras" para TTS

### Correcciones de errores

- Se ha corregido un problema de compilación de IL2CPP en Unity 2019 para Android

- Se ha corregido un problema con los encabezados con formato incorrecto en la entrada de archivo WAV que se procesa de forma incorrecta
- Se ha corregido un problema con UUID que no es único en algunas propiedades de conexión
- Se han corregido algunas advertencias sobre los especificadores de nulabilidad en los enlaces SWIFT (puede que se requieran pequeños cambios en el código)
- Se ha corregido un error que provocaba que las conexiones de WebSocket se cerraran de manera incorrecta en la carga de red
- Se ha corregido un problema en Android que a veces provoca que `DialogServiceConnector` use identificadores de impresión duplicados.
- Se han introducido mejoras en la estabilidad de las conexiones entre interacciones multiproceso y la generación de informes de errores (a través de eventos `Canceled`) cuando se producen con `DialogServiceConnector`.
- Los inicios de sesión de `DialogServiceConnector` ahora proporcionarán eventos correctamente, incluso si se llama a `ListenOnceAsync()` durante una operación `StartKeywordRecognitionAsync()` activa.
- Se ha resuelto un bloqueo asociado a la recepción de actividades `DialogServiceConnector`.

## Muestras

- Inicio rápido para Xamarin
- Se ha actualizado el inicio rápido de CPP con información de ARM64 de Linux
- Se ha actualizado el inicio rápido de Unity con información de iOS

# SDK de Voz 1.6.0: versión de junio de 2019

## Muestras

- Ejemplos de inicio rápido para Texto a voz en UWP y Unity
- Ejemplo de inicio rápido para Swift en iOS
- Ejemplos de Unity para Traducción y Reconocimiento de la intención comunicativa y Voz
- Ejemplos de inicios rápidos actualizados para `DialogServiceConnector`

## Mejoras y cambios

- Espacio de nombres de cuadro de diálogo:
  - El nombre de `SpeechBotConnector` ha cambiado a `DialogServiceConnector`
  - El nombre de `BotConfig` ha cambiado a `DialogServiceConfig`
  - `BotConfig::FromChannelSecret()` se ha reasignado a `DialogServiceConfig::FromBotSecret()`
  - Todos los clientes de Voz de Direct Line existentes siguen siendo compatibles después del cambio de nombre
- Actualización del adaptador REST de TTS para admitir una conexión persistente de proxy
- Un mejor mensaje de error cuando se pasa una región no válida
- Swift/Objective-C:
  - Mejores informes de errores: los métodos que pueden generar un error ahora se encuentran en dos versiones: una que expone un objeto `NSError` para el control de errores y una que genera una excepción. La primera se expone a Swift. Este cambio requiere adaptaciones en el código Swift existente.
  - Mejor control de eventos

## Correcciones de errores

- Corrección de TTS: donde el futuro de `SpeakTextAsync` se devolvió sin esperar al fin de la representación del audio
- Corrección para la serialización de las cadenas en C# para permitir la compatibilidad total con idiomas
- Corrección del problema de las aplicaciones centrales de .NET para cargar la biblioteca principal con un marco

de destino net461 en ejemplos

- Corrección de problemas ocasionales para implementar bibliotecas nativas en la carpeta de salida en los ejemplos
- Corrección para cerrar el socket web de manera confiable
- Corrección de un posible bloqueo al abrir una conexión con una carga muy elevada en Linux
- Corrección de metadatos faltantes en el paquete de marcos para macOS
- Corrección de problemas con `pip install --user` en Windows

## Speech SDK 1.5.1

Se trata de una versión de corrección de errores y solo afecta al SDK nativo o administrado. No afecta a la versión de JavaScript del SDK.

### Correcciones de errores

- Corrección de FromSubscription cuando se usa con la transcripción de la conversación.
- Corrección de errores en la detección de palabras clave para los asistentes por voz.

## Speech SDK 1.5.0 Versión de mayo de 2019

### Nuevas características:

- La detección de palabras clave (KWS) ahora está disponible para Windows y Linux. La funcionalidad KWS podría funcionar con cualquier tipo de micrófono; no obstante, la compatibilidad oficial de KWS está limitada actualmente a las matrices de micrófonos que se encuentran en el hardware de Azure Kinect DK o el SDK de dispositivos de voz.
- La funcionalidad de sugerencia de frases está disponible a través del SDK. Para más información, consulte [esta página](#).
- La funcionalidad de transcripción de conversaciones está disponible a través del SDK. Consulte [aquí](#).
- Compatibilidad agregada con los asistentes por voz mediante el canal Direct Line Speech.

### Muestras

- Se han agregado ejemplos para nuevas características o nuevos servicios admitidos por el SDK.

### Mejoras y cambios

- Se han agregado varias propiedades de reconocimiento para ajustar el comportamiento del servicio o los resultados del servicio (por ejemplo, enmascaramiento de palabras soeces etc.).
- Ahora puede configurar el reconocimiento a través de las propiedades de configuración estándar, incluso si ha creado el valor de `FromEndpoint` del reconocedor.
- Objective-C: se agregó la propiedad `OutputFormat` a `SPXSpeechConfiguration`.
- El SDK ahora admite Debian 9 como una distribución de Linux.

### Correcciones de errores

- Se ha corregido un problema donde el recurso de altavoz se destruía demasiado pronto en la conversión de texto a voz.

## Speech SDK 1.4.2

Se trata de una versión de corrección de errores y solo afecta al SDK nativo o administrado. No afecta a la versión de JavaScript del SDK.

## Speech SDK 1.4.1

Esta es una versión solo para JavaScript. No se agregó ninguna característica. Se realizaron las siguientes correcciones:

- Se impide que el paquete web cargue https-proxy-agent.

## Speech SDK 1.4.0 Versión de abril de 2019

### Nuevas características:

- El SDK admite ahora el servicio de conversión de texto a voz en versión beta. Se admite en Windows y Linux Desktop desde C++ y C#. Para más información, consulte la [información general sobre la conversión de texto a voz](#).
- El SDK ahora admite archivos de audio MP3 y Opus/OGG como archivos de entrada de secuencia. Esta característica solo está disponible en Linux desde C++ y C# y está actualmente en versión beta (más detalles [aquí](#)).
- Speech SDK para Java, .NET Core, C++ y Objective-C ha conseguido compatibilidad con macOS. La compatibilidad de Objective-C con macOS está actualmente en versión beta.
- iOS: Speech SDK para iOS (Objective-C) ahora también se publica como una instancia de CocoaPod.
- JavaScript: compatibilidad con micrófono no predeterminada como dispositivo de entrada.
- JavaScript: compatibilidad con servidores proxy para Node.js.

### Muestras

- se han agregado ejemplos para usar Speech SDK con C++ y con Objective-C en macOS.
- Se han agregado ejemplos que muestran el uso del servicio de conversión de texto a voz.

### Mejoras y cambios

- Python: ahora se exponen propiedades adicionales de los resultados del reconocimiento mediante la propiedad `properties`.
- Para la compatibilidad adicional con el desarrollo y la depuración, puede redirigir la información de registro y diagnóstico del SDK a un archivo de registro (más información [aquí](#)).
- JavaScript: mejora del rendimiento del procesamiento de audio.

### Correcciones de errores

- Mac/iOS: se corrigió un error que daba lugar a una larga espera cuando no se podía establecer una conexión con el servicio de voz.
- Python: mejora del control de errores en los argumentos de las devoluciones de llamada de Python.
- JavaScript: se corrigieron los informes de estado erróneos de la voz que finalizaban en RequestSession.

## Speech SDK 1.3.1 Actualización de febrero de 2019

Se trata de una versión de corrección de errores y solo afecta al SDK nativo o administrado. No afecta a la versión de JavaScript del SDK.

### Corrección de error

- Se ha corregido una fuga de memoria cuando se usa la entrada de micrófono. No afecta a la entrada de archivos o basada en secuencias.

## Speech SDK 1.3.0: versión de febrero de 2019

### Nuevas características

- El SDK de voz admite la selección del micrófono de entrada mediante la clase `AudioConfig`. Esto permite transmitir datos de audio al servicio de voz desde un micrófono no predeterminado. Para más información, consulte la documentación en la que se describe cómo [seleccionar un dispositivo de entrada de audio](#). Esta característica aún no está disponible en JavaScript.
- Speech SDK ahora es compatible con Unity en una versión beta. Proporcione sus comentarios en la sección de problemas en el [repositorio de ejemplos de GitHub](#). Esta versión es compatible con Unity en Windows x86 y x64 (aplicaciones de escritorio o de la Plataforma universal de Windows) y Android (ARM32/64, x86). Puede encontrar más información en nuestra [guía de inicio rápido sobre Unity](#).
- El archivo `Microsoft.CognitiveServices.Speech.csharp.bindings.dll` (incluido en versiones anteriores) ya no es necesario. La funcionalidad está ahora integrada en el SDK principal.

## Muestras

El siguiente contenido nuevo está disponible en nuestro [repositorio de ejemplo](#):

- Ejemplos adicionales para `AudioConfig.FromMicrophoneInput`.
- Ejemplos adicionales de Python para traducción y reconocimiento de intenciones.
- Ejemplos adicionales para usar el objeto `Connection` en iOS.
- Ejemplos adicionales de Java para la traducción con la salida de audio.
- Nuevo ejemplo de uso de la [API de REST de transcripción de lotes](#).

## Mejoras y cambios

- Python
  - Mensajes de error y verificación de parámetros mejorada en `SpeechConfig`.
  - Adición de compatibilidad para el objeto `Connection`.
  - Compatibilidad con Python (x86) de 32 bits en Windows.
  - Speech SDK para Python ya no está disponible como beta.
- iOS
  - El SDK ahora se compila en función de la versión 12.1 del SDK de iOS.
  - El SDK ahora es compatible con las versiones 9.2 y posteriores de iOS.
  - Documentación de referencia mejorada y corrección de varios nombres de propiedad.
- JavaScript
  - Adición de compatibilidad para el objeto `Connection`.
  - Archivos de definición de tipos agregados para JavaScript agrupado.
  - Compatibilidad e implementación iniciales para sugerencias de frases.
  - Colección de propiedades devuelta con JSON del servicio para reconocimiento.
- Los archivos DLL de Windows contienen ahora un recurso de versión.
- Si crea un valor de `FromEndpoint` de reconocedor, puede agregar parámetros directamente a la dirección URL del punto de conexión. Con `FromEndpoint` no puede configurar el reconocedor mediante las propiedades de configuración estándar.

## Correcciones de errores

- La contraseña de proxy y el nombre de usuario de proxy vacíos no se administraron correctamente. Con esta versión, si establece el nombre de usuario de proxy y la contraseña de proxy en una cadena vacía, no se enviarán al conectarse al proxy.
- El identificador de sesión creado por el SDK no siempre es realmente aleatorio para algunos lenguajes o entornos. Se ha agregado la inicialización del generador aleatorio para corregir este problema.
- Control mejorado del token de autorización. Si desea usar un token de autorización, especifíquelo en `SpeechConfig` y deje la clave de suscripción vacía. A continuación, cree el reconocedor como de costumbre.

- En algunos casos, el objeto `Connection` no se publicó correctamente. Ahora se ha corregido.
- Se corrigió el ejemplo de JavaScript para admitir la salida de audio para la síntesis de traducción también en Safari.

## Speech SDK 1.2.1

Esta es una versión solo para JavaScript. No se agregó ninguna característica. Se realizaron las siguientes correcciones:

- Activar el final del flujo en `turn.end`, y no en `speech.end`.
- Corregir error de la bomba de audio por el que no se programaba el siguiente envío en caso de error del envío actual.
- Corregir el reconocimiento continuo con el token de autenticación.
- Corrección de errores de diferentes reconocedores y puntos de conexión.
- Mejoras en la documentación.

## Speech SDK 1.2.0: Versión de diciembre de 2018

### Nuevas características

- Python
  - La versión beta de la compatibilidad con Python (3.5 y versiones posteriores) está disponible con esta versión. Para más información, consulte [aquí](#)(quickstart-python.md).
- JavaScript
  - Speech SDK para JavaScript ha sido de código abierto. El código fuente está disponible en [GitHub](#).
  - Ya se admite Node.js; puede encontrar más información [aquí](#).
  - Se quitó la restricción de longitud para las sesiones de audio; la reconexión se realizará automáticamente en la portada.
- Objeto `Connection`
  - Desde el objeto `Recognizer`, puede acceder al objeto `Connection`. Este objeto le permite iniciar la conexión al servicio y suscribirse para conectar y desconectar eventos explícitamente. (Esta característica no está disponible aún ni en JavaScript ni en Python).
- Compatibilidad con Ubuntu 18.04.
- Android
  - Compatibilidad con ProGuard habilitada durante la generación del APK.

### Mejoras

- Mejoras en el uso de subprocessos internos, lo que reduce el número de subprocessos, bloqueos y exclusiones mutuas.
- Se mejoraron los informes de errores y la información. En algunos casos, los mensajes de error no se propagan totalmente.
- Se actualizaron las dependencias de desarrollo en JavaScript para usar los módulos actualizados.

### Correcciones de errores

- Se han corregido las fugas de causadas por un error de coincidencia de tipos en `RecognizeAsync`.
- En algunos casos, se perdieron excepciones.
- Corrección de las fugas de memoria en los argumentos de eventos de traducción.
- Se ha corregido un problema de bloqueo al volver a conectar en sesiones de larga ejecución.
- Se ha corregido un problema que podría dar lugar a que faltase el resultado final para las traducciones con errores.

- C#: Si no se esperaba una operación `async` en el subprocesso principal, es posible que se pudiese desechar el reconocedor antes de completarse la tarea asíncrona.
- Java: Se ha corregido un problema que provocaba un bloqueo de la VM de Java.
- Objective-C: Se ha corregido la asignación de la enumeración; se devolvió `RecognizedIntent` en lugar de `RecognizingIntent`.
- JavaScript: Se ha establecido el formato de salida predeterminado en "simple" en `SpeechConfig`.
- JavaScript: Se ha quitado una incoherencia entre las propiedades del objeto de configuración en JavaScript y otros lenguajes.

## Muestras

- Se han actualizado y corregido varios ejemplos, como las voces de salida para la traducción, etc.
- Se han agregado ejemplos de Node.js en el [repositorio de ejemplo](#).

# Speech SDK 1.1.0

## Nuevas características

- Compatibilidad con Android x86/x64.
- Compatibilidad con proxy: En el objeto `SpeechConfig`, ahora puede llamar a una función para establecer la información del proxy (nombre de host, puerto, nombre de usuario y contraseña). Esta característica no está disponible aún en iOS.
- Mensajes y códigos de error mejorados. Si un reconocimiento devolvió un error, esto ya ha establecido `Reason` (en el evento cancelado) o `CancellationDetails` (en el resultado del reconocimiento) en `Error`. El evento cancelado ahora contiene dos miembros adicionales, `ErrorCode` y `ErrorDetails`. Si el servidor devolvió información de error adicional con el error notificado, ahora estará disponible en los nuevos miembros.

## Mejoras

- Verificación adicional agregada en la configuración del reconocedor y mensaje de error adicional agregado.
- Control mejorado del silencio prolongado en medio de un archivo de audio.
- Paquete NuGet: para proyectos de .NET Framework, evita la compilación con la configuración de AnyCPU.

## Correcciones de errores

- En los reconocedores se han encontrado varias excepciones corregidas. Además, las excepciones se detectan y se convierten en un evento `Canceled`.
- Corrección de una fuga de memoria en la administración de propiedades.
- Se corrigió el error en el que un archivo de entrada de audio podría bloquear el reconocedor.
- Se corrigió un error donde se podrían recibir eventos después de un evento de detención de la sesión.
- Se corrigieron algunas condiciones de subprocessos.
- Se corrigió un problema de compatibilidad de iOS que podría dar lugar a un bloqueo.
- Mejoras de estabilidad para la compatibilidad del micrófono en Android.
- Se corrigió un error donde un reconocedor en JavaScript ignoraría el lenguaje de reconocimiento.
- Se corrigió un error que impedía establecer el valor `EndpointId` (en algunos casos) en JavaScript.
- Se cambió el orden de los parámetros en `AddIntent` en JavaScript y se agregó la firma de JavaScript `AddIntent` que faltaba.

## Muestras

- Se han agregado ejemplos de C++ y C# para el uso de transmisiones de inserción y extracción en el [repositorio de ejemplos](#).

## Speech SDK 1.0.1

Mejoras en la confiabilidad y correcciones de errores:

- Corrección de un potencial error grave debido a una condición de carrera al desechar un reconocedor.
- Corrección de un potencial error grave en el caso de propiedades sin establecer.
- Comprobación adicional de errores y parámetros.
- Objective-C: corrección de posibles errores graves causados por la invalidación de nombres en NSString.
- Objective-C: ajuste de visibilidad en la API.
- JavaScript: corrección con respecto a los eventos y sus cargas.
- Mejoras en la documentación.

Se ha agregado un nuevo ejemplo de Javascript en nuestro [repositorio de ejemplos](#).

## SDK de Voz 1.0.0 de Cognitive Services: Versión de septiembre de 2018

### Nuevas características:

- Compatibilidad con Objective-C en iOS. Consulte la [Guía de inicio rápido de Objective-C para iOS](#).
- Se admite JavaScript en el explorador. Consulte la [Guía de inicio rápido de JavaScript](#).

### Cambios importantes

- Con esta versión se presentan una serie de cambios importantes. Consulte [esta página](#) para más información.

## SDK de Voz 0.6.0 de Cognitive Services: Versión de agosto de 2018

### Nuevas características:

- Ahora, las aplicaciones de UWP creadas con SDK de Voz superan el Kit para la certificación de aplicaciones en Windows (WACK). Consulte la [Guía de inicio rápido de UWP](#).
- Compatibilidad con .NET Standard 2.0 en Linux (Ubuntu 16.04 x64).
- Experimental: compatibilidad con Java 8 en Windows (64 bits) y Linux (Ubuntu 16.04 x 64). Consulte la [Guía de inicio rápido de Java Runtime Environment](#).

### Cambios funcionales

- Se expone más información detallada sobre los errores de conexión.

### Cambios importantes

- En Java (Android), la función `SpeechFactory.configureNativePlatformBindingWithDefaultCertificate` ya no requiere un parámetro de ruta de acceso. Ahora, la ruta de acceso se detecta automáticamente en todas las plataformas compatibles.
- En Java y C#, se ha quitado el descriptor de acceso get- de la propiedad `EndpointUrl`.

### Correcciones de errores

- En Java, se implementa ahora el resultado de la síntesis de audio en el reconocedor de traducción.
- Se ha corregido un error que podía provocar subprocessos inactivos y un mayor número de sockets abiertos y sin usar.
- Se ha corregido un problema por el que un proceso de reconocimiento de larga ejecución podía terminar en mitad de la transmisión.
- Se ha corregido una condición de carrera en el proceso de apagado del reconocedor.

# SDK de Voz 0.5.0 de Cognitive Services: Versión de julio de 2018

## Nuevas características:

- Compatibilidad con la plataforma Android (API 23: Android Marshmallow 6.0 o posterior). Consulte el [inicio rápido de Android](#).
- Compatibilidad con .NET Standard 2.0 en Windows. Consulte el [inicio rápido de .NET Core](#).
- Experimental: compatibilidad con UWP en Windows (versión 1709 o posterior).
  - Consulte la [Guía de inicio rápido de UWP](#).
  - Nota: Las aplicaciones de UWP creadas con el SDK de Voz no pasan aún el Kit para la certificación de aplicaciones en Windows (WACK).
- Compatibilidad con el reconocimiento de ejecución prolongada con reconexión automática.

## Cambios funcionales

- `StartContinuousRecognitionAsync()` admite reconocimiento de ejecución prolongada.
- El resultado del reconocimiento contiene más campos. Tienen un desplazamiento desde el principio del audio y la duración (ambos en tics) del texto reconocido y valores adicionales que representan el estado de reconocimiento, por ejemplo, `InitialSilenceTimeout` e `InitialBabbleTimeout`.
- Compatibilidad con `AuthorizationToken` para la creación de instancias de fábrica.

## Cambios importantes

- Eventos de reconocimiento: el tipo de evento `NoMatch` se combina con el evento `Error`.
- `SpeechOutputFormat` en C# se llama ahora `OutputFormat` para concordar con C++.
- El tipo de valor devuelto de algunos métodos de la interfaz `AudioInputStream` se ha modificado ligeramente:
  - En Java, el método `read` ahora devuelve `long` en lugar de `int`.
  - En C#, el método `Read` ahora devuelve `uint` en lugar de `int`.
  - En C++, los métodos `Read` y `GetFormat` ahora devuelven `size_t` en lugar de `int`.
- C++: las instancias de secuencias de entrada de audio ahora solo se pueden pasar como un valor `shared_ptr`.

## Correcciones de errores

- Se han corregido los valores devueltos incorrectos cuando se agota el tiempo de espera de `RecognizeAsync()`.
- Se ha eliminado la dependencia de las bibliotecas de Media Foundation en Windows. El SDK ahora usa las API de audio básicas.
- Corrección de la documentación: se ha agregado una página de [regiones](#) para describir cuáles son las regiones admitidas.

## Problema conocido

- SDK de Voz para Android no informa de los resultados de la síntesis de voz para la traducción. Este problema se solucionará en la próxima versión.

# SDK de Voz 0.4.0 de Cognitive Services: Versión de junio de 2018

## Cambios funcionales

- `AudioInputStream`

Un reconocedor ahora puede consumir una secuencia como origen de audio. Para más información, consulte la [guía de procedimientos](#) relacionada.

- Formato de salida detallado

Al crear un elemento `SpeechRecognizer`, puede solicitar el formato de salida `Detailed` o `Simple`.

`DetailedSpeechRecognitionResult` contiene una puntuación de confianza, texto reconocido, formato léxico sin formato, formato normalizado y formato normalizado con palabras soeces enmascaradas.

## Cambio importante

- En C# se cambia de `SpeechRecognitionResult.RecognizedText` a `SpeechRecognitionResult.Text`.

## Correcciones de errores

- Se ha corregido un posible problema de devolución de llamada en la capa USP durante el apagado.
- Si un reconocedor usaba un archivo de entrada de audio, mantenía el identificador de archivo más tiempo del necesario.
- Se han eliminado varios interbloqueos entre el suministro de mensajes y el reconocedor.
- Se desencadena un resultado `NoMatch` cuando se agota la respuesta del servicio.
- Las bibliotecas de Media Foundation en Windows son de carga retrasada. Esta biblioteca solo es necesaria para la entrada del micrófono.
- La velocidad de carga de los datos de audio se limita al doble de la velocidad de audio original.
- En Windows, los ensamblados .NET de C# ahora son de nombre seguro.
- Corrección de la documentación: `Region` necesita información para crear un reconocedor.

Se han agregado más ejemplos y se actualizan constantemente. Para obtener el conjunto más reciente de ejemplos, consulte el [repositorio de GitHub de ejemplos de SDK de Voz](#).

## SDK de Voz 0.2.12733 de Cognitive Services: Versión de mayo de 2018

Esta versión es la primera versión preliminar pública de SDK de Voz de Cognitive Services.

# Text-to-speech REST API

13/01/2020 • 18 minutes to read • [Edit Online](#)

El servicio de voz le permite [convertir texto en voz sintetizada y obtener una lista de voces admitidas para una región con un conjunto de API REST](#). Cada punto de conexión disponible se asocia con una región. Se requiere una clave de suscripción para el punto de conexión o región que se va a usar.

Text to Speech REST API admite voces neuronales y de texto a voz estándar, y cada una de ellas admite un idioma y un dialecto específicos, que se identifican mediante la configuración regional.

- Para ver una lista completa de voces, consulte [compatibilidad con idiomas](#).
- Para obtener información acerca de la disponibilidad regional, consulte las [regiones](#).

## IMPORTANT

Los costes varían para voces estándar, personalizadas y neuronales. Para obtener más información, consulte el apartado [Precios](#).

Antes de utilizar esta API, comprenda:

- La API REST de texto a voz requiere un encabezado de autorización. Esto significa que tiene que completar un intercambio de tokens para acceder al servicio. Para más información, consulte [Autenticación](#).

## Authentication

Cada solicitud requiere un encabezado de autorización. Esta tabla muestra qué encabezados son compatibles con cada servicio:

ENCABEZADOS DE AUTORIZACIÓN COMPATIBLES	VOZ A TEXTO	TEXTO A VOZ
Ocp-Apim-Subscription-Key	Sí	Sin
Autorización: Portador	Sí	Sí

Cuando se usa el encabezado `Ocp-Apim-Subscription-Key`, solo se le pide que proporcione la clave de suscripción. Por ejemplo:

```
'Ocp-Apim-Subscription-Key': 'YOUR_SUBSCRIPTION_KEY'
```

Cuando se usa el encabezado `Authorization: Bearer`, se le pide que haga una solicitud al punto de conexión `issueToken`. En esta solicitud, va a intercambiar la clave de suscripción de un token de acceso que es válido durante 10 minutos. En las secciones siguientes obtendrá información sobre cómo obtener un token y usar un token.

### Obtención de un token de acceso

Para obtener un token de acceso, tiene que realizar una solicitud al punto de conexión `issueToken` mediante `Ocp-Apim-Subscription-Key` y su clave de suscripción.

Se admiten estas regiones y puntos de conexión:

REGION	PUNTO DE CONEXIÓN DE SERVICIO DE TOKEN
Este de Australia	<code>https://australiaeast.api.cognitive.microsoft.com/sts/v1.0/issueToken</code>
Centro de Canadá	<code>https://canadacentral.api.cognitive.microsoft.com/sts/v1.0/issueToken</code>
Centro de EE. UU.	<code>https://centralus.api.cognitive.microsoft.com/sts/v1.0/issueToken</code>
Asia oriental	<code>https://eastasia.api.cognitive.microsoft.com/sts/v1.0/issueToken</code>

REGION	PUNTO DE CONEXIÓN DE SERVICIO DE TOKEN
Este de EE. UU	<a href="https://eastus.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://eastus.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
Este de EE. UU. 2	<a href="https://eastus2.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://eastus2.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
Centro de Francia	<a href="https://francecentral.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://francecentral.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
India central	<a href="https://centralindia.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://centralindia.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
Este de Japón	<a href="https://japaneast.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://japaneast.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
Corea Central	<a href="https://koreacentral.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://koreacentral.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
Centro-Norte de EE. UU	<a href="https://northcentralus.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://northcentralus.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
Europa del Norte	<a href="https://northeurope.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://northeurope.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
Centro-Sur de EE. UU	<a href="https://southcentralus.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://southcentralus.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
Sudeste asiático	<a href="https://southeastasia.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://southeastasia.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
Sur del Reino Unido 2	<a href="https://uksouth.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://uksouth.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
Europa occidental	<a href="https://westeurope.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://westeurope.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
Oeste de EE. UU.	<a href="https://westus.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://westus.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
Oeste de EE. UU. 2	<a href="https://westus2.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://westus2.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>

Use estos ejemplos para crear la solicitud de token de acceso.

#### Ejemplo de HTTP

Este ejemplo es una solicitud HTTP para obtener un token. Reemplace `YOUR_SUBSCRIPTION_KEY` por la clave de suscripción del servicio Voz. Si la suscripción no está en la región Oeste de EE. UU., reemplace el encabezado `Host` por el nombre de host de la región.

```
POST /sts/v1.0/issueToken HTTP/1.1
Ocp-Apim-Subscription-Key: YOUR_SUBSCRIPTION_KEY
Host: westus.api.cognitive.microsoft.com
Content-type: application/x-www-form-urlencoded
Content-Length: 0
```

El cuerpo de la respuesta contiene el token de acceso en formato JSON Web Token (JWT).

#### Ejemplo de PowerShell

En este ejemplo se muestra un script sencillo de PowerShell para obtener un token de acceso. Reemplace `YOUR_SUBSCRIPTION_KEY` por la clave de suscripción del servicio Voz. Asegúrese de usar el punto de conexión correcto para la región que coincide con su suscripción. En este ejemplo la región es Oeste de EE. UU.

```

$FetchTokenHeader = @{
    'Content-type'='application/x-www-form-urlencoded';
    'Content-Length'='0';
    'Ocp-Apim-Subscription-Key' = 'YOUR_SUBSCRIPTION_KEY'
}

$OAuthToken = Invoke-RestMethod -Method POST -Uri https://westus.api.cognitive.microsoft.com/sts/v1.0/issueToken
-Headers $FetchTokenHeader

# show the token received
$OAuthToken

```

#### Ejemplo de cURL

cURL es una herramienta de la línea de comandos disponible en Linux (y en el subsistema Windows para Linux). Este comando cURL muestra cómo obtener un token de acceso. Reemplace `YOUR_SUBSCRIPTION_KEY` por la clave de suscripción del servicio Voz. Asegúrese de usar el punto de conexión correcto para la región que coincide con su suscripción. En este ejemplo la región es Oeste de EE. UU.

```

curl -v -X POST
"https://westus.api.cognitive.microsoft.com/sts/v1.0/issueToken" \
-H "Content-type: application/x-www-form-urlencoded" \
-H "Content-Length: 0" \
-H "Ocp-Apim-Subscription-Key: YOUR_SUBSCRIPTION_KEY"

```

#### Ejemplo de C#

Esta clase de C# muestra cómo obtener un token de acceso. Pase la clave de suscripción del servicio Voz al crear una instancia de la clase. Si su suscripción no está en la región Oeste de EE. UU., cambie el valor de `FetchTokenUri` para que coincida con la región de su suscripción.

```

public class Authentication
{
    public static readonly string FetchTokenUri =
        "https://westus.api.cognitive.microsoft.com/sts/v1.0/issueToken";
    private string subscriptionKey;
    private string token;

    public Authentication(string subscriptionKey)
    {
        this.subscriptionKey = subscriptionKey;
        this.token = FetchTokenAsync(FetchTokenUri, subscriptionKey).Result;
    }

    public string GetAccessToken()
    {
        return this.token;
    }

    private async Task<string> FetchTokenAsync(string fetchUri, string subscriptionKey)
    {
        using (var client = new HttpClient())
        {
            client.DefaultRequestHeaders.Add("Ocp-Apim-Subscription-Key", subscriptionKey);
            UriBuilder uriBuilder = new UriBuilder(fetchUri);

            var result = await client.PostAsync(uriBuilder.Uri.AbsoluteUri, null);
            Console.WriteLine("Token Uri: {0}", uriBuilder.Uri.AbsoluteUri);
            return await result.Content.ReadAsStringAsync();
        }
    }
}

```

#### Ejemplo de Python

```

# Request module must be installed.
# Run pip install requests if necessary.
import requests

subscription_key = 'REPLACE_WITH_YOUR_KEY'

def get_token(subscription_key):
    fetch_token_url = 'https://westus.api.cognitive.microsoft.com/sts/v1.0/issueToken'
    headers = {
        'Ocp-Apim-Subscription-Key': subscription_key
    }
    response = requests.post(fetch_token_url, headers=headers)
    access_token = str(response.text)
    print(access_token)

```

## Uso de un token de acceso

Se debe enviar el token de acceso al servicio como encabezado `Authorization: Bearer <TOKEN>`. Cada token de acceso tiene una validez de 10 minutos. Puede obtener un nuevo token en cualquier momento. No obstante, para reducir el tráfico de red y la latencia, se recomienda usar el mismo token durante nueve minutos.

Este es un ejemplo de solicitud HTTP a la API REST de texto a voz:

```

POST /cognitiveservices/v1 HTTP/1.1
Authorization: Bearer YOUR_ACCESS_TOKEN
Host: westus.stt.speech.microsoft.com
Content-type: application/ssml+xml
Content-Length: 199
Connection: Keep-Alive

// Message body here...

```

## Obtener una lista de voces

El punto de conexión `voices/list` le permite obtener una lista completa de las voces de una región o punto de conexión en concreto.

### Regiones y puntos de conexión

REGION	PUNTO DE CONEXIÓN
Este de Australia	<a href="https://australiaeast.tts.speech.microsoft.com/cognitiveservices/voices/list">https://australiaeast.tts.speech.microsoft.com/cognitiveservices/voices/list</a>
Sur de Brasil	<a href="https://brazilsouth.tts.speech.microsoft.com/cognitiveservices/voices/list">https://brazilsouth.tts.speech.microsoft.com/cognitiveservices/voices/list</a>
Centro de Canadá	<a href="https://canadacentral.tts.speech.microsoft.com/cognitiveservices/voices/list">https://canadacentral.tts.speech.microsoft.com/cognitiveservices/voices/list</a>
Centro de EE. UU.	<a href="https://centralus.tts.speech.microsoft.com/cognitiveservices/voices/list">https://centralus.tts.speech.microsoft.com/cognitiveservices/voices/list</a>
Asia oriental	<a href="https://eastasia.tts.speech.microsoft.com/cognitiveservices/voices/list">https://eastasia.tts.speech.microsoft.com/cognitiveservices/voices/list</a>
East US	<a href="https://eastus.tts.speech.microsoft.com/cognitiveservices/voices/list">https://eastus.tts.speech.microsoft.com/cognitiveservices/voices/list</a>
Este de EE. UU. 2	<a href="https://eastus2.tts.speech.microsoft.com/cognitiveservices/voices/list">https://eastus2.tts.speech.microsoft.com/cognitiveservices/voices/list</a>
Centro de Francia	<a href="https://francecentral.tts.speech.microsoft.com/cognitiveservices/voices/list">https://francecentral.tts.speech.microsoft.com/cognitiveservices/voices/list</a>
India central	<a href="https://centralindia.tts.speech.microsoft.com/cognitiveservices/voices/list">https://centralindia.tts.speech.microsoft.com/cognitiveservices/voices/list</a>
Este de Japón	<a href="https://japaneast.tts.speech.microsoft.com/cognitiveservices/voices/list">https://japaneast.tts.speech.microsoft.com/cognitiveservices/voices/list</a>
Corea Central	<a href="https://koreacentral.tts.speech.microsoft.com/cognitiveservices/voices/list">https://koreacentral.tts.speech.microsoft.com/cognitiveservices/voices/list</a>

REGION	PUNTO DE CONEXIÓN
Centro-Norte de EE. UU.	<a href="https://northcentralus.tts.speech.microsoft.com/cognitiveservices/voices/list">https://northcentralus.tts.speech.microsoft.com/cognitiveservices/voices/list</a>
Europa del Norte	<a href="https://northeurope.tts.speech.microsoft.com/cognitiveservices/voices/list">https://northeurope.tts.speech.microsoft.com/cognitiveservices/voices/list</a>
Centro-Sur de EE. UU.	<a href="https://southcentralus.tts.speech.microsoft.com/cognitiveservices/voices/list">https://southcentralus.tts.speech.microsoft.com/cognitiveservices/voices/list</a>
Sudeste asiático	<a href="https://southeastasia.tts.speech.microsoft.com/cognitiveservices/voices/list">https://southeastasia.tts.speech.microsoft.com/cognitiveservices/voices/list</a>
Sur del Reino Unido 2	<a href="https://uksouth.tts.speech.microsoft.com/cognitiveservices/voices/list">https://uksouth.tts.speech.microsoft.com/cognitiveservices/voices/list</a>
Europa occidental	<a href="https://westeurope.tts.speech.microsoft.com/cognitiveservices/voices/list">https://westeurope.tts.speech.microsoft.com/cognitiveservices/voices/list</a>
Oeste de EE. UU.	<a href="https://westus.tts.speech.microsoft.com/cognitiveservices/voices/list">https://westus.tts.speech.microsoft.com/cognitiveservices/voices/list</a>
Oeste de EE. UU. 2	<a href="https://westus2.tts.speech.microsoft.com/cognitiveservices/voices/list">https://westus2.tts.speech.microsoft.com/cognitiveservices/voices/list</a>

### Encabezados de solicitud

En esta tabla se enumeran los encabezados obligatorios y opcionales para las solicitudes de texto a voz.

ENCABEZADO	DESCRIPCIÓN	OBLIGATORIO U OPCIONAL
<code>Authorization</code>	Un token de autorización precedido por la palabra <code>Bearer</code> . Para más información, consulte <a href="#">Autenticación</a> .	Obligatorio

### Cuerpo de la solicitud

No es necesario un cuerpo para las solicitudes `GET` a este punto de conexión.

### Solicitud de ejemplo

Esta solicitud solo requiere un encabezado de autorización.

```
GET /cognitiveservices/voices/list HTTP/1.1
Host: westus.tts.speech.microsoft.com
Authorization: Bearer [Base64 access_token]
```

### Respuesta de muestra

Esta respuesta se ha truncado para ilustrar la estructura de una respuesta.

#### NOTE

La disponibilidad de voces varía según la región o el punto de conexión.

```
[
  {
    "Name": "Microsoft Server Speech Text to Speech Voice (ar-EG, Hoda)",
    "ShortName": "ar-EG-Hoda",
    "Gender": "Female",
    "Locale": "ar-EG"
  },
  {
    "Name": "Microsoft Server Speech Text to Speech Voice (ar-SA, Naayf)",
    "ShortName": "ar-SA-Naayf",
    "Gender": "Male",
    "Locale": "ar-SA"
  },
  {
    "Name": "Microsoft Server Speech Text to Speech Voice (bg-BG, Ivan)",
    "ShortName": "bg-BG-Ivan",
    "Gender": "Male",
    "Locale": "bg-BG"
  },
  {
    "Name": "Microsoft Server Speech Text to Speech Voice (ca-ES, HerenaRUS)",
    "ShortName": "ca-ES-HerenaRUS",
    "Gender": "Female",
    "Locale": "ca-ES"
  },
  {
    "Name": "Microsoft Server Speech Text to Speech Voice (cs-CZ, Jakub)",
    "ShortName": "cs-CZ-Jakub",
    "Gender": "Male",
    "Locale": "cs-CZ"
  },
  ...
]
]
```

## Códigos de estado HTTP

El estado HTTP de cada respuesta indica estados de corrección o error comunes.

CÓDIGO DE ESTADO HTTP	DESCRIPCIÓN	POSSIBLE MOTIVO
200	OK	La solicitud fue correcta.
400	Bad Request	Falta un parámetro requerido, está vacío o es nulo. O bien, el valor pasado a un parámetro obligatorio u opcional no es válido. Un problema común es que el encabezado sea demasiado largo.
401	No autorizado	La solicitud no está autenticada. Asegúrese de que la clave de suscripción o el token sean válidos y de la región correcta.
429	Demasiadas solicitudes	Ha superado la cuota o la tasa de solicitudes permitidas para su suscripción.
502	Puerta de enlace incorrecta	Problema de red o de servidor. Podría indicar también encabezados no válidos.

## Conversión de texto a voz

El punto de conexión `v1` le permite convertir texto a voz usando [lenguaje de marcado de síntesis de voz \(SSML\)](#).

### Regiones y puntos de conexión

Estas regiones son compatibles con la conversión de texto a voz mediante la API REST. Asegúrese de que selecciona el punto de conexión que coincide con la región de su suscripción.

### Voces estándares y neuronales

Utilice esta tabla para determinar la disponibilidad de las voces estándar y neuronales por región o punto de conexión:

REGION	PUNTO DE CONEXIÓN	VOCES ESTÁNDAR	VOCES NEURONALES
Este de Australia	<a href="https://australiaeast.tts.speech.microsoft.com/cognitiveservices/v1">https://australiaeast.tts.speech.microsoft.com/cognitiveservices/v1</a>	Sí	
Centro de Canadá	<a href="https://canadacentral.tts.speech.microsoft.com/cognitiveservices/v1">https://canadacentral.tts.speech.microsoft.com/cognitiveservices/v1</a>	Sí	
Centro de EE. UU.	<a href="https://centralus.tts.speech.microsoft.com/cognitiveservices/v1">https://centralus.tts.speech.microsoft.com/cognitiveservices/v1</a>	Sí	
Asia oriental	<a href="https://eastasia.tts.speech.microsoft.com/cognitiveservices/v1">https://eastasia.tts.speech.microsoft.com/cognitiveservices/v1</a>	Sí	
Este de EE. UU.	<a href="https://eastus.tts.speech.microsoft.com/cognitiveservices/v1">https://eastus.tts.speech.microsoft.com/cognitiveservices/v1</a>	Sí	
Este de EE. UU. 2	<a href="https://eastus2.tts.speech.microsoft.com/cognitiveservices/v1">https://eastus2.tts.speech.microsoft.com/cognitiveservices/v1</a>	Sí	
Centro de Francia	<a href="https://francecentral.tts.speech.microsoft.com/cognitiveservices/v1">https://francecentral.tts.speech.microsoft.com/cognitiveservices/v1</a>	Sí	
India central	<a href="https://centralindia.tts.speech.microsoft.com/cognitiveservices/v1">https://centralindia.tts.speech.microsoft.com/cognitiveservices/v1</a>	Sí	
Este de Japón	<a href="https://japaneast.tts.speech.microsoft.com/cognitiveservices/v1">https://japaneast.tts.speech.microsoft.com/cognitiveservices/v1</a>	Sí	
Corea Central	<a href="https://koreacentral.tts.speech.microsoft.com/cognitiveservices/v1">https://koreacentral.tts.speech.microsoft.com/cognitiveservices/v1</a>	Sí	
Centro-Norte de EE. UU.	<a href="https://northcentralus.tts.speech.microsoft.com/cognitiveservices/v1">https://northcentralus.tts.speech.microsoft.com/cognitiveservices/v1</a>	Sí	
Europa del Norte	<a href="https://northeurope.tts.speech.microsoft.com/cognitiveservices/v1">https://northeurope.tts.speech.microsoft.com/cognitiveservices/v1</a>	Sí	
Centro-Sur de EE. UU.	<a href="https://southcentralus.tts.speech.microsoft.com/cognitiveservices/v1">https://southcentralus.tts.speech.microsoft.com/cognitiveservices/v1</a>	Sí	
Sudeste asiático	<a href="https://southeastasia.tts.speech.microsoft.com/cognitiveservices/v1">https://southeastasia.tts.speech.microsoft.com/cognitiveservices/v1</a>	Sí	
Sur de Reino Unido 2	<a href="https://uksouth.tts.speech.microsoft.com/cognitiveservices/v1">https://uksouth.tts.speech.microsoft.com/cognitiveservices/v1</a>	Sí	
Europa occidental	<a href="https://westeurope.tts.speech.microsoft.com/cognitiveservices/v1">https://westeurope.tts.speech.microsoft.com/cognitiveservices/v1</a>	Sí	
Oeste de EE. UU.	<a href="https://westus.tts.speech.microsoft.com/cognitiveservices/v1">https://westus.tts.speech.microsoft.com/cognitiveservices/v1</a>	Sí	
Oeste de EE. UU. 2	<a href="https://westus2.tts.speech.microsoft.com/cognitiveservices/v1">https://westus2.tts.speech.microsoft.com/cognitiveservices/v1</a>	Sí	

### Voces personalizadas

Si ha creado una fuente de voz personalizada, use el punto de conexión que ha creado. También puede utilizar los puntos de conexión que se indican a continuación; para ello, reemplace `{deploymentId}` por el identificador de implementación para el modelo de voz.

REGION	PUNTO DE CONEXIÓN
Este de Australia	<a href="https://australiaeast.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}">https://australiaeast.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}</a>
Centro de Canadá	<a href="https://canadacentral.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}">https://canadacentral.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}</a>
Centro de EE. UU.	<a href="https://centralus.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}">https://centralus.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}</a>
Asia oriental	<a href="https://eastasia.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}">https://eastasia.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}</a>

REGION	PUNTO DE CONEXIÓN
Este de EE. UU	<code>https://eastus.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}</code>
Este de EE. UU. 2	<code>https://eastus2.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}</code>
Centro de Francia	<code>https://francecentral.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}</code>
India central	<code>https://centralindia.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}</code>
Este de Japón	<code>https://japaneast.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}</code>
Corea Central	<code>https://koreacentral.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}</code>
Centro-Norte de EE. UU	<code>https://northcentralus.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}</code>
Europa del Norte	<code>https://northeurope.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}</code>
Centro-Sur de EE. UU	<code>https://southcentralus.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}</code>
Sudeste asiático	<code>https://southeastasia.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}</code>
Sur del Reino Unido 2	<code>https://uksouth.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}</code>
Europa occidental	<code>https://westeurope.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}</code>
Oeste de EE. UU.	<code>https://westus.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}</code>
Oeste de EE. UU. 2	<code>https://westus2.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}</code>

## Encabezados de solicitud

En esta tabla se enumeran los encabezados obligatorios yopcionales para las solicitudes de texto a voz.

ENCABEZADO	DESCRIPCIÓN	OBLIGATORIO U OPCIONAL
<code>Authorization</code>	Un token de autorización precedido por la palabra <code>Bearer</code> . Para más información, consulte <a href="#">Autenticación</a> .	Obligatorio
<code>Content-Type</code>	Especifica el tipo de contenido para el texto proporcionado. Valor aceptable: <code>application/ssml+xml</code> .	Obligatorio
<code>X-Microsoft-OutputFormat</code>	Especifica el formato de salida del audio. Para obtener una lista completa de los valores aceptados, consulte <a href="#">salidas de audio</a> .	Obligatorio
<code>User-Agent</code>	Nombre de la aplicación. El valor proporcionado debe tener menos de 255 caracteres.	Obligatorio

## Salidas de audio

Esta es una lista de formatos de audio admitidos que se envían en cada solicitud como encabezado `X-Microsoft-OutputFormat`. Cada uno de ellos incorpora una velocidad de bits y el tipo de codificación. El servicio de voz admite salidas de audio de 24 kHz, 16 kHz y 8 kHz.

raw-16khz-16bit-mono-pcm	raw-8khz-8bit-mono-mulaw
riff-8khz-8bit-mono-alaw	riff-8khz-8bit-mono-mulaw
riff-16khz-16bit-mono-pcm	audio-16khz-128kbitrate-mono-mp3
audio-16khz-64kbitrate-mono-mp3	audio-16khz-32kbitrate-mono-mp3
raw-24khz-16bit-mono-pcm	riff-24khz-16bit-mono-pcm
audio-24khz-160kbitrate-mono-mp3	audio-24khz-96kbitrate-mono-mp3
audio-24khz-48kbitrate-mono-mp3	

### NOTE

Si la voz y el formato de salida seleccionados tienen velocidades de bits diferentes, se vuelve a muestrear el audio según sea necesario. Sin embargo, las voces de 24 kHz no admiten los formatos de salida `audio-16khz-16kbps-mono-siren` y `riff-16khz-16kbps-mono-siren`.

## Cuerpo de la solicitud

El cuerpo de cada solicitud `POST` se envía como [lenguaje de marcado de síntesis de voz \(SSML\)](#). SSML le permite elegir la voz y el idioma de la voz sintetizada que devuelve el servicio de texto a voz. Para ver una lista completa de voces compatibles, consulte [compatibilidad con idiomas](#).

### NOTE

Si usa una voz personalizada, el cuerpo de una solicitud puede enviarse como texto sin formato (ASCII o UTF-8).

## Solicitud de ejemplo

Esta solicitud HTTP utiliza SSML para especificar el idioma y la voz. El cuerpo no puede superar los 1000 caracteres.

```
POST /cognitiveservices/v1 HTTP/1.1

X-Microsoft-OutputFormat: raw-16khz-16bit-mono-pcm
Content-Type: application/ssml+xml
Host: westus.tts.speech.microsoft.com
Content-Length: 225
Authorization: Bearer [Base64 access_token]

<speak version='1.0' xml:lang='en-US'><voice xml:lang='en-US' xml:gender='Female'
name='en-US-JessaRUS'>
    Microsoft Speech Service Text-to-Speech API
</voice></speak>
```

Consulte nuestras guías de inicio rápido para ver ejemplos específicos del idioma:

- [.NET Core, C#](#)
- [Python](#)
- [Node.js](#)

## Códigos de estado HTTP

El estado HTTP de cada respuesta indica estados de corrección o error comunes.

CÓDIGO DE ESTADO HTTP	DESCRIPCIÓN	POSIBLE MOTIVO
200	OK	La solicitud es correcta; el cuerpo de la respuesta es un archivo de audio.
400	Bad Request	Falta un parámetro requerido, está vacío o es nulo. O bien, el valor pasado a un parámetro obligatorio u opcional no es válido. Un problema común es que el encabezado sea demasiado largo.
401	No autorizado	La solicitud no está autenticada. Asegúrese de que la clave de suscripción o el token sean válidos y de la región correcta.
413	Entidad de solicitud demasiado larga	La entrada de SSML tiene más de 1024 caracteres.
415	Tipo de medio no compatible	Es posible que se haya proporcionado el <code>Content-Type</code> incorrecto. <code>Content-Type</code> se debe establecer en <code>application/ssml+xml</code> .
429	Demasiadas solicitudes	Ha superado la cuota o la tasa de solicitudes permitidas para su suscripción.
502	Puerta de enlace incorrecta	Problema de red o de servidor. Podría indicar también encabezados no válidos.

Si el estado HTTP es `200 OK`, el cuerpo de la respuesta contiene un archivo de audio en el formato solicitado. Este archivo se puede reproducir mientras se transfiere o guardarse en un búfer o en un archivo.

## Pasos siguientes

- [Obtenga su suscripción de prueba a Voz](#)
- [Personalización de modelos acústicos](#)
- [Personalización de modelos de lenguaje](#)

# Preguntas más frecuentes sobre Text to Speech

13/01/2020 • 7 minutes to read • [Edit Online](#)

Si no encuentra respuestas a sus preguntas en estas P+F, consulte otras opciones de soporte técnico.

## General

### **P: ¿Cuál es la diferencia entre un modelo de voz estándar y un modelo de voz personalizado?**

**R.** : El modelo de voz estándar (también denominado *fuente de voz*) se ha entrenado con datos de Microsoft y ya está implementado en la nube. Puede usar un modelo de voz personalizado para adaptar un modelo promedio y transferir la expresión y el timbre de acuerdo con el estilo de voz del orador, o bien entrenar un nuevo modelo completo en función de los datos de entrenamiento que preparó el usuario. Actualmente son cada vez más los clientes que quieren una voz única y distintiva para sus bots. La plataforma de compilación de voz personalizada es la opción correcta para eso.

### **P: ¿Dónde empiezo si quiero usar un modelo de voz estándar?**

**R.** : Hay más de 80 modelos de voz estándar en más de 45 idiomas disponibles mediante solicitudes HTTP. Primero, obtenga una [clave suscripción](#). Para realizar llamadas REST a los modelos de voz ya implementados, consulte la [API de REST](#).

### **P: Si quiero usar un modelo de voz personalizado, ¿es la API la misma que la usada para las voces estándar?**

**R.** : Cuando se crea e implementa un modelo de voz personalizado, obtiene un punto de conexión único para el modelo. Para usar la voz para hablar en las aplicaciones, deberá especificar el punto de conexión en las solicitudes HTTP. La misma funcionalidad que está disponible a través de la API de REST para el servicio Text to Speech está disponible para el punto de conexión personalizado. Aprenda a [crear y usar el punto de conexión personalizado](#).

### **P: ¿Es necesario preparar los datos de entrenamiento para crear modelos de voz personalizados por mi cuenta?**

**R.** : Sí, debe preparar los datos de entrenamiento por sí mismo para un modelo de voz personalizado.

Se requiere una colección de datos de voz para crear un modelo de voz personalizado. Esta colección se compone de un conjunto de archivos de audio de las grabaciones de voz y un archivo de texto de la transcripción de cada uno de los archivos de audio. El resultado de su voz digital se basa principalmente en la calidad de los datos de entrenamiento. Para generar una buena voz de texto a voz, es importante que las grabaciones se realicen en una sala silenciosa con un micrófono de pedestal de alta calidad. El volumen constante, la velocidad de la conversación, el tono al hablar e incluso la coherencia en las particularidades expresivas del habla son esenciales para compilar una gran voz digital. Se recomienda encarecidamente grabar las voces en un estudio de grabación.

Actualmente, no proporcionamos compatibilidad con la grabación en línea ni tenemos ninguna recomendación sobre estudios de grabación. Si quiere conocer el requisito de formato, consulte [cómo preparar las grabaciones y transcripciones](#).

### **P: ¿Qué guiones debo usar para grabar los datos de voz para el entrenamiento de voz personalizado?**

**R.** : No hay límite en cuanto a los guiones que se pueden usar para las grabaciones de voz. Puede usar sus propios guiones para grabar el discurso. Solo debe asegurarse de tener la cobertura fonética suficiente en los datos de voz. Para entrenar una voz personalizada, puede empezar con un volumen pequeño de datos de voz, que podrían ser 50 frases distintas (unos 3 a 5 minutos de habla). Cuantos más datos proporcione, más natural será su voz. Para

empezar a entrenar una fuente de voz completa, puede proporcionar grabaciones de más de 2000 frases (unas 3 a 4 horas de habla). Para obtener una voz completa de alta calidad, deberá preparar grabaciones de más de 6000 frases (unas 8 a 10 horas de habla).

Proporcionamos servicios adicionales para ayudarlo a preparar guiones para la grabación. Póngase en contacto con la [asistencia al cliente de Voz personalizada](#) en caso de consultas.

**P: ¿Qué pasa si necesito más simultaneidad que el valor predeterminado o la que se ofrece en el portal?**

**R.** : Se puede escalar verticalmente el modelo en incrementos de 20 solicitudes simultáneas. Póngase en contacto con la [asistencia al cliente de Voz personalizada](#) en caso de consultas sobre un mayor escalamiento.

**P: ¿Puedo descargar mi modelo y ejecutarlo localmente?**

**R.** : Los modelos no se pueden descargar ni ejecutar localmente.

**P: ¿Están limitadas mis solicitudes?**

**R.** : La API REST limita las solicitudes a 25 cada 5 segundos. Encontrará los detalles en nuestras páginas para [Text to Speech](#).

## Pasos siguientes

- [Solución de problemas](#)
- [Notas de la versión](#)

# Inicio rápido: Reconocimiento de voz, intenciones y entidades con Language Understanding (LUIS)

16/01/2020 • 44 minutes to read • [Edit Online](#)

En este inicio rápido, usará el [SDK de Voz](#) y el servicio Language Understanding (LUIS) para reconocer los intentos de los datos de audio capturados de un micrófono. En concreto, usará el SDK de Voz para capturar la voz y un dominio predefinido de LUIS para identificar las intenciones de automatización doméstica, como encender y apagar una luz.

Tras cumplir algunos requisitos previos, no es preciso seguir muchos pasos para el reconocimiento de voz y la identificación de intenciones a través de un micrófono:

- Cree un objeto `SpeechConfig` a partir de la clave y la región de suscripción.
- Cree un objeto `IntentRecognizer` con el objeto `SpeechConfig` anterior.
- Con el objeto `IntentRecognizer`, inicie el proceso de reconocimiento de una única expresión.
- Inspeccione el objeto `IntentRecognitionResult` devuelto.

Si prefiere ponerse a trabajar de inmediato, vea o descargue todos los [ejemplos para C# del SDK de Voz](#) en GitHub. Si no, vamos a comenzar.

## Prerequisites

Antes de comenzar:

- Si este es su primer proyecto en C#, use esta guía para [crear un proyecto de ejemplo vacío](#).
- [Instale el SDK de Voz de su entorno de desarrollo](#).

## Creación de una aplicación de LUIS para el reconocimiento de la intención

Para completar el inicio rápido de reconocimiento de la intención, deberá crear una cuenta de LUIS y un proyecto mediante el portal de la versión preliminar de LUIS. Este inicio rápido solo requiere una suscripción a LUIS. No se requiere una suscripción al servicio de voz.

Lo primero que debe hacer es crear una cuenta y una aplicación de LUIS mediante el portal de vista previa de LUIS. La aplicación de LUIS que cree usará un dominio precompilado para la automatización doméstica, que proporciona intenciones, entidades y expresiones de ejemplo. Cuando termine, tendrá un punto de conexión de LUIS que se ejecuta en la nube al que puede llamar mediante el SDK de Voz.

Siga estas instrucciones para crear una aplicación de LUIS:

- [Inicio rápido: Compilación de una aplicación de un dominio creado previamente](#)

Cuando haya terminado, necesitará tres cosas:

- Su clave de LUIS
- Su región de LUIS
- El identificador de la aplicación de LUIS

Aquí es donde puede encontrar esta información en el [portal de vista previa de LUIS](#):

1. En el portal de vista previa de LUIS, seleccione **Administrary**, después, seleccione **Recursos de Azure**. En esta página, encontrará la clave y la ubicación de LUIS (que a veces se denomina *región*).

The screenshot shows the LUIS Admin interface. The top navigation bar has tabs for DASHBOARD, BUILD, and MANAGE, with MANAGE being the active tab and highlighted with a red box. Below the tabs, there are four main sections: Application Settings, Publish Settings, Azure Resources, and Versions. The Azure Resources section is also highlighted with a red box. It contains fields for Location (set to westus), Primary Key (a long string of characters), Secondary Key (another long string), Endpoint URL (https://contoso.cognitiveservices.azure.com/), and Pricing Tier (S0 (Standard)).

2. Una vez que tenga la clave y ubicación, necesitará el identificador de la aplicación. Seleccione **Configuración de la aplicación** (el identificador de la aplicación está disponible en esta página).

The screenshot shows the LUIS Admin interface with the Application Settings page selected. The left sidebar has sections for Application Settings, Publish Settings, Azure Resources, and Versions, with Application Settings highlighted with a red box. The main content area is titled "Application Settings". It contains fields for Name (set to intent-recognition), App ID (a long string of characters), Description (Intent recognition with the Speech SDK), and Culture (set to en-us).

## Abra el proyecto en Visual Studio.

Después abra el proyecto en Visual Studio.

1. Inicie Visual Studio 2019.
2. Cargue el proyecto y abra `Program.cs`.

## Inicio con código reutilizable

Vamos a agregar código que funcione como el esqueleto del proyecto. Tenga en cuenta que ha creado un método asíncrono llamado `RecognizeIntentAsync()`.

```
using System;
using System.Threading.Tasks;
using Microsoft.CognitiveServices.Speech;
using Microsoft.CognitiveServices.Speech.Intent;

namespace helloworld
{
    class Program
    {
        public static async Task RecognizeIntentAsync()
        {
        }

        static void Main()
        {
            RecognizeIntentAsync().Wait();
            Console.WriteLine("Please press <Return> to continue.");
            Console.ReadLine();
        }
    }
}
```

# Creación de una configuración de Voz

Para poder inicializar un objeto `IntentRecognizer`, es preciso crear una configuración que use la clave y ubicación del recurso de predicción de LUIS.

## IMPORTANT

La clave de inicio y las claves de creación no funcionarán. Debe usar la clave de predicción y la ubicación que creó anteriormente. Para más información, consulte [Creación de una aplicación de LUIS para el reconocimiento de la intención](#).

Inserte este código en el método `RecognizeIntentAsync()`. Asegúrese de actualizar estos valores:

- Reemplace `"YourLanguageUnderstandingSubscriptionKey"` por la clave de predicción de LUIS.
- Reemplace `"YourLanguageUnderstandingServiceRegion"` por la ubicación de LUIS.

## TIP

Si necesita ayuda para encontrar estos valores, consulte [Creación de una aplicación de LUIS para el reconocimiento de la intención](#).

```
var config = SpeechConfig.FromSubscription("YourLanguageUnderstandingSubscriptionKey",
    "YourLanguageUnderstandingServiceRegion");
```

En este ejemplo se usa el método `FromSubscription()` para compilar la clase `SpeechConfig`. Para ver una lista completa de los métodos disponibles, consulte [Clase SpeechConfig](#).

El SDK de Voz se usará de forma predeterminada para reconocer el uso de en-us como idioma. Para más información sobre cómo elegir el idioma de origen, consulte [Especificación del idioma de origen para la conversión de voz a texto](#).

## Inicialización de IntentRecognizer

Ahora, vamos a crear un objeto `IntentRecognizer`. Este objeto se crea dentro de una instrucción `using` para garantizar la versión correcta de los recursos no administrados. Inserte este código en el método `RecognizeIntentAsync()`, justo debajo de la configuración de Voz.

```
// Creates an intent recognizer using microphone as audio input.
using (var recognizer = new IntentRecognizer(config))
{}
```

## Adición de un objeto LanguageUnderstandingModel e intenciones

Debe asociar un objeto `LanguageUnderstandingModel` con el reconocedor de intenciones y agregar las intenciones que desee que se reconozcan. Vamos a usar las intenciones del dominio precompilado para la automatización doméstica. Inserte este código en la instrucción `using` de la sección anterior. Asegúrese de reemplazar `"YourLanguageUnderstandingAppId"` por el identificador de la aplicación de LUIS.

**TIP**

Si necesita ayuda para encontrar este valor, consulte [Creación de una aplicación de LUIS para el reconocimiento de la intención](#).

```
// Creates a Language Understanding model using the app id, and adds specific intents from your model
var model = LanguageUnderstandingModel.FromAppId("YourLanguageUnderstandingAppId");
recognizer.AddIntent(model, "YourLanguageUnderstandingIntentName1", "id1");
recognizer.AddIntent(model, "YourLanguageUnderstandingIntentName2", "id2");
recognizer.AddIntent(model, "YourLanguageUnderstandingIntentName3", "any-IntentId-here");
```

## Reconocimiento de una intención

En el objeto `IntentRecognizer`, va a llamar al método `RecognizeOnceAsync()`. Este método permite que el servicio Voz sepa que solo va a enviar una frase para el reconocimiento y que, una vez que se identifica la frase, se detendrá el reconocimiento de voz.

Dentro de la instrucción `using`, agregue este código debajo del modelo: [!code-csharp]

## Visualización de los resultados (o errores) del reconocimiento

Cuando el servicio Voz devuelva el resultado del reconocimiento, querrá hacer algo con él. Vamos a imprimir el resultado en la consola.

Dentro de la instrucción `using`, debajo de `RecognizeOnceAsync()`, agregue este código:

```
// Checks result.
if (result.Reason == ResultReason.RecognizedIntent)
{
    Console.WriteLine($"RECOGNIZED: Text={result.Text}");
    Console.WriteLine($"    Intent Id: {result.IntentId}.");
    Console.WriteLine($"    Language Understanding JSON:");
    {result.Properties.GetProperty(PropertyId.LanguageUnderstandingServiceResponse_JsonResult)}.");
}
else if (result.Reason == ResultReason.RecognizedSpeech)
{
    Console.WriteLine($"RECOGNIZED: Text={result.Text}");
    Console.WriteLine($"    Intent not recognized.");
}
else if (result.Reason == ResultReason.NoMatch)
{
    Console.WriteLine($"NOMATCH: Speech could not be recognized.");
}
else if (result.Reason == ResultReason.Canceled)
{
    var cancellation = CancellationDetails.FromResult(result);
    Console.WriteLine($"CANCELED: Reason={cancellation.Reason}");

    if (cancellation.Reason == CancellationReason.Error)
    {
        Console.WriteLine($"CANCELED: ErrorCode={cancellation.ErrorCode}");
        Console.WriteLine($"CANCELED: ErrorDetails={cancellation.ErrorDetails}");
        Console.WriteLine($"CANCELED: Did you update the subscription info?");
    }
}
```

## Comprobación del código

En este momento, el código debe tener esta apariencia:

**NOTE**

Se han agregado algunos comentarios a esta versión.

```
// <code>
using System;
using System.Threading.Tasks;
using Microsoft.CognitiveServices.Speech;
using Microsoft.CognitiveServices.Speech.Intent;

namespace helloworld
{
    class Program
    {
        public static async Task RecognizeIntentAsync()
        {
            // Creates an instance of a speech config with specified subscription key
            // and service region. Note that in contrast to other services supported by
            // the Cognitive Services Speech SDK, the Language Understanding service
            // requires a specific subscription key from https://www.luis.ai/.
            // The Language Understanding service calls the required key 'endpoint key'.
            // Once you've obtained it, replace with below with your own Language Understanding subscription
key
            // and service region (e.g., "westus").
            // The default language is "en-us".
            var config = SpeechConfig.FromSubscription("YourLanguageUnderstandingSubscriptionKey",
"YourLanguageUnderstandingServiceRegion");

            // Creates an intent recognizer using microphone as audio input.
            using (var recognizer = new IntentRecognizer(config))
            {
                // Creates a Language Understanding model using the app id, and adds specific intents from
your model
                var model = LanguageUnderstandingModel.FromAppId("YourLanguageUnderstandingAppId");
                recognizer.AddIntent(model, "YourLanguageUnderstandingIntentName1", "id1");
                recognizer.AddIntent(model, "YourLanguageUnderstandingIntentName2", "id2");
                recognizer.AddIntent(model, "YourLanguageUnderstandingIntentName3", "any-IntentId-here");

                // Starts recognizing.
                Console.WriteLine("Say something...");

                // Starts intent recognition, and returns after a single utterance is recognized. The end of
a
                // single utterance is determined by listening for silence at the end or until a maximum of
15
                // seconds of audio is processed. The task returns the recognition text as result.
                // Note: Since RecognizeOnceAsync() returns only a single utterance, it is suitable only for
single
                // shot recognition like command or query.
                // For long-running multi-utterance recognition, use StartContinuousRecognitionAsync()
instead.
                var result = await recognizer.RecognizeOnceAsync().ConfigureAwait(false);

                // Checks result.
                if (result.Reason == ResultReason.RecognizedIntent)
                {
                    Console.WriteLine($"RECOGNIZED: Text={result.Text}");
                    Console.WriteLine($"      Intent Id: {result	IntentId}.");
                    Console.WriteLine($"      Language Understanding JSON:
{result.Properties.GetProperty(PropertyId.LanguageUnderstandingServiceResponse_JsonResult)}.");
                }
                else if (result.Reason == ResultReason.RecognizedSpeech)
                {

```

```

        Console.WriteLine($"RECOGNIZED: Text={result.Text}");
        Console.WriteLine($"    Intent not recognized.");
    }
    else if (result.Reason == ResultReason.NoMatch)
    {
        Console.WriteLine($"NOMATCH: Speech could not be recognized.");
    }
    else if (result.Reason == ResultReason.Canceled)
    {
        var cancellation = CancellationDetails.FromResult(result);
        Console.WriteLine($"CANCELED: Reason={cancellation.Reason}");

        if (cancellation.Reason == CancellationReason.Error)
        {
            Console.WriteLine($"CANCELED: ErrorCode={cancellation.ErrorCode}");
            Console.WriteLine($"CANCELED: ErrorDetails={cancellation.ErrorDetails}");
            Console.WriteLine($"CANCELED: Did you update the subscription info?");
        }
    }
}

static void Main()
{
    RecognizeIntentAsync().Wait();
    Console.WriteLine("Please press <Return> to continue.");
    Console.ReadLine();
}
}
}

```

## Compilación y ejecución de la aplicación

Ya está listo para compilar la aplicación y probar el reconocimiento de voz con el servicio Voz.

1. **Compile el código:** en la barra de menús de Visual Studio, elija **Compilar > Compilar solución**.
2. **Inicie la aplicación:** en la barra de menús, elija **Depurar > Iniciar depuración** o presione **F5**.
3. **Inicie el reconocimiento:** se le pedirá que diga una frase en inglés. La voz se envía al servicio Voz, se transcribe como texto y se representa en la consola.

## Pasos siguientes

[Exploración de ejemplos de C# en GitHub](#)

En este inicio rápido, usará el [SDK de Voz](#) y el servicio Language Understanding (LUIS) para reconocer los intentos de los datos de audio capturados de un micrófono. En concreto, usará el SDK de Voz para capturar la voz y un dominio predefinido de LUIS para identificar las intenciones de automatización doméstica, como encender y apagar una luz.

Tras cumplir algunos requisitos previos, no es preciso seguir muchos pasos para el reconocimiento de voz y la identificación de intenciones a través de un micrófono:

- Cree un objeto `SpeechConfig` a partir de la clave y la región de suscripción.
- Cree un objeto `IntentRecognizer` con el objeto `SpeechConfig` anterior.
- Con el objeto `IntentRecognizer`, inicie el proceso de reconocimiento de una única expresión.
- Inspeccione el objeto `IntentRecognitionResult` devuelto.

Si prefiere ponerse a trabajar de inmediato, vea o descargue todos los [ejemplos para C++ del SDK de Voz](#) en GitHub. Si no, vamos a comenzar.

# Prerequisites

Antes de comenzar:

- Si este es su primer proyecto en C++, use esta guía para [crear un proyecto de ejemplo vacío](#).
- [Instale el SDK de Voz de su entorno de desarrollo](#).

## Creación de una aplicación de LUIS para el reconocimiento de la intención

Para completar el inicio rápido de reconocimiento de la intención, deberá crear una cuenta de LUIS y un proyecto mediante el portal de la versión preliminar de LUIS. Este inicio rápido solo requiere una suscripción a LUIS. No se requiere una suscripción al servicio de voz.

Lo primero que debe hacer es crear una cuenta y una aplicación de LUIS mediante el portal de vista previa de LUIS. La aplicación de LUIS que cree usará un dominio precompilado para la automatización doméstica, que proporciona intenciones, entidades y expresiones de ejemplo. Cuando termine, tendrá un punto de conexión de LUIS que se ejecuta en la nube al que puede llamar mediante el SDK de Voz.

Siga estas instrucciones para crear una aplicación de LUIS:

- [Inicio rápido: Compilación de una aplicación de un dominio creado previamente](#)

Cuando haya terminado, necesitará tres cosas:

- Su clave de LUIS
- Su región de LUIS
- El identificador de la aplicación de LUIS

Aquí es donde puede encontrar esta información en el [portal de vista previa de LUIS](#):

1. En el portal de vista previa de LUIS, seleccione **Administrar**, después, seleccione **Recursos de Azure**. En esta página, encontrará la clave y la ubicación de LUIS (que a veces se denomina *región*).

The screenshot shows the LUIS Admin interface. At the top, there's a navigation bar with tabs: DASHBOARD, BUILD, MANAGE (which is highlighted with a red box), Train, Publish, and Test. Below the navigation bar, there's a sidebar with links: Application Settings, Publish Settings, Azure Resources (which is highlighted with a red box), and Versions. The main content area displays application settings for 'intent-recognition (V0.1)'. It includes fields for Location ('westus'), Primary Key (a long string of characters), Secondary Key (another long string), Endpoint URL ('https://contoso.cognitiveservices.azure.com/'), and Pricing Tier ('S0 (Standard)').

2. Una vez que tenga la clave y ubicación, necesitará el identificador de la aplicación. Seleccione **Configuración de la aplicación** (el identificador de la aplicación está disponible en esta página).

The screenshot shows the LUIS Admin interface. At the top, there's a navigation bar with tabs: DASHBOARD, BUILD, MANAGE (which is highlighted with a red box), Train, Publish, and Test. Below the navigation bar, there's a sidebar with links: Application Settings (which is highlighted with a red box), Publish Settings, Azure Resources, and Versions. The main content area is titled 'Application Settings'. It includes fields for Name (\* 'intent-recognition') and App ID (a long string of characters). There are also fields for Description ('Intent recognition with the Speech SDK') and Culture ('en-us').

Abra el proyecto en Visual Studio.

Después abra el proyecto en Visual Studio.

1. Inicie Visual Studio 2019.
2. Cargue el proyecto y abra `helloworld.cpp`.

## Inicio con código reutilizable

Vamos a agregar código que funcione como el esqueleto del proyecto. Tenga en cuenta que ha creado un método asíncrono llamado `recognizeIntent()`.

```
#include "stdafx.h"
// <code>
#include <iostream>
#include <speechapi_cxx.h>

using namespace std;
using namespace Microsoft::CognitiveServices::Speech;
using namespace Microsoft::CognitiveServices::Speech::Intent;

void recognizeIntent()
{
}

int wmain()
{
    recognizeIntent();
    cout << "Please press a key to continue.\n";
    cin.get();
    return 0;
}
```

## Creación de una configuración de Voz

Para poder inicializar un objeto `IntentRecognizer`, es preciso crear una configuración que use la clave y ubicación del recurso de predicción de LUIS.

### IMPORTANT

La clave de inicio y las claves de creación no funcionarán. Debe usar la clave de predicción y la ubicación que creó anteriormente. Para más información, consulte [Creación de una aplicación de LUIS para el reconocimiento de la intención](#).

Inserte este código en el método `recognizeIntent()`. Asegúrese de actualizar estos valores:

- Reemplace `"YourLanguageUnderstandingSubscriptionKey"` por la clave de predicción de LUIS.
- Reemplace `"YourLanguageUnderstandingServiceRegion"` por la ubicación de LUIS.

### TIP

Si necesita ayuda para encontrar estos valores, consulte [Creación de una aplicación de LUIS para el reconocimiento de la intención](#).

```
auto config = SpeechConfig::FromSubscription("YourLanguageUnderstandingSubscriptionKey",
"YourLanguageUnderstandingServiceRegion");
```

En este ejemplo se usa el método `FromSubscription()` para compilar la clase `SpeechConfig`. Para ver una lista

completa de los métodos disponibles, consulte [Clase SpeechConfig](#).

El SDK de Voz se usará de forma predeterminada para reconocer el uso de en-us como idioma. Para más información sobre cómo elegir el idioma de origen, consulte [Especificación del idioma de origen para la conversión de voz a texto](#).

## Inicialización de IntentRecognizer

Ahora, vamos a crear un objeto `IntentRecognizer`. Inserte este código en el método `recognizeIntent()`, justo debajo de la configuración de Voz.

```
auto recognizer = IntentRecognizer::FromConfig(config);
```

## Adición de un objeto LanguageUnderstandingModel e intenciones

Debe asociar un objeto `LanguageUnderstandingModel` con el reconocedor de intenciones y agregar las intenciones que desee que se reconozcan. Vamos a usar las intenciones del dominio precompilado para la automatización doméstica.

Inserte este código debajo de `IntentRecognizer`. Asegúrese de reemplazar `"YourLanguageUnderstandingAppId"` por el identificador de la aplicación de LUIS.

### TIP

Si necesita ayuda para encontrar este valor, consulte [Creación de una aplicación de LUIS para el reconocimiento de la intención](#).

```
auto model = LanguageUnderstandingModel::FromAppId("YourLanguageUnderstandingAppId");
recognizer->AddIntent(model, "YourLanguageUnderstandingIntentName1", "id1");
recognizer->AddIntent(model, "YourLanguageUnderstandingIntentName2", "id2");
recognizer->AddIntent(model, "YourLanguageUnderstandingIntentName3", "any-IntentId-here");
```

## Reconocimiento de una intención

En el objeto `IntentRecognizer`, va a llamar al método `RecognizeOnceAsync()`. Este método permite que el servicio Voz sepa que solo va a enviar una frase para el reconocimiento y que, una vez que se identifica la frase, se detendrá el reconocimiento de voz. Por motivos de simplicidad, esperaremos a que se complete la devolución futura.

Inserte este código debajo del modelo:

```
auto result = recognizer->RecognizeOnceAsync().get();
```

## Visualización de los resultados (o errores) del reconocimiento

Cuando el servicio Voz devuelva el resultado del reconocimiento, querrá hacer algo con él. Vamos a hacer algo tan sencillo como imprimir el resultado en la consola.

Inserte este código debajo de `auto result = recognizer->RecognizeOnceAsync().get();`:

```

if (result->Reason == ResultReason::RecognizedIntent)
{
    cout << "RECOGNIZED: Text=" << result->Text << std::endl;
    cout << " Intent Id: " << result->IntentId << std::endl;
    cout << " Intent Service JSON: " << result-
>Properties.GetProperty(PropertyId::LanguageUnderstandingServiceResponse_JsonResult) << std::endl;
}
else if (result->Reason == ResultReason::RecognizedSpeech)
{
    cout << "RECOGNIZED: Text=" << result->Text << " (intent could not be recognized)" << std::endl;
}
else if (result->Reason == ResultReason::NoMatch)
{
    cout << "NOMATCH: Speech could not be recognized." << std::endl;
}
else if (result->Reason == ResultReason::Canceled)
{
    auto cancellation = CancellationDetails::FromResult(result);
    cout << "CANCELED: Reason=" << (int)cancellation->Reason << std::endl;

    if (cancellation->Reason == CancellationReason::Error)
    {
        cout << "CANCELED: ErrorCode=" << (int)cancellation->ErrorCode << std::endl;
        cout << "CANCELED: ErrorDetails=" << cancellation->ErrorDetails << std::endl;
        cout << "CANCELED: Did you update the subscription info?" << std::endl;
    }
}

```

## Comprobación del código

En este momento, el código debe tener esta apariencia:

### NOTE

Se han agregado algunos comentarios a esta versión.

```

#include "stdafx.h"
// <code>
#include <iostream>
#include <speechapi_cxx.h>

using namespace std;
using namespace Microsoft::CognitiveServices::Speech;
using namespace Microsoft::CognitiveServices::Speech::Intent;

void recognizeIntent()
{
    // Creates an instance of a speech config with specified subscription key
    // and service region. Note that in contrast to other services supported by
    // the Cognitive Services Speech SDK, the Language Understanding service
    // requires a specific subscription key from https://www.luis.ai/.
    // The Language Understanding service calls the required key 'endpoint key'.
    // Once you've obtained it, replace with below with your own Language Understanding subscription key
    // and service region (e.g., "westus").
    // The default recognition language is "en-us".
    auto config = SpeechConfig::FromSubscription("YourLanguageUnderstandingSubscriptionKey",
"YourLanguageUnderstandingServiceRegion");

    // Creates an intent recognizer using microphone as audio input.
    auto recognizer = IntentRecognizer::FromConfig(config);

    // Creates a Language Understanding model using the app id, and adds specific intents from your model
    auto model = LanguageUnderstandingModel::FromAppId("YourLanguageUnderstandingAppId");

```

```

recognizer->AddIntent(model, "YourLanguageUnderstandingIntentName1", "id1");
recognizer->AddIntent(model, "YourLanguageUnderstandingIntentName2", "id2");
recognizer->AddIntent(model, "YourLanguageUnderstandingIntentName3", "any-IntentId-here");

cout << "Say something...\n";

// Starts intent recognition, and returns after a single utterance is recognized. The end of a
// single utterance is determined by listening for silence at the end or until a maximum of 15
// seconds of audio is processed. The task returns the recognition text as result.
// Note: Since RecognizeOnceAsync() returns only a single utterance, it is suitable only for single
// shot recognition like command or query.
// For long-running multi-utterance recognition, use StartContinuousRecognitionAsync() instead.
auto result = recognizer->RecognizeOnceAsync().get();

// Checks result.
if (result->Reason == ResultReason::RecognizedIntent)
{
    cout << "RECOGNIZED: Text=" << result->Text << std::endl;
    cout << " Intent Id: " << result->IntentId << std::endl;
    cout << " Intent Service JSON: " << result-
>Properties.GetProperty(PropertyId::LanguageUnderstandingServiceResponse_JsonResult) << std::endl;
}
else if (result->Reason == ResultReason::RecognizedSpeech)
{
    cout << "RECOGNIZED: Text=" << result->Text << " (intent could not be recognized)" << std::endl;
}
else if (result->Reason == ResultReason::NoMatch)
{
    cout << "NOMATCH: Speech could not be recognized." << std::endl;
}
else if (result->Reason == ResultReason::Canceled)
{
    auto cancellation = CancellationDetails::FromResult(result);
    cout << "CANCELED: Reason=" << (int)cancellation->Reason << std::endl;

    if (cancellation->Reason == CancellationReason::Error)
    {
        cout << "CANCELED: ErrorCode=" << (int)cancellation->ErrorCode << std::endl;
        cout << "CANCELED: ErrorDetails=" << cancellation->ErrorDetails << std::endl;
        cout << "CANCELED: Did you update the subscription info?" << std::endl;
    }
}
}

int wmain()
{
    recognizeIntent();
    cout << "Please press a key to continue.\n";
    cin.get();
    return 0;
}

```

## Compilación y ejecución de la aplicación

Ya está listo para compilar la aplicación y probar el reconocimiento de voz con el servicio Voz.

- Compile el código:** en la barra de menús de Visual Studio, elija **Compilar > Compilar solución**.
- Inicie la aplicación:** en la barra de menús, elija **Depurar > Iniciar depuración** o presione **F5**.
- Inicie el reconocimiento:** se le pedirá que diga una frase en inglés. La voz se envía al servicio Voz, se transcribe como texto y se representa en la consola.

## Pasos siguientes

[Exploración de ejemplos de C++ en GitHub](#)

---

En este inicio rápido, usará el [SDK de Voz](#) y el servicio Language Understanding (LUIS) para reconocer los intentos de los datos de audio capturados de un micrófono. En concreto, usará el SDK de Voz para capturar la voz y un dominio predefinido de LUIS para identificar las intenciones de automatización doméstica, como encender y apagar una luz.

Tras cumplir algunos requisitos previos, no es preciso seguir muchos pasos para el reconocimiento de voz y la identificación de intenciones a través de un micrófono:

- Cree un objeto `SpeechConfig` a partir de la clave y la región de suscripción.
- Cree un objeto `IntentRecognizer` con el objeto `SpeechConfig` anterior.
- Con el objeto `IntentRecognizer`, inicie el proceso de reconocimiento de una única expresión.
- Inspeccione el objeto `IntentRecognitionResult` devuelto.

Si prefiere ponerse a trabajar de inmediato, vea o descargue todos los [ejemplos para Java del SDK de Voz](#) en GitHub. Si no, vamos a comenzar.

## Prerequisites

Antes de comenzar:

- Si este es su primer proyecto en Java (JRE), use esta guía para [crear un proyecto de ejemplo vacío](#).
- [Instale el SDK de Voz de su entorno de desarrollo](#).

## Creación de una aplicación de LUIS para el reconocimiento de la intención

Para completar el inicio rápido de reconocimiento de la intención, deberá crear una cuenta de LUIS y un proyecto mediante el portal de la versión preliminar de LUIS. Este inicio rápido solo requiere una suscripción a LUIS. No se requiere una suscripción al servicio de voz.

Lo primero que debe hacer es crear una cuenta y una aplicación de LUIS mediante el portal de vista previa de LUIS. La aplicación de LUIS que cree usará un dominio precompilado para la automatización doméstica, que proporciona intenciones, entidades y expresiones de ejemplo. Cuando termine, tendrá un punto de conexión de LUIS que se ejecuta en la nube al que puede llamar mediante el SDK de Voz.

Siga estas instrucciones para crear una aplicación de LUIS:

- [Inicio rápido: Compilación de una aplicación de un dominio creado previamente](#)

Cuando haya terminado, necesitará tres cosas:

- Su clave de LUIS
- Su región de LUIS
- El identificador de la aplicación de LUIS

Aquí es donde puede encontrar esta información en el [portal de vista previa de LUIS](#):

1. En el portal de vista previa de LUIS, seleccione **Administrador**, después, seleccione **Recursos de Azure**. En esta página, encontrará la clave y la ubicación de LUIS (que a veces se denomina *región*).

intent-recognition (V0.1) ▾

DASHBOARD BUILD MANAGE Train Publish Test

Application Settings

Azure Resources

Publish Settings

Versions

Location: westus

Primary Key: xxxxxxxxxxxxxxxxxxxxxxxxx

Secondary Key: xxxxxxxxxxxxxxxxxxxxxxxxx

Endpoint URL: https://contoso.cognitiveservices.azure.com/

Pricing Tier: S0 (Standard)

- Una vez que tenga la clave y ubicación, necesitará el identificador de la aplicación. Seleccione **Configuración de la aplicación** (el identificador de la aplicación está disponible en esta página).

intent-recognition (V0.1) ▾

DASHBOARD BUILD MANAGE Train Publish Test

Application Settings

Publish Settings

Azure Resources

Versions

Name \* intent-recognition

App ID xxxxxxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx

Description Intent recognition with the Speech SDK

Culture en-us

## Apertura del proyecto

- Abra el entorno de desarrollo integrado que prefiera.
- Cargue el proyecto y abra `Main.java`.

## Inicio con código reutilizable

Vamos a agregar código que funcione como el esqueleto del proyecto.

```
package speechsdk.quickstart;

import com.microsoft.cognitiveservices.speech.*;
import com.microsoft.cognitiveservices.speech.intent.*;

/**
 * Quickstart: recognize speech using the Speech SDK for Java.
 */
public class Main {

    /**
     * @param args Arguments are ignored in this sample.
     */
    public static void main(String[] args) {
        try {
        } catch (Exception ex) {
            System.out.println("Unexpected exception: " + ex.getMessage());

            assert(false);
            System.exit(1);
        }
    }
}
```

## Creación de una configuración de Voz

Para poder inicializar un objeto `IntentRecognizer`, es preciso crear una configuración que use la clave y ubicación del recurso de predicción de LUIS.

Inserte este código en el bloque try/catch en `main()`. Asegúrese de actualizar estos valores:

- Reemplace `"YourLanguageUnderstandingSubscriptionKey"` por la clave de predicción de LUIS.
- Reemplace `"YourLanguageUnderstandingServiceRegion"` por la ubicación de LUIS.

#### TIP

Si necesita ayuda para encontrar estos valores, consulte [Creación de una aplicación de LUIS para el reconocimiento de la intención](#).

```
SpeechConfig config = SpeechConfig.fromSubscription("YourLanguageUnderstandingSubscriptionKey",  
"YourLanguageUnderstandingServiceRegion");
```

En este ejemplo se usa el método `FromSubscription()` para compilar la clase `SpeechConfig`. Para ver una lista completa de los métodos disponibles, consulte [Clase SpeechConfig](#).

El SDK de Voz se usará de forma predeterminada para reconocer el uso de en-us como idioma. Para más información sobre cómo elegir el idioma de origen, consulte [Especificación del idioma de origen para la conversión de voz a texto](#).

## Inicialización de IntentRecognizer

Ahora, vamos a crear un objeto `IntentRecognizer`. Inserte este código justo debajo de la configuración de Voz.

```
IntentRecognizer recognizer = new IntentRecognizer(config);
```

## Adición de un objeto LanguageUnderstandingModel e intenciones

Debe asociar un objeto `LanguageUnderstandingModel` con el reconocedor de intenciones y agregar las intenciones que desee que se reconozcan. Vamos a usar las intenciones del dominio precompilado para la automatización doméstica.

Inserte este código debajo de `IntentRecognizer`. Asegúrese de reemplazar `"YourLanguageUnderstandingAppId"` por el identificador de la aplicación de LUIS.

#### TIP

Si necesita ayuda para encontrar este valor, consulte [Creación de una aplicación de LUIS para el reconocimiento de la intención](#).

```
LanguageUnderstandingModel model = LanguageUnderstandingModel.fromAppId("YourLanguageUnderstandingAppId");  
recognizer.addIntent(model, "YourLanguageUnderstandingIntentName1", "id1");  
recognizer.addIntent(model, "YourLanguageUnderstandingIntentName2", "id2");  
recognizer.addIntent(model, "YourLanguageUnderstandingIntentName3", "any-IntentId-here");
```

## Reconocimiento de una intención

En el objeto `IntentRecognizer`, va a llamar al método `recognizeOnceAsync()`. Este método permite que el servicio Voz sepa que solo va a enviar una frase para el reconocimiento y que, una vez que se identifica la frase, se detendrá el reconocimiento de voz.

Inserte este código debajo del modelo:

```
IntentRecognitionResult result = recognizer.recognizeOnceAsync().get();
```

## Visualización de los resultados (o errores) del reconocimiento

Cuando el servicio Voz devuelva el resultado del reconocimiento, querrá hacer algo con él. Vamos a hacer algo tan sencillo como imprimir el resultado en la consola.

Inserte este código debajo de la llamada a `recognizeOnceAsync()` : [!code-java]

## Liberación de recursos

Es importante liberar los recursos de voz cuando haya terminado de usarlos. Inserte este código al final del bloque try/catch:

```
result.close();
recognizer.close();
```

## Comprobación del código

En este momento, el código debe tener esta apariencia:

### NOTE

Se han agregado algunos comentarios a esta versión.

```
package speechsdk.quickstart;

import com.microsoft.cognitiveservices.speech.*;
import com.microsoft.cognitiveservices.speech.intent.*;

/**
 * Quickstart: recognize speech using the Speech SDK for Java.
 */
public class Main {

    /**
     * @param args Arguments are ignored in this sample.
     */
    public static void main(String[] args) {
        try {
            // <IntentRecognitionWithMicrophone>
            // Creates an instance of a speech config with specified
            // subscription key (called 'endpoint key' by the Language Understanding service)
            // and service region. Replace with your own subscription (endpoint) key
            // and service region (e.g., "westus2").
            // The default language is "en-us".
            SpeechConfig config = SpeechConfig.fromSubscription("YourLanguageUnderstandingSubscriptionKey",
"YourLanguageUnderstandingServiceRegion");

            // Creates an intent recognizer using microphone as audio input.
            IntentRecognizer recognizer = new IntentRecognizer(config);

            // Creates a language understanding model using the app id, and adds specific intents from your
            // model
            LanguageUnderstandingModel model =
            LanguageUnderstandingModel.fromAppId("YourLanguageUnderstandingAppId");
        }
    }
}
```

```

recognizer.addIntent(model, "YourLanguageUnderstandingIntentName1", "id1");
recognizer.addIntent(model, "YourLanguageUnderstandingIntentName2", "id2");
recognizer.addIntent(model, "YourLanguageUnderstandingIntentName3", "any-IntentId-here");

System.out.println("Say something...");

// Starts recognition. It returns when the first utterance has been recognized.
IntentRecognitionResult result = recognizer.recognizeOnceAsync().get();

// Checks result.
if (result.getReason() == ResultReason.RecognizedIntent) {
    System.out.println("RECOGNIZED: Text=" + result.getText());
    System.out.println("    Intent Id: " + result.getId());
    System.out.println("    Intent Service JSON: " +
result.getProperties().getProperty(PropertyId.LanguageUnderstandingServiceResponse_JsonResult));
}
else if (result.getReason() == ResultReason.RecognizedSpeech) {
    System.out.println("RECOGNIZED: Text=" + result.getText());
    System.out.println("    Intent not recognized.");
}
else if (result.getReason() == ResultReason.NoMatch) {
    System.out.println("NOMATCH: Speech could not be recognized.");
}
else if (result.getReason() == ResultReason.Canceled) {
    CancellationDetails cancellation = CancellationDetails.fromResult(result);
    System.out.println("CANCELED: Reason=" + cancellation.getReason());

    if (cancellation.getReason() == CancellationReason.Error) {
        System.out.println("CANCELED: ErrorCode=" + cancellation.getErrorCode());
        System.out.println("CANCELED: ErrorDetails=" + cancellation.getErrorDetails());
        System.out.println("CANCELED: Did you update the subscription info?");
    }
}
result.close();
recognizer.close();
} catch (Exception ex) {
    System.out.println("Unexpected exception: " + ex.getMessage());

    assert(false);
    System.exit(1);
}
}
}

```

## Compilación y ejecución de la aplicación

Presione F11, o seleccione **Run (Ejecutar) > Debug (Depurar)**. Los próximos 15 segundos de la entrada de voz del micrófono se reconocen y se registran en la ventana de consola.

## Pasos siguientes

[Exploración de ejemplos de Java en GitHub](#)

En este inicio rápido, usará el [SDK de Voz](#) y el servicio Language Understanding (LUIS) para reconocer los intentos de los datos de audio capturados de un micrófono. En concreto, usará el SDK de Voz para capturar la voz y un dominio predefinido de LUIS para identificar las intenciones de automatización doméstica, como encender y apagar una luz.

Tras cumplir algunos requisitos previos, no es preciso seguir muchos pasos para el reconocimiento de voz y la identificación de intenciones a través de un micrófono:

- Cree un objeto `SpeechConfig` a partir de la clave y la región de suscripción.

- Cree un objeto `IntentRecognizer` con el objeto `SpeechConfig` anterior.
- Con el objeto `IntentRecognizer`, inicie el proceso de reconocimiento de una única expresión.
- Inspeccione el objeto `IntentRecognitionResult` devuelto.

Si prefiere ponerse a trabajar de inmediato, vea o descargue todos los [ejemplos para Python del SDK de Voz](#) en GitHub. Si no, vamos a comenzar.

## Prerequisites

Antes de comenzar:

- Si este es su primer proyecto en Python, use esta guía para [crear un proyecto de ejemplo vacío](#).
- [Instale el SDK de Voz de su entorno de desarrollo](#).

## Creación de una aplicación de LUIS para el reconocimiento de la intención

Para completar el inicio rápido de reconocimiento de la intención, deberá crear una cuenta de LUIS y un proyecto mediante el portal de la versión preliminar de LUIS. Este inicio rápido solo requiere una suscripción a LUIS. No se requiere una suscripción al servicio de voz.

Lo primero que debe hacer es crear una cuenta y una aplicación de LUIS mediante el portal de vista previa de LUIS. La aplicación de LUIS que cree usará un dominio precompilado para la automatización doméstica, que proporciona intenciones, entidades y expresiones de ejemplo. Cuando termine, tendrá un punto de conexión de LUIS que se ejecuta en la nube al que puede llamar mediante el SDK de Voz.

Siga estas instrucciones para crear una aplicación de LUIS:

- [Inicio rápido: Compilación de una aplicación de un dominio creado previamente](#)

Cuando haya terminado, necesitará tres cosas:

- Su clave de LUIS
- Su región de LUIS
- El identificador de la aplicación de LUIS

Aquí es donde puede encontrar esta información en el [portal de vista previa de LUIS](#):

1. En el portal de vista previa de LUIS, seleccione **Administrador**, después, seleccione **Recursos de Azure**. En esta página, encontrará la clave y la ubicación de LUIS (que a veces se denomina *región*).

The screenshot shows the LUIS Admin portal interface. At the top, there's a navigation bar with tabs: DASHBOARD, BUILD, MANAGE (which is highlighted with a red box), Train, Publish, and Test. Below the navigation bar, there are several sections: Application Settings, Publish Settings, Azure Resources (which is also highlighted with a red box), and Versions. In the Azure Resources section, there are fields for Location (set to westus), Primary Key (a long string of characters), Secondary Key, Endpoint URL (https://contoso.cognitiveservices.azure.com/), and Pricing Tier (S0 (Standard)).

2. Una vez que tenga la clave y ubicación, necesitará el identificador de la aplicación. Seleccione **Configuración de la aplicación** (el identificador de la aplicación está disponible en esta página).

The screenshot shows the Azure portal interface with the 'intent-recognition (V0.1)' project selected. The top navigation bar includes 'DASHBOARD', 'BUILD', 'MANAGE' (which is highlighted with a red box), 'Train' (with a red dot icon), 'Publish', and 'Test'. On the left sidebar, under 'Application Settings', there are links for 'Publish Settings', 'Azure Resources', and 'Versions'. The main content area is titled 'Application Settings' and contains fields for 'Name \*' (set to 'intent-recognition'), 'App ID' (containing a placeholder GUID), 'Description' ('Intent recognition with the Speech SDK'), and 'Culture' ('en-us').

## Apertura del proyecto

1. Abra el entorno de desarrollo integrado que prefiera.
2. Cree un proyecto y un archivo llamado `quickstart.py`, y, a continuación, ábralo.

## Inicio con código reutilizable

Vamos a agregar código que funcione como el esqueleto del proyecto.

```
import azure.cognitiveservices.speech as speechsdk  
  
print("Say something...")
```

## Creación de una configuración de Voz

Para poder inicializar un objeto `IntentRecognizer`, es preciso crear una configuración que use la clave y ubicación del recurso de predicción de LUIS.

Inserte este código en `quickstart.py`. Asegúrese de actualizar estos valores:

- Reemplace `"YourLanguageUnderstandingSubscriptionKey"` por la clave de predicción de LUIS.
- Reemplace `"YourLanguageUnderstandingServiceRegion"` por la ubicación de LUIS.

### TIP

Si necesita ayuda para encontrar estos valores, consulte [Creación de una aplicación de LUIS para el reconocimiento de la intención](#).

```
intent_config = speechsdk.SpeechConfig(subscription="YourLanguageUnderstandingSubscriptionKey",  
region="YourLanguageUnderstandingServiceRegion")
```

En este ejemplo se crea el objeto `SpeechConfig` mediante la clave y la región de LUIS. Para ver una lista completa de los métodos disponibles, consulte [Clase SpeechConfig](#).

El SDK de Voz se usará de forma predeterminada para reconocer el uso de en-us como idioma. Para más información sobre cómo elegir el idioma de origen, consulte [Especificación del idioma de origen para la conversión de voz a texto](#).

## Inicialización de IntentRecognizer

Ahora, vamos a crear un objeto `IntentRecognizer`. Inserte este código justo debajo de la configuración de Voz.

```
intent_recognizer = speechsdk.intent.IntentRecognizer(speech_config=intent_config)
```

## Adición de un objeto LanguageUnderstandingModel e intenciones

Debe asociar un objeto `LanguageUnderstandingModel` con el reconocedor de intenciones y agregar las intenciones que deseé que se reconozcan. Vamos a usar las intenciones del dominio precompilado para la automatización doméstica.

Inserte este código debajo de `IntentRecognizer`. Asegúrese de reemplazar `"YourLanguageUnderstandingAppId"` por el identificador de la aplicación de LUIS.

### TIP

Si necesita ayuda para encontrar este valor, consulte [Creación de una aplicación de LUIS para el reconocimiento de la intención](#).

```
model = speechsdk.intent.LanguageUnderstandingModel(app_id="YourLanguageUnderstandingAppId")
intents = [
    (model, "HomeAutomation.TurnOn"),
    (model, "HomeAutomation.TurnOff"),
    ("This is a test.", "test"),
    ("Switch to channel 34.", "34"),
    ("what's the weather like", "weather"),
]
intent_recognizer.add_intents(intents)
```

## Reconocimiento de una intención

En el objeto `IntentRecognizer`, va a llamar al método `recognize_once()`. Este método permite que el servicio Voz sepa que solo va a enviar una frase para el reconocimiento y que, una vez que se identifica la frase, se detendrá el reconocimiento de voz.

Inserte este código debajo del modelo:

```
intent_result = intent_recognizer.recognize_once()
```

## Visualización de los resultados (o errores) del reconocimiento

Cuando el servicio Voz devuelva el resultado del reconocimiento, querrá hacer algo con él. Vamos a hacer algo tan sencillo como imprimir el resultado en la consola.

Debajo de la llamada a `recognize_once()`, agregue este código: [!code-python]

## Comprobación del código

En este momento, el código debe tener esta apariencia:

### NOTE

Se han agregado algunos comentarios a esta versión.

```

import azure.cognitiveservices.speech as speechsdk

print("Say something...")

"""performs one-shot intent recognition from input from the default microphone"""
# <IntentRecognitionOnceWithMic>
# Set up the config for the intent recognizer (remember that this uses the Language Understanding key, not
# the Speech Services key)!
intent_config = speechsdk.SpeechConfig(subscription="YourLanguageUnderstandingSubscriptionKey",
region="YourLanguageUnderstandingServiceRegion")

# Set up the intent recognizer
intent_recognizer = speechsdk.intent.IntentRecognizer(speech_config=intent_config)

# set up the intents that are to be recognized. These can be a mix of simple phrases and
# intents specified through a LanguageUnderstanding Model.
model = speechsdk.intent.LanguageUnderstandingModel(app_id="YourLanguageUnderstandingAppId")
intents = [
    (model, "HomeAutomation.TurnOn"),
    (model, "HomeAutomation.TurnOff"),
    ("This is a test.", "test"),
    ("Switch to channel 34.", "34"),
    ("what's the weather like", "weather"),
]
intent_recognizer.add_intents(intents)

# Starts intent recognition, and returns after a single utterance is recognized. The end of a
# single utterance is determined by listening for silence at the end or until a maximum of 15
# seconds of audio is processed. It returns the recognition text as result.
# Note: Since recognize_once() returns only a single utterance, it is suitable only for single
# shot recognition like command or query.
# For long-running multi-utterance recognition, use start_continuous_recognition() instead.
intent_result = intent_recognizer.recognize_once()

# Check the results
if intent_result.reason == speechsdk.ResultReason.RecognizedIntent:
    print("Recognized: \"{}\" with intent id `{}`.".format(intent_result.text, intent_result.intent_id))
elif intent_result.reason == speechsdk.ResultReason.RecognizedSpeech:
    print("Recognized: {}".format(intent_result.text))
elif intent_result.reason == speechsdk.ResultReason.NoMatch:
    print("No speech could be recognized: {}".format(intent_result.no_match_details))
elif intent_result.reason == speechsdk.ResultReason.Canceled:
    print("Intent recognition canceled: {}".format(intent_result.cancellation_details.reason))
    if intent_result.cancellation_details.reason == speechsdk.CancellationReason.Error:
        print("Error details: {}".format(intent_result.cancellation_details.error_details))

```

## Compilación y ejecución de la aplicación

Ejecute el ejemplo desde la consola o en el IDE:

```
python quickstart.py
```

Los próximos 15 segundos de la entrada de voz del micrófono se reconocen y se registran en la ventana de consola.

## Pasos siguientes

[Exploración de los ejemplos de Python en GitHub](#)

Vea o descargue todos los [ejemplos del SDK de Voz](#) en GitHub.

## Compatibilidad con plataformas y lenguajes adicionales

Si ha hecho clic en esta pestaña, es probable que no vea un inicio rápido en su lenguaje de programación favorito. No se preocupe, tenemos materiales de inicio rápido y ejemplos de código adicionales disponibles en GitHub. Use la tabla para encontrar el ejemplo correcto para su lenguaje de programación y combinación de plataforma y sistema operativo.

IDIOMA	INICIOS RÁPIDOS ADICIONALES	EJEMPLOS DE CÓDIGO
C++		<a href="#">Inicios rápidos</a> , <a href="#">Ejemplos</a>
C#		<a href="#">.NET Framework</a> , <a href="#">.NET Core</a> , <a href="#">UWP</a> , <a href="#">Unity</a> , <a href="#">Xamarin</a>
Java		<a href="#">Android</a> , <a href="#">JRE</a>
JavaScript		<a href="#">Browser</a>
Node.js		<a href="#">Windows</a> , <a href="#">Linux</a> , <a href="#">macOS</a>
Objective-C	<a href="#">macOs</a> , <a href="#">iOS</a>	<a href="#">iOS</a> , <a href="#">macOS</a>
Python		<a href="#">Windows</a> , <a href="#">Linux</a> , <a href="#">macOS</a>
Swift	<a href="#">macOs</a> , <a href="#">iOS</a>	<a href="#">iOS</a> , <a href="#">macOS</a>

# Reconocimiento de intenciones a partir de contenido de voz mediante el SDK de Voz para C#

13/01/2020 • 23 minutes to read • [Edit Online](#)

El [SDK de Voz](#) de Cognitive Services se integra con [Language Understanding Intelligent Service \(LUIS\)](#) para proporcionar un **reconocimiento de la intención**. Una intención es algo que el usuario quiere hacer: reservar un vuelo, comprobar el tiempo o hacer una llamada. El usuario puede utilizar cualquier término que le parezca natural. Mediante el aprendizaje automático, LUIS asigna las solicitudes de los usuarios a las intenciones que se hayan definido.

## NOTE

Una aplicación LUIS define las intenciones y entidades que desea reconocer. Es independiente de la aplicación C# que utiliza el servicio Voz. En este artículo, con la palabra "aplicación" se indica tanto la aplicación LUIS, como el código C#.

En esta guía, se utiliza el SDK de Voz para desarrollar una aplicación de consola C# que deriva las intenciones de las expresiones de los usuarios mediante el micrófono del dispositivo. Aprenderá a:

- Crear un proyecto de Visual Studio que haga referencia al paquete NuGet del SDK de Voz
- Crear una configuración de voz y obtener un reconocedor de intenciones
- Obtener el modelo para la aplicación LUIS y agregar las intenciones que necesita
- Especificar el idioma para el reconocimiento de voz
- Reconocer la voz a partir de un archivo
- Usar el reconocimiento asincrónico, continuo y controlado por eventos

## Requisitos previos

Asegúrese de disponer de los siguientes elementos antes de empezar esta guía:

- Una cuenta de LUIS. Puede obtener una gratis mediante el [portal de LUIS](#).
- [Visual Studio 2019](#) (cualquier edición).

## LUIS y voz

LUIS se integra con el servicio Voz para reconocer las intenciones a partir de contenido de voz. No necesita una suscripción al servicio Voz, solo LUIS.

LUIS usa tres tipos de claves:

TIPO DE CLAVE	PROPÓSITO
Creación	Le permite crear y modificar aplicaciones de LUIS mediante programación
Inicio	Permite probar la aplicación de LUIS mediante el uso solo de texto.
Punto de conexión	Autoriza el acceso a una aplicación de LUIS concreta

Para esta guía, necesitará el tipo de clave de punto de conexión. Esta guía utiliza la aplicación de LUIS Home Automation de ejemplo, que se puede crear siguiendo el inicio rápido [Uso de automatización del hogar compilada previamente](#). Si ha creado su propia aplicación de LUIS, puede usarla si lo prefiere.

Al crear una aplicación LUIS, el propio LUIS genera automáticamente una clave de inicio para que pueda probarla aplicación mediante consultas de texto. Esta clave no permite la integración del servicio de voz y no funcionará con esta guía. Cree un recurso de LUIS en el panel de Azure y ásiganelo a la aplicación de LUIS. Puede usar el nivel de suscripción gratis para esta guía.

Después de crear el recurso de LUIS en el panel de Azure, inicie sesión en el [portal de LUIS](#), elija la aplicación en la página **My Apps** (Mis aplicaciones) y, después, cambie a la página **Manage** (Administrar) de la aplicación. Por último, seleccione **Keys and Endpoints** (Claves y puntos de conexión) en la barra lateral.

The screenshot shows the LUIS application management interface. At the top, there's a navigation bar with links for Language Understanding, My apps, Docs, Pricing, Support, and About. Below that is a secondary navigation bar for the 'HomeAutomation (V 0.1)' application, with options for DASHBOARD, BUILD, and MANAGE (which is currently selected). A 'Train' button is also present. On the left, a sidebar lists Application Information, Keys and Endpoints (which is highlighted in blue), Publish Settings, Versions, and Collaborators. The main content area is titled 'Keys and Endpoint settings'. It contains a section for 'Authoring Key' with a file icon. Below this is a table with two rows of data.

Resource Name	Region	Time zone	Key 1	Key 2
Starter_Key	westus	GMT -6:00	a38608 ...	
luis	westus	GMT -6:00	ad544d ...	0e6269 ...

En la página **Keys and Endpoint settings** (Configuración de claves y puntos de conexión):

1. Desplácese hacia abajo hasta la sección **Resources and Keys** (Recursos y claves) y seleccione **Assign resource** (Asignar recurso).
2. En el cuadro de diálogo **Assign a key to your app** (Asignar una clave a la aplicación), realices los siguientes cambios:
  - En **Tenant** (Inquilino), elija **Microsoft**.
  - En **Subscription Name** (Nombre de suscripción), elija la suscripción de Azure que contiene el recurso de LUIS que desea usar.
  - En **Key** (Clave), elija el recurso de LUIS que desea usar con la aplicación.

En unos instantes, la nueva suscripción aparecerá en la tabla en la parte inferior de la página.

3. Seleccione el ícono situado junto a una clave para copiarla al Portapapeles. (Puede usar cualquiera de las claves).

The screenshot shows the 'Assign resource' dialog. At the top, there's a button labeled '+ Assign resource'. Below it is a table with columns for Resource Name, Region, Time zone, Key 1, and Key 2. The table has two rows. In the first row, the 'Key 1' column contains a copy icon followed by the value 'a38608 ...'. In the second row, the 'Key 1' column contains a copy icon followed by the value 'ad544d ...'. The 'Key 2' column for both rows contains a copy icon followed by the value '0e6269 ...'.

Resource Name	Region	Time zone	Key 1	Key 2
Starter_Key	westus	GMT -6:00	a38608 ...	
luis	westus	GMT -6:00	ad544d ...	0e6269 ...

## Creación de un proyecto de contenido de voz en Visual Studio

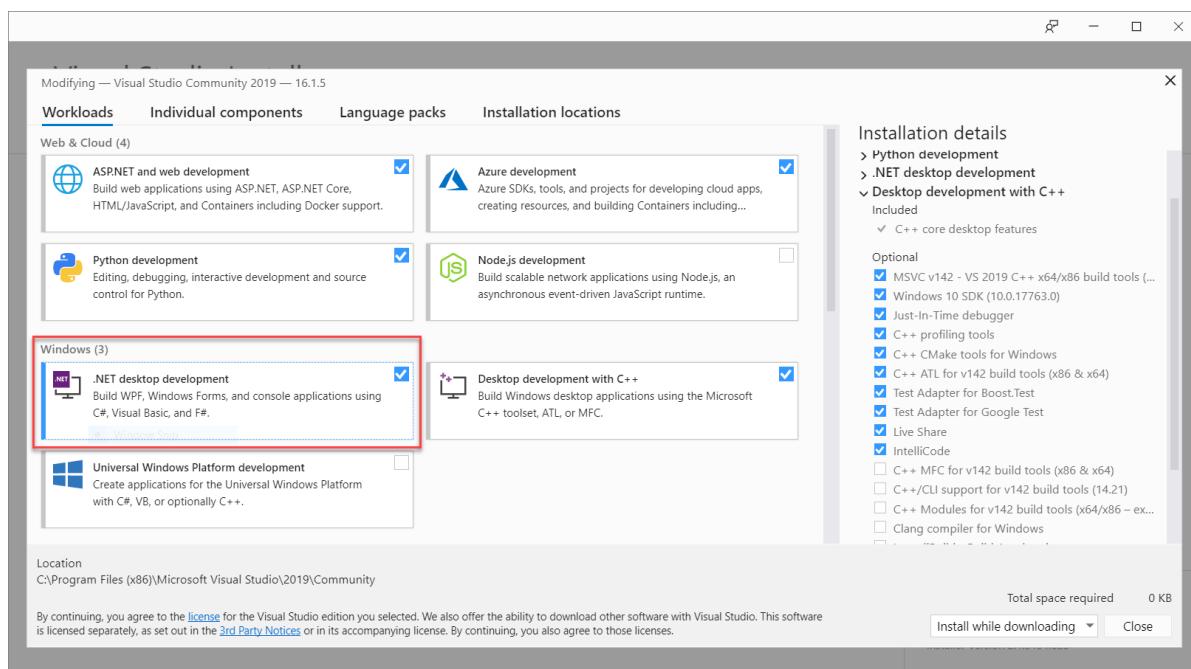
Para crear un proyecto de Visual Studio para el desarrollo de Windows, debe crear el proyecto, configurar Visual Studio para el desarrollo de escritorio de .NET, instalar el SDK de Voz y elegir la arquitectura de destino.

## Creación del proyecto e incorporación de la carga de trabajo

Para empezar, cree el proyecto en Visual Studio y asegúrese de que Visual Studio está configurado para el desarrollo de escritorio de .NET:

1. Abra Visual Studio 2019.
2. En la ventana Inicio, seleccione **Crear un proyecto**.
3. En la ventana **Crear un proyecto**, elija **Aplicación de consola (.NET Framework)** y seleccione **Siguiente**.
4. En la ventana **Configure su nuevo proyecto**, escriba *helloworld* en **Nombre del proyecto**, elija o cree la ruta de acceso del directorio en **Ubicación** y seleccione **Crear**.
5. En la barra de menús de Visual Studio, seleccione **Herramientas > Obtener herramientas y características**, que abre el Instalador de Visual Studio y muestra el cuadro de diálogo **Modificando**.
6. Compruebe si la carga de trabajo **Desarrollo de escritorio de .NET** está disponible. Si la carga de trabajo aún no se ha instalado, active la casilla que hay al lado y seleccione **Modificar** para iniciar la instalación. La descarga e instalación pueden tardar unos minutos.

Si la casilla que está junto a **Desarrollo de escritorio de .NET** ya está seleccionada, seleccione **Cerrar** para salir del cuadro de diálogo.

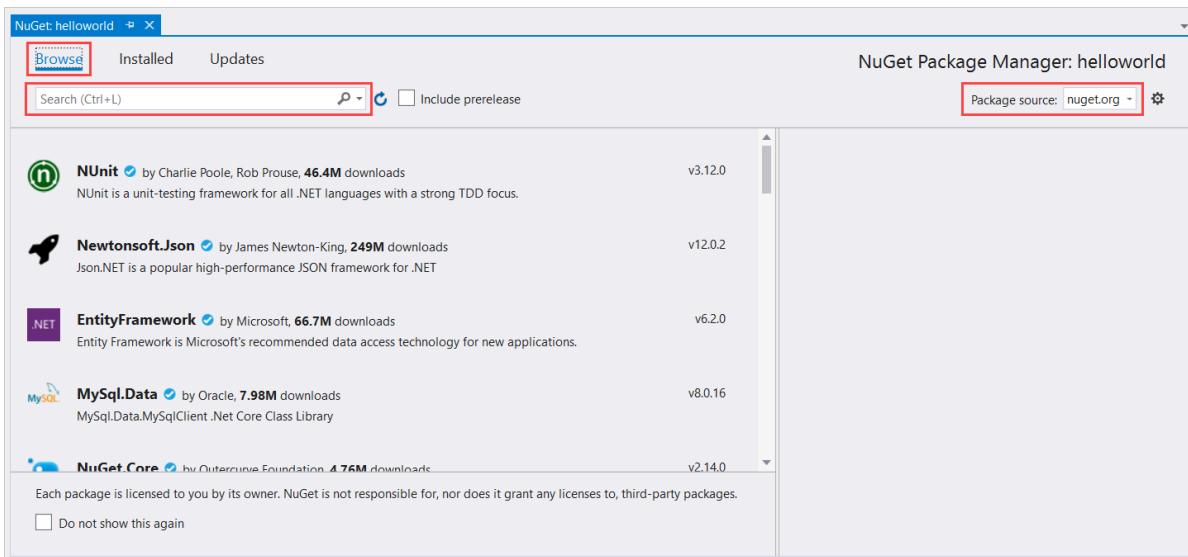


7. Cierre el Instalador de Visual Studio.

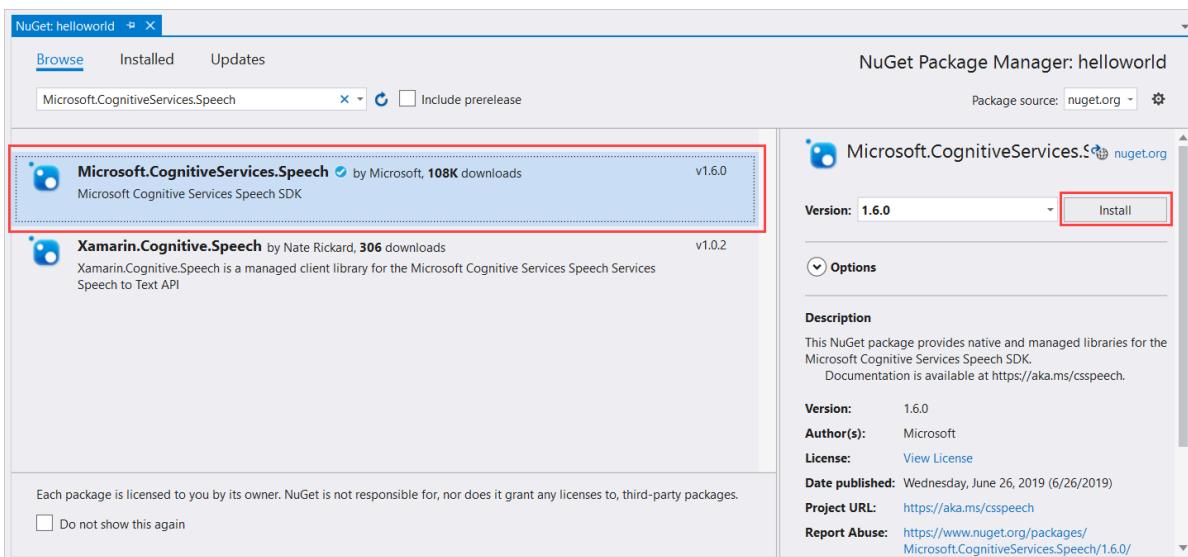
## Instalación de Speech SDK

El siguiente paso consiste en instalar el [paquete NuGet del SDK de Voz](#) para que pueda hacer referencia a él en el código.

1. En el Explorador de soluciones, haga clic con el botón derecho en el proyecto **helloworld** y seleccione **Administrar paquetes NuGet** para mostrar el Administrador de paquetes NuGet.



2. En la esquina superior derecha, busque el cuadro desplegable **Origen del paquete** y asegúrese de que **nuget.org** está seleccionado.
3. En la esquina superior izquierda, seleccione **Examinar**.
4. En el cuadro de búsqueda, escriba *Microsoft.CognitiveServices.Speech* y seleccione **Entrar**.
5. En los resultados de la búsqueda, seleccione el paquete **Microsoft.CognitiveServices.Speech** y, después, seleccione **Instalar** para instalar la versión estable más reciente.



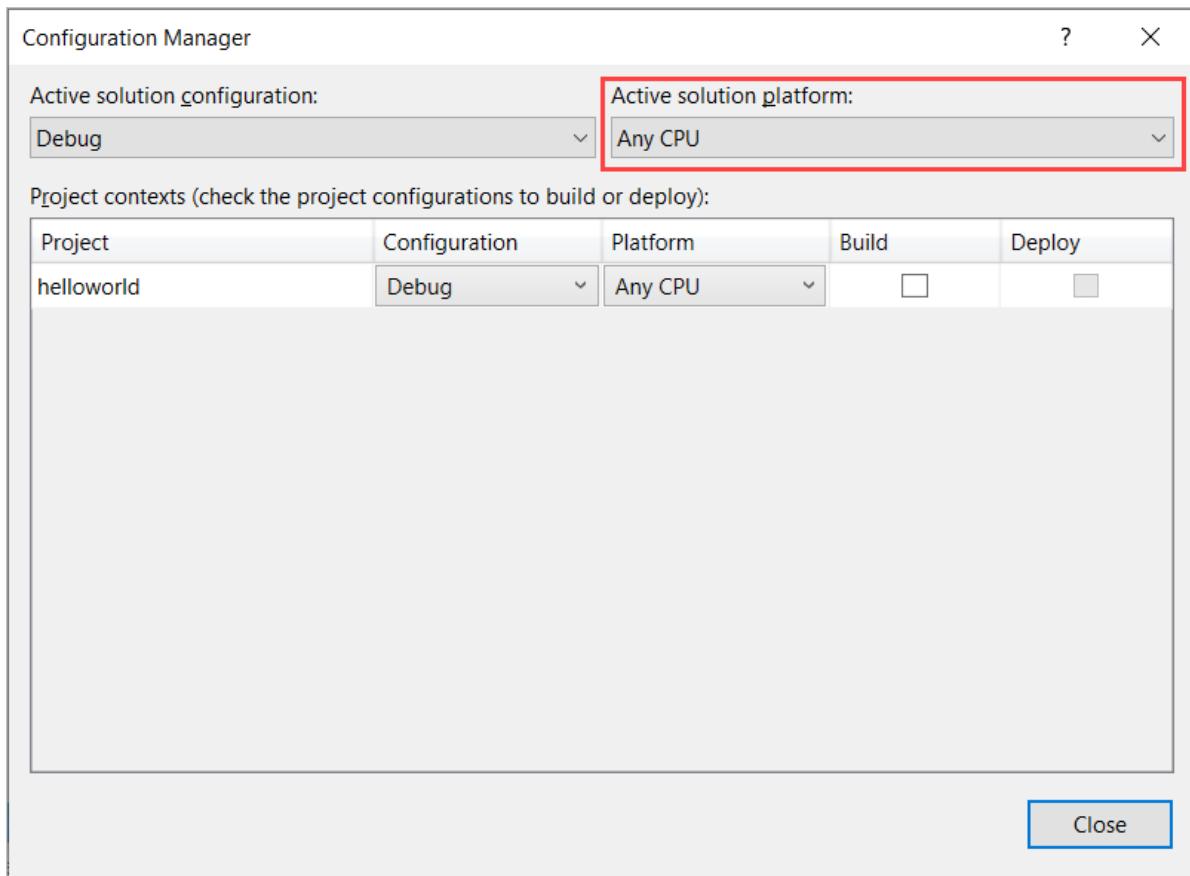
6. Acepte todos los contratos y licencias para iniciar la instalación.

Después de instalar el paquete aparecerá una confirmación en la ventana **Consola del administrador de paquetes**.

### Elección de la arquitectura de destino

Ahora, para compilar y ejecutar la aplicación de consola, cree una configuración de plataforma que coincida con la arquitectura del equipo.

1. En la barra de menús, seleccione **Compilar > Administrador de configuración**. Aparecerá el cuadro de diálogo **Administrador de configuración**.



2. En el cuadro desplegable **Active solution platform** (Plataforma de soluciones activas), seleccione **Nuevo**. Aparecerá el cuadro de diálogo **Nueva plataforma de solución**.
3. En el cuadro desplegable **escriba o seleccione la nueva plataforma**:
  - Si está ejecutando Windows de 64 bits, seleccione **x64**.
  - Si está ejecutando Windows de 32 bits, seleccione **x86**.
4. Seleccione **Aceptar** y, después, **Cerrar**.

## Adición del código

Después, agregue al código al proyecto.

1. En el **Explorador de soluciones**, abra el archivo **Program.cs**.
2. Reemplace el bloque de sentencias `using` del principio del archivo por las siguientes declaraciones:

```
using System;
using System.Threading.Tasks;
using Microsoft.CognitiveServices.Speech;
using Microsoft.CognitiveServices.Speech.Audio;
using Microsoft.CognitiveServices.Speech.Intent;
```

3. En el método `Main()` que se proporciona, agregue el código siguiente:

```
RecognizeIntentAsync().Wait();
Console.WriteLine("Please press Enter to continue.");
Console.ReadLine();
```

4. Cree un método asincrónico vacío `RecognizeIntentAsync()`, tal como se muestra aquí:

```
static async Task RecognizeIntentAsync()
{
}
```

5. En el cuerpo del nuevo método, agregue este código:

```

// Creates an instance of a speech config with specified subscription key
// and service region. Note that in contrast to other services supported by
// the Cognitive Services Speech SDK, the Language Understanding service
// requires a specific subscription key from https://www.luis.ai/.
// The Language Understanding service calls the required key 'endpoint key'.
// Once you've obtained it, replace with below with your own Language Understanding subscription key
// and service region (e.g., "westus").
// The default language is "en-us".
var config = SpeechConfig.FromSubscription("YourLanguageUnderstandingSubscriptionKey",
"YourLanguageUnderstandingServiceRegion");

// Creates an intent recognizer using microphone as audio input.
using (var recognizer = new IntentRecognizer(config))
{
    // Creates a Language Understanding model using the app id, and adds specific intents from your
    model
    var model = LanguageUnderstandingModel.FromAppId("YourLanguageUnderstandingAppId");
    recognizer.AddIntent(model, "YourLanguageUnderstandingIntentName1", "id1");
    recognizer.AddIntent(model, "YourLanguageUnderstandingIntentName2", "id2");
    recognizer.AddIntent(model, "YourLanguageUnderstandingIntentName3", "any-IntentId-here");

    // Starts recognizing.
    Console.WriteLine("Say something...");

    // Starts intent recognition, and returns after a single utterance is recognized. The end of a
    // single utterance is determined by listening for silence at the end or until a maximum of 15
    // seconds of audio is processed. The task returns the recognition text as result.
    // Note: Since RecognizeOnceAsync() returns only a single utterance, it is suitable only for single
    // shot recognition like command or query.
    // For long-running multi-utterance recognition, use StartContinuousRecognitionAsync() instead.
    var result = await recognizer.RecognizeOnceAsync().ConfigureAwait(false);

    // Checks result.
    if (result.Reason == ResultReason.RecognizedIntent)
    {
        Console.WriteLine($"RECOGNIZED: Text={result.Text}");
        Console.WriteLine($"    Intent Id: {result.IntentId}.");
        Console.WriteLine($"    Language Understanding JSON:");
        {result.Properties.GetProperty(PropertyId.LanguageUnderstandingServiceResponse_JsonResult)."};
    }
    else if (result.Reason == ResultReason.RecognizedSpeech)
    {
        Console.WriteLine($"RECOGNIZED: Text={result.Text}");
        Console.WriteLine($"    Intent not recognized.");
    }
    else if (result.Reason == ResultReason.NoMatch)
    {
        Console.WriteLine($"NOMATCH: Speech could not be recognized.");
    }
    else if (result.Reason == ResultReason.Canceled)
    {
        var cancellation = CancellationDetails.FromResult(result);
        Console.WriteLine($"CANCELED: Reason={cancellation.Reason}");

        if (cancellation.Reason == CancellationReason.Error)
        {
            Console.WriteLine($"CANCELED: ErrorCode={cancellation.ErrorCode}");
            Console.WriteLine($"CANCELED: ErrorDetails={cancellation.ErrorDetails}");
            Console.WriteLine($"CANCELED: Did you update the subscription info?");
        }
    }
}

```

6. Reemplace los marcadores de posición en este método por su clave de suscripción, región e identificador de aplicación de LUIS, como se indica a continuación.

MARCADOR DE POSICIÓN	REEMPLAZAR POR
YourLanguageUnderstandingSubscriptionKey	La clave de punto de conexión de LUIS. Una vez más, este elemento se debe obtener en el panel de Azure, no una "clave de inicio". Se puede encontrar en la página <b>Keys and Endpoints</b> (Claves y puntos de conexión), (en <b>Manage [Administrar]</b> ) en el <a href="#">portal de LUIS</a> .
YourLanguageUnderstandingServiceRegion	El identificador corto para la región en la que se encuentra la suscripción a LUIS, como <code>westus</code> para Oeste de EE. UU. Consulte <a href="#">Regiones</a> .
YourLanguageUnderstandingAppId	El id. de la aplicación LUIS. Se puede encontrar en la página <b>Settings</b> (Configuración) del <a href="#">portal de LUIS</a> .

Con estos cambios realizados, puede crear (**Control-Mayús-B**) y ejecutar (**F5**) la aplicación de la guía. Cuando se le solicite, pruebe a decir "Apagar las luces" al micrófono del equipo. La aplicación muestra el resultado en la ventana de la consola.

Las secciones siguientes incluyen una explicación del código.

## Creación de un reconocedor de intenciones

En primer lugar, es preciso crear una configuración de voz desde la región y la clave del punto de conexión de LUIS. Las configuraciones de voz se pueden utilizar para crear reconocedores para las distintas funcionalidades del SDK de Voz. La configuración de voz tiene varias formas de especificar la suscripción que se desea usar; aquí usamos `FromSubscription`, que toma la clave de suscripción y la región.

### NOTE

Utilice la clave y la región de su suscripción a LUIS, no de su suscripción al servicio de voz.

Después, cree un reconocedor de intenciones mediante `new IntentRecognizer(config)`. Dado que la configuración ya sabe la suscripción que hay que utilizar, no es preciso volver a especificar la clave de suscripción y el punto de conexión al crear el reconocedor.

## Importación de un modelo de LUIS y adición de intenciones

Ahora importe el modelo desde la aplicación LUIS mediante `LanguageUnderstandingModel.FromAppId()` y agregue las intenciones de LUIS que deseé reconocer mediante el método `AddIntent()` del reconocedor. Estos dos pasos mejoran la precisión del reconocimiento de voz al indicar palabras que el usuario probablemente utilizará en sus solicitudes. No es preciso agregar todas las intenciones de la aplicación, salvo que se necesite reconocer todas en la aplicación.

Para agregar intenciones, es preciso especificar tres argumentos: el modelo de LUIS (que se ha creado y se llama `model`), el nombre de la intención y un identificador de la intención. La diferencia entre el identificador y el nombre es como sigue.

ARGUMENTO DE <code>ADDINTENT()</code>	PROPOSITO
<code>intentName</code>	El nombre de la intención, tal como se define en la aplicación LUIS. Este valor debe coincidir exactamente con el nombre de la intención de LUIS.

ARGUMENTO DE <code>ADDINTENT()</code>	PROPÓSITO
<code>intentID</code>	Identificador asignado a una intención reconocida por el SDK de Voz. Este valor puede ser el que desee; no es preciso que se corresponda con el nombre de la intención definido en la aplicación LUIS. Si se controlan varias intenciones con el mismo código, por ejemplo, se puede utilizar el mismo identificador para ellos.

La aplicación de LUIS Home Automation tiene dos intenciones: una para encender un dispositivo y otra para apagar un dispositivo. En las líneas siguientes se agregan estas intenciones al reconocedor; reemplace las tres líneas `AddIntent` del método `RecognizeIntentAsync()` por este código.

```
recognizer.AddIntent(model, "HomeAutomation.TurnOff", "off");
recognizer.AddIntent(model, "HomeAutomation.TurnOn", "on");
```

En lugar de agregar intenciones individuales, puede usar el método `AddAllIntents` para agregar todas las intenciones de un modelo al reconocedor.

## Inicio del reconocimiento

Cuando haya creado el reconocedor y haya agregado las intenciones, puede empezar el reconocimiento. El SDK de Voz admite tanto el reconocimiento continuo como el de una sola emisión.

MODO DE RECONOCIMIENTO	MÉTODOS PARA LA LLAMADA	RESULTADO
Emisión única	<code>RecognizeOnceAsync()</code>	Devuelve la intención reconocida, si la hubiera, después de una expresión.
Continuo	<code>StartContinuousRecognitionAsync()</code> <code>StopContinuousRecognitionAsync()</code>	Reconoce varias expresiones; emite eventos (por ejemplo, <code>IntermediateResultReceived</code> ) cuando los resultados están disponibles.

La aplicación utiliza el modo de emisión única y, por lo tanto, llama a `RecognizeOnceAsync()` para iniciar el reconocimiento. El resultado es un objeto `IntentRecognitionResult` que contiene información sobre la intención reconocida. La respuesta JSON de LUIS se extrae mediante la siguiente expresión:

```
result.Properties.GetProperty(PropertyId.LanguageUnderstandingServiceResponse_JsonResult)
```

La aplicación no analiza el resultado de JSON. Solo muestra el texto JSON en la ventana de la consola.

```
Say something...
RECOGNIZED: Text=Hey turn off the light.
  Intent Id: off.
  Language Understanding JSON: {
    "query": "Hey turn off the light",
    "topScoringIntent": {
      "intent": "HomeAutomation.TurnOff",
      "score": 0.997960269
    },
    "entities": [
      {
        "entity": "light",
        "type": "HomeAutomation.DeviceType",
        "startIndex": 17,
        "endIndex": 21,
        "resolution": {
          "values": [
            "light"
          ]
        }
      }
    ]
}.
Please press Enter to continue.
```

## Especificación de un idioma de reconocimiento

De forma predeterminada, LUIS reconoce las intenciones en idioma inglés de Estados Unidos (`en-us`). Al asignar un código de configuración regional a la propiedad `SpeechRecognitionLanguage` de la configuración de voz, puede reconocer las intenciones en otros idiomas. Por ejemplo, agregue `config.SpeechRecognitionLanguage = "de-de";` en la aplicación antes de crear el reconocedor para reconocer las intenciones en idioma alemán. Para más información, consulte [Lenguajes admitidos](#).

## Reconocimiento continuo desde un archivo

El código siguiente muestra dos funcionalidades adicionales de reconocimiento de intenciones mediante el SDK de Voz. La primera, mencionada anteriormente, es el reconocimiento continuo, donde el reconocedor emite eventos cuando los resultados están disponibles. Estos eventos los pueden procesar los controladores de eventos que proporcione. Con el reconocimiento continuo, se llama al método `StartContinuousRecognitionAsync()` del reconocedor para iniciar el reconocimiento, en lugar de a `RecognizeOnceAsync()`.

La otra funcionalidad es leer el audio que contiene la voz que se va a procesar desde un archivo WAV. La implementación implica la creación de una configuración de audio que se puede utilizar al crear el reconocedor de intenciones. El archivo debe ser monocanal (mono) con una frecuencia de muestreo de 16 kHz.

Para probar estas características, elimine o marque como comentario el cuerpo del método `RecognizeIntentAsync()` y agregue el siguiente código en su lugar.

```
// Creates an instance of a speech config with specified subscription key
// and service region. Note that in contrast to other services supported by
// the Cognitive Services Speech SDK, the Language Understanding service
// requires a specific subscription key from https://www.luis.ai/.
// The Language Understanding service calls the required key 'endpoint key'.
// Once you've obtained it, replace with below with your own Language Understanding subscription key
// and service region (e.g., "westus").
var config = SpeechConfig.FromSubscription("YourLanguageUnderstandingSubscriptionKey",
"YourLanguageUnderstandingServiceRegion");

// Creates an intent recognizer using file as audio input.
// Replace with your own audio file name.
using (var audioInput = AudioConfig.FromWavFileInput("whatstheweatherlike.wav"))
{
```

```

using (var recognizer = new IntentRecognizer(config, audioInput))
{
    // The TaskCompletionSource to stop recognition.
    var stopRecognition = new TaskCompletionSource<int>();

    // Creates a Language Understanding model using the app id, and adds specific intents from your model
    var model = LanguageUnderstandingModel.FromAppId("YourLanguageUnderstandingAppId");
    recognizer.AddIntent(model, "YourLanguageUnderstandingIntentName1", "id1");
    recognizer.AddIntent(model, "YourLanguageUnderstandingIntentName2", "id2");
    recognizer.AddIntent(model, "YourLanguageUnderstandingIntentName3", "any-IntentId-here");

    // Subscribes to events.
    recognizer.Recognizing += (s, e) => {
        Console.WriteLine($"RECOGNIZING: Text={e.Result.Text}");
    };

    recognizer.Recognized += (s, e) => {
        if (e.Result.Reason == ResultReason.RecognizedIntent)
        {
            Console.WriteLine($"RECOGNIZED: Text={e.Result.Text}");
            Console.WriteLine($"    Intent Id: {e.Result.IntentId}.");
            Console.WriteLine($"    Language Understanding JSON:");
            {e.Result.Properties.GetProperty(PropertyId.LanguageUnderstandingServiceResponse_JsonResult).};
        }
        else if (e.Result.Reason == ResultReason.RecognizedSpeech)
        {
            Console.WriteLine($"RECOGNIZED: Text={e.Result.Text}");
            Console.WriteLine($"    Intent not recognized.");
        }
        else if (e.Result.Reason == ResultReason.NoMatch)
        {
            Console.WriteLine($"NOMATCH: Speech could not be recognized.");
        }
    };
};

recognizer.Canceled += (s, e) => {
    Console.WriteLine($"CANCELED: Reason={e.Reason}");

    if (e.Reason == CancellationReason.Error)
    {
        Console.WriteLine($"CANCELED: ErrorCode={e.ErrorCode}");
        Console.WriteLine($"CANCELED: ErrorDetails={e.ErrorDetails}");
        Console.WriteLine($"CANCELED: Did you update the subscription info?");
    }

    stopRecognition.TrySetResult(0);
};

recognizer.SessionStarted += (s, e) => {
    Console.WriteLine("\n    Session started event.");
};

recognizer.SessionStopped += (s, e) => {
    Console.WriteLine("\n    Session stopped event.");
    Console.WriteLine("\nStop recognition.");
    stopRecognition.TrySetResult(0);
};

// Starts continuous recognition. Uses StopContinuousRecognitionAsync() to stop recognition.
Console.WriteLine("Say something...");
await recognizer.StartContinuousRecognitionAsync().ConfigureAwait(false);

// Waits for completion.
// Use Task.WaitAny to keep the task rooted.
Task.WaitAny(new[] { stopRecognition.Task });

// Stops recognition.

```

```
        await recognizer.StopContinuousRecognitionAsync().ConfigureAwait(false);
```

```
}
```

Revise el código para incluir la clave del punto de conexión de LUIS, la región y el identificador de la aplicación y para agregar las intenciones de automatización de dispositivos del hogar, como antes. Cambie `whatstheweatherlike.wav` por el nombre del archivo de audio grabado. Luego realice la compilación, copie el archivo de audio en el directorio de compilación y ejecute la aplicación.

Por ejemplo, si dice "Apagar las luces", hace una pausa y, después, dice "Encender las luces" en el archivo de audio grabado, puede aparecer una salida en la consola similar a la siguiente:

```
Say something...

Session started event.
RECOGNIZING: Text=turn off
RECOGNIZING: Text=turn off the
RECOGNIZING: Text=turn off the light
RECOGNIZING: Text=turn off the lights
RECOGNIZED: Text=Turn off the lights.
    Intent Id: off.
    Language Understanding JSON: {
        "query": "turn off the lights",
        "topScoringIntent": {
            "intent": "HomeAutomation.TurnOff",
            "score": 0.997679055
        },
        "entities": [
            {
                "entity": "lights",
                "type": "HomeAutomation.DeviceType",
                "startIndex": 13,
                "endIndex": 18,
                "resolution": {
                    "values": [
                        "light"
                    ]
                }
            }
        ]
    }.
RECOGNIZING: Text=turn on
RECOGNIZING: Text=turn on the
RECOGNIZING: Text=turn on the light
RECOGNIZING: Text=turn on the lights
RECOGNIZED: Text=Turn on the lights.
    Intent Id: on.
    Language Understanding JSON: {
        "query": "turn on the lights",
        "topScoringIntent": {
            "intent": "HomeAutomation.TurnOn",
            "score": 0.987454832
        },
        "entities": [
            {
                "entity": "lights",
                "type": "HomeAutomation.DeviceType",
                "startIndex": 12,
                "endIndex": 17,
                "resolution": {
                    "values": [
                        "light"
                    ]
                }
            }
        ]
    }.
NOMATCH: Speech could not be recognized.
NOMATCH: Speech could not be recognized.
CANCELED: Reason=EndOfStream

Session stopped event.

Stop recognition.
Please press Enter to continue.
```

## Obtención de los ejemplos

Para ver los ejemplos más recientes, consulte el repositorio de GitHub de [ejemplos de código de Cognitive Services Speech SDK](#).

Busque el código de este artículo en la carpeta **samples/csharp/sharedcontent/console**.

## Pasos siguientes

[Inicio rápido: Reconocimiento de voz a través de un micrófono](#)

# Notas de la versión

15/01/2020 • 33 minutes to read • [Edit Online](#)

## SDK de Voz 1.8.0: Versión de noviembre de 2019

### Nuevas características

- Se ha agregado una API `FromHost()` para facilitar su uso con contenedores locales y nubes soberanas.
- Se ha agregado la detección automática de idioma de origen para el reconocimiento de voz (en Java y C++)
- Se ha agregado el objeto `SourceLanguageConfig` para el reconocimiento de voz, que se usa para especificar los idiomas de origen esperados (en Java y C++).
- Se ha agregado compatibilidad con `KeywordRecognizer` en Windows (UWP), Android e iOS mediante los paquetes de Nuget y Unity.
- Se ha agregado la API de Java de conversación remota para realizar la transcripción de conversaciones en lotes asíncronos.

### Cambios importantes

- Las funcionalidades de transcripción de conversaciones se han movido al espacio de nombres `Microsoft.CognitiveServices.Speech.Transcription`.
- Parte de los métodos de transcripción de conversaciones se han movido a la nueva clase `Conversation`.
- Compatibilidad eliminada para iOS de 32 bits (ARMv7 y x86)

### Correcciones de errores

- Se ha corregido un bloqueo si se usa `KeywordRecognizer` local sin una clave de suscripción válida al servicio de voz.

### Muestras

- Ejemplo de Xamarin para `KeywordRecognizer`
- Ejemplo de Unity para `KeywordRecognizer`
- Ejemplos de C++ y Java de detección automática de idioma de origen

## SDK de voz 1.7.0: versión de septiembre de 2019

### Nuevas características

- Compatibilidad con la versión beta agregada para Xamarin en la Plataforma universal de Windows (UWP), Android e iOS
- Compatibilidad con iOS agregada para Unity
- Se ha agregado compatibilidad con entradas `Compressed` para ALaw, Mulaw, FLAC en Android, iOS y Linux.
- Se ha agregado `SendMessageAsync` en la clase `Connection` para enviar un mensaje al servicio.
- Se ha agregado `SetMessageProperty` en la clase `Connection` para establecer la propiedad de un mensaje.
- TTS agregó compatibilidad para Java (JRE y Android), Python, Swift y Objective-C
- TTS agregó compatibilidad de reproducción para macOS, iOS y Android
- Se ha agregado información de "límite de palabras" para TTS

### Correcciones de errores

- Se ha corregido un problema de compilación de IL2CPP en Unity 2019 para Android

- Se ha corregido un problema con los encabezados con formato incorrecto en la entrada de archivo WAV que se procesa de forma incorrecta
- Se ha corregido un problema con UUID que no es único en algunas propiedades de conexión
- Se han corregido algunas advertencias sobre los especificadores de nulabilidad en los enlaces SWIFT (puede que se requieran pequeños cambios en el código)
- Se ha corregido un error que provocaba que las conexiones de WebSocket se cerraran de manera incorrecta en la carga de red
- Se ha corregido un problema en Android que a veces provoca que `DialogServiceConnector` use identificadores de impresión duplicados.
- Se han introducido mejoras en la estabilidad de las conexiones entre interacciones multiproceso y la generación de informes de errores (a través de eventos `Canceled`) cuando se producen con `DialogServiceConnector`.
- Los inicios de sesión de `DialogServiceConnector` ahora proporcionarán eventos correctamente, incluso si se llama a `ListenOnceAsync()` durante una operación `StartKeywordRecognitionAsync()` activa.
- Se ha resuelto un bloqueo asociado a la recepción de actividades `DialogServiceConnector`.

## Muestras

- Inicio rápido para Xamarin
- Se ha actualizado el inicio rápido de CPP con información de ARM64 de Linux
- Se ha actualizado el inicio rápido de Unity con información de iOS

# SDK de Voz 1.6.0: versión de junio de 2019

## Muestras

- Ejemplos de inicio rápido para Texto a voz en UWP y Unity
- Ejemplo de inicio rápido para Swift en iOS
- Ejemplos de Unity para Traducción y Reconocimiento de la intención comunicativa y Voz
- Ejemplos de inicios rápidos actualizados para `DialogServiceConnector`

## Mejoras y cambios

- Espacio de nombres de cuadro de diálogo:
  - El nombre de `SpeechBotConnector` ha cambiado a `DialogServiceConnector`
  - El nombre de `BotConfig` ha cambiado a `DialogServiceConfig`
  - `BotConfig::FromChannelSecret()` se ha reasignado a `DialogServiceConfig::FromBotSecret()`
  - Todos los clientes de Voz de Direct Line existentes siguen siendo compatibles después del cambio de nombre
- Actualización del adaptador REST de TTS para admitir una conexión persistente de proxy
- Un mejor mensaje de error cuando se pasa una región no válida
- Swift/Objective-C:
  - Mejores informes de errores: los métodos que pueden generar un error ahora se encuentran en dos versiones: una que expone un objeto `NSError` para el control de errores y una que genera una excepción. La primera se expone a Swift. Este cambio requiere adaptaciones en el código Swift existente.
  - Mejor control de eventos

## Correcciones de errores

- Corrección de TTS: donde el futuro de `SpeakTextAsync` se devolvió sin esperar al fin de la representación del audio
- Corrección para la serialización de las cadenas en C# para permitir la compatibilidad total con idiomas
- Corrección del problema de las aplicaciones centrales de .NET para cargar la biblioteca principal con un marco

de destino net461 en ejemplos

- Corrección de problemas ocasionales para implementar bibliotecas nativas en la carpeta de salida en los ejemplos
- Corrección para cerrar el socket web de manera confiable
- Corrección de un posible bloqueo al abrir una conexión con una carga muy elevada en Linux
- Corrección de metadatos faltantes en el paquete de marcos para macOS
- Corrección de problemas con `pip install --user` en Windows

## Speech SDK 1.5.1

Se trata de una versión de corrección de errores y solo afecta al SDK nativo o administrado. No afecta a la versión de JavaScript del SDK.

### Correcciones de errores

- Corrección de FromSubscription cuando se usa con la transcripción de la conversación.
- Corrección de errores en la detección de palabras clave para los asistentes por voz.

## Speech SDK 1.5.0 Versión de mayo de 2019

### Nuevas características:

- La detección de palabras clave (KWS) ahora está disponible para Windows y Linux. La funcionalidad KWS podría funcionar con cualquier tipo de micrófono; no obstante, la compatibilidad oficial de KWS está limitada actualmente a las matrices de micrófonos que se encuentran en el hardware de Azure Kinect DK o el SDK de dispositivos de voz.
- La funcionalidad de sugerencia de frases está disponible a través del SDK. Para más información, consulte [esta página](#).
- La funcionalidad de transcripción de conversaciones está disponible a través del SDK. Consulte [aquí](#).
- Compatibilidad agregada con los asistentes por voz mediante el canal Direct Line Speech.

### Muestras

- Se han agregado ejemplos para nuevas características o nuevos servicios admitidos por el SDK.

### Mejoras y cambios

- Se han agregado varias propiedades de reconocimiento para ajustar el comportamiento del servicio o los resultados del servicio (por ejemplo, enmascaramiento de palabras soeces etc.).
- Ahora puede configurar el reconocimiento a través de las propiedades de configuración estándar, incluso si ha creado el valor de `FromEndpoint` del reconocedor.
- Objective-C: se agregó la propiedad `OutputFormat` a `SPXSpeechConfiguration`.
- El SDK ahora admite Debian 9 como una distribución de Linux.

### Correcciones de errores

- Se ha corregido un problema donde el recurso de altavoz se destruía demasiado pronto en la conversión de texto a voz.

## Speech SDK 1.4.2

Se trata de una versión de corrección de errores y solo afecta al SDK nativo o administrado. No afecta a la versión de JavaScript del SDK.

## Speech SDK 1.4.1

Esta es una versión solo para JavaScript. No se agregó ninguna característica. Se realizaron las siguientes correcciones:

- Se impide que el paquete web cargue https-proxy-agent.

## Speech SDK 1.4.0 Versión de abril de 2019

### Nuevas características:

- El SDK admite ahora el servicio de conversión de texto a voz en versión beta. Se admite en Windows y Linux Desktop desde C++ y C#. Para más información, consulte la [información general sobre la conversión de texto a voz](#).
- El SDK ahora admite archivos de audio MP3 y Opus/OGG como archivos de entrada de secuencia. Esta característica solo está disponible en Linux desde C++ y C# y está actualmente en versión beta (más detalles [aquí](#)).
- Speech SDK para Java, .NET Core, C++ y Objective-C ha conseguido compatibilidad con macOS. La compatibilidad de Objective-C con macOS está actualmente en versión beta.
- iOS: Speech SDK para iOS (Objective-C) ahora también se publica como una instancia de CocoaPod.
- JavaScript: compatibilidad con micrófono no predeterminada como dispositivo de entrada.
- JavaScript: compatibilidad con servidores proxy para Node.js.

### Muestras

- se han agregado ejemplos para usar Speech SDK con C++ y con Objective-C en macOS.
- Se han agregado ejemplos que muestran el uso del servicio de conversión de texto a voz.

### Mejoras y cambios

- Python: ahora se exponen propiedades adicionales de los resultados del reconocimiento mediante la propiedad `properties`.
- Para la compatibilidad adicional con el desarrollo y la depuración, puede redirigir la información de registro y diagnóstico del SDK a un archivo de registro (más información [aquí](#)).
- JavaScript: mejora del rendimiento del procesamiento de audio.

### Correcciones de errores

- Mac/iOS: se corrigió un error que daba lugar a una larga espera cuando no se podía establecer una conexión con el servicio de voz.
- Python: mejora del control de errores en los argumentos de las devoluciones de llamada de Python.
- JavaScript: se corrigieron los informes de estado erróneos de la voz que finalizaban en RequestSession.

## Speech SDK 1.3.1 Actualización de febrero de 2019

Se trata de una versión de corrección de errores y solo afecta al SDK nativo o administrado. No afecta a la versión de JavaScript del SDK.

### Corrección de error

- Se ha corregido una fuga de memoria cuando se usa la entrada de micrófono. No afecta a la entrada de archivos o basada en secuencias.

## Speech SDK 1.3.0: versión de febrero de 2019

### Nuevas características

- El SDK de voz admite la selección del micrófono de entrada mediante la clase `AudioConfig`. Esto permite transmitir datos de audio al servicio de voz desde un micrófono no predeterminado. Para más información, consulte la documentación en la que se describe cómo [seleccionar un dispositivo de entrada de audio](#). Esta característica aún no está disponible en JavaScript.
- Speech SDK ahora es compatible con Unity en una versión beta. Proporcione sus comentarios en la sección de problemas en el [repositorio de ejemplos de GitHub](#). Esta versión es compatible con Unity en Windows x86 y x64 (aplicaciones de escritorio o de la Plataforma universal de Windows) y Android (ARM32/64, x86). Puede encontrar más información en nuestra [guía de inicio rápido sobre Unity](#).
- El archivo `Microsoft.CognitiveServices.Speech.csharp.bindings.dll` (incluido en versiones anteriores) ya no es necesario. La funcionalidad está ahora integrada en el SDK principal.

## Muestras

El siguiente contenido nuevo está disponible en nuestro [repositorio de ejemplo](#):

- Ejemplos adicionales para `AudioConfig.FromMicrophoneInput`.
- Ejemplos adicionales de Python para traducción y reconocimiento de intenciones.
- Ejemplos adicionales para usar el objeto `Connection` en iOS.
- Ejemplos adicionales de Java para la traducción con la salida de audio.
- Nuevo ejemplo de uso de la [API de REST de transcripción de lotes](#).

## Mejoras y cambios

- Python
  - Mensajes de error y verificación de parámetros mejorada en `SpeechConfig`.
  - Adición de compatibilidad para el objeto `Connection`.
  - Compatibilidad con Python (x86) de 32 bits en Windows.
  - Speech SDK para Python ya no está disponible como beta.
- iOS
  - El SDK ahora se compila en función de la versión 12.1 del SDK de iOS.
  - El SDK ahora es compatible con las versiones 9.2 y posteriores de iOS.
  - Documentación de referencia mejorada y corrección de varios nombres de propiedad.
- JavaScript
  - Adición de compatibilidad para el objeto `Connection`.
  - Archivos de definición de tipos agregados para JavaScript agrupado.
  - Compatibilidad e implementación iniciales para sugerencias de frases.
  - Colección de propiedades devuelta con JSON del servicio para reconocimiento.
- Los archivos DLL de Windows contienen ahora un recurso de versión.
- Si crea un valor de `FromEndpoint` de reconocedor, puede agregar parámetros directamente a la dirección URL del punto de conexión. Con `FromEndpoint` no puede configurar el reconocedor mediante las propiedades de configuración estándar.

## Correcciones de errores

- La contraseña de proxy y el nombre de usuario de proxy vacíos no se administraron correctamente. Con esta versión, si establece el nombre de usuario de proxy y la contraseña de proxy en una cadena vacía, no se enviarán al conectarse al proxy.
- El identificador de sesión creado por el SDK no siempre es realmente aleatorio para algunos lenguajes o entornos. Se ha agregado la inicialización del generador aleatorio para corregir este problema.
- Control mejorado del token de autorización. Si desea usar un token de autorización, especifíquelo en `SpeechConfig` y deje la clave de suscripción vacía. A continuación, cree el reconocedor como de costumbre.

- En algunos casos, el objeto `Connection` no se publicó correctamente. Ahora se ha corregido.
- Se corrigió el ejemplo de JavaScript para admitir la salida de audio para la síntesis de traducción también en Safari.

## Speech SDK 1.2.1

Esta es una versión solo para JavaScript. No se agregó ninguna característica. Se realizaron las siguientes correcciones:

- Activar el final del flujo en `turn.end`, y no en `speech.end`.
- Corregir error de la bomba de audio por el que no se programaba el siguiente envío en caso de error del envío actual.
- Corregir el reconocimiento continuo con el token de autenticación.
- Corrección de errores de diferentes reconocedores y puntos de conexión.
- Mejoras en la documentación.

## Speech SDK 1.2.0: Versión de diciembre de 2018

### Nuevas características

- Python
  - La versión beta de la compatibilidad con Python (3.5 y versiones posteriores) está disponible con esta versión. Para más información, consulte [aquí](#)(quickstart-python.md).
- JavaScript
  - Speech SDK para JavaScript ha sido de código abierto. El código fuente está disponible en [GitHub](#).
  - Ya se admite Node.js; puede encontrar más información [aquí](#).
  - Se quitó la restricción de longitud para las sesiones de audio; la reconexión se realizará automáticamente en la portada.
- Objeto `Connection`
  - Desde el objeto `Recognizer`, puede acceder al objeto `Connection`. Este objeto le permite iniciar la conexión al servicio y suscribirse para conectar y desconectar eventos explícitamente. (Esta característica no está disponible aún ni en JavaScript ni en Python).
- Compatibilidad con Ubuntu 18.04.
- Android
  - Compatibilidad con ProGuard habilitada durante la generación del APK.

### Mejoras

- Mejoras en el uso de subprocessos internos, lo que reduce el número de subprocessos, bloqueos y exclusiones mutuas.
- Se mejoraron los informes de errores y la información. En algunos casos, los mensajes de error no se propagan totalmente.
- Se actualizaron las dependencias de desarrollo en JavaScript para usar los módulos actualizados.

### Correcciones de errores

- Se han corregido las fugas de causadas por un error de coincidencia de tipos en `RecognizeAsync`.
- En algunos casos, se perdieron excepciones.
- Corrección de las fugas de memoria en los argumentos de eventos de traducción.
- Se ha corregido un problema de bloqueo al volver a conectar en sesiones de larga ejecución.
- Se ha corregido un problema que podría dar lugar a que faltase el resultado final para las traducciones con errores.

- C#: Si no se esperaba una operación `async` en el subprocesso principal, es posible que se pudiese desechar el reconocedor antes de completarse la tarea asíncrona.
- Java: Se ha corregido un problema que provocaba un bloqueo de la VM de Java.
- Objective-C: Se ha corregido la asignación de la enumeración; se devolvió `RecognizedIntent` en lugar de `RecognizingIntent`.
- JavaScript: Se ha establecido el formato de salida predeterminado en "simple" en `SpeechConfig`.
- JavaScript: Se ha quitado una incoherencia entre las propiedades del objeto de configuración en JavaScript y otros lenguajes.

## Muestras

- Se han actualizado y corregido varios ejemplos, como las voces de salida para la traducción, etc.
- Se han agregado ejemplos de Node.js en el [repositorio de ejemplo](#).

# Speech SDK 1.1.0

## Nuevas características

- Compatibilidad con Android x86/x64.
- Compatibilidad con proxy: En el objeto `SpeechConfig`, ahora puede llamar a una función para establecer la información del proxy (nombre de host, puerto, nombre de usuario y contraseña). Esta característica no está disponible aún en iOS.
- Mensajes y códigos de error mejorados. Si un reconocimiento devolvió un error, esto ya ha establecido `Reason` (en el evento cancelado) o `CancellationDetails` (en el resultado del reconocimiento) en `Error`. El evento cancelado ahora contiene dos miembros adicionales, `ErrorCode` y `ErrorDetails`. Si el servidor devolvió información de error adicional con el error notificado, ahora estará disponible en los nuevos miembros.

## Mejoras

- Verificación adicional agregada en la configuración del reconocedor y mensaje de error adicional agregado.
- Control mejorado del silencio prolongado en medio de un archivo de audio.
- Paquete NuGet: para proyectos de .NET Framework, evita la compilación con la configuración de AnyCPU.

## Correcciones de errores

- En los reconocedores se han encontrado varias excepciones corregidas. Además, las excepciones se detectan y se convierten en un evento `Canceled`.
- Corrección de una fuga de memoria en la administración de propiedades.
- Se corrigió el error en el que un archivo de entrada de audio podría bloquear el reconocedor.
- Se corrigió un error donde se podrían recibir eventos después de un evento de detención de la sesión.
- Se corrigieron algunas condiciones de subprocessos.
- Se corrigió un problema de compatibilidad de iOS que podría dar lugar a un bloqueo.
- Mejoras de estabilidad para la compatibilidad del micrófono en Android.
- Se corrigió un error donde un reconocedor en JavaScript ignoraría el lenguaje de reconocimiento.
- Se corrigió un error que impedía establecer el valor `EndpointId` (en algunos casos) en JavaScript.
- Se cambió el orden de los parámetros en `AddIntent` en JavaScript y se agregó la firma de JavaScript `AddIntent` que faltaba.

## Muestras

- Se han agregado ejemplos de C++ y C# para el uso de transmisiones de inserción y extracción en el [repositorio de ejemplos](#).

## Speech SDK 1.0.1

Mejoras en la confiabilidad y correcciones de errores:

- Corrección de un potencial error grave debido a una condición de carrera al desechar un reconocedor.
- Corrección de un potencial error grave en el caso de propiedades sin establecer.
- Comprobación adicional de errores y parámetros.
- Objective-C: corrección de posibles errores graves causados por la invalidación de nombres en NSString.
- Objective-C: ajuste de visibilidad en la API.
- JavaScript: corrección con respecto a los eventos y sus cargas.
- Mejoras en la documentación.

Se ha agregado un nuevo ejemplo de Javascript en nuestro [repositorio de ejemplos](#).

## SDK de Voz 1.0.0 de Cognitive Services: Versión de septiembre de 2018

### Nuevas características:

- Compatibilidad con Objective-C en iOS. Consulte la [Guía de inicio rápido de Objective-C para iOS](#).
- Se admite JavaScript en el explorador. Consulte la [Guía de inicio rápido de JavaScript](#).

### Cambios importantes

- Con esta versión se presentan una serie de cambios importantes. Consulte [esta página](#) para más información.

## SDK de Voz 0.6.0 de Cognitive Services: Versión de agosto de 2018

### Nuevas características:

- Ahora, las aplicaciones de UWP creadas con SDK de Voz superan el Kit para la certificación de aplicaciones en Windows (WACK). Consulte la [Guía de inicio rápido de UWP](#).
- Compatibilidad con .NET Standard 2.0 en Linux (Ubuntu 16.04 x64).
- Experimental: compatibilidad con Java 8 en Windows (64 bits) y Linux (Ubuntu 16.04 x 64). Consulte la [Guía de inicio rápido de Java Runtime Environment](#).

### Cambios funcionales

- Se expone más información detallada sobre los errores de conexión.

### Cambios importantes

- En Java (Android), la función `SpeechFactory.configureNativePlatformBindingWithDefaultCertificate` ya no requiere un parámetro de ruta de acceso. Ahora, la ruta de acceso se detecta automáticamente en todas las plataformas compatibles.
- En Java y C#, se ha quitado el descriptor de acceso get- de la propiedad `EndpointUrl`.

### Correcciones de errores

- En Java, se implementa ahora el resultado de la síntesis de audio en el reconocedor de traducción.
- Se ha corregido un error que podía provocar subprocessos inactivos y un mayor número de sockets abiertos y sin usar.
- Se ha corregido un problema por el que un proceso de reconocimiento de larga ejecución podía terminar en mitad de la transmisión.
- Se ha corregido una condición de carrera en el proceso de apagado del reconocedor.

# SDK de Voz 0.5.0 de Cognitive Services: Versión de julio de 2018

## Nuevas características:

- Compatibilidad con la plataforma Android (API 23: Android Marshmallow 6.0 o posterior). Consulte el [inicio rápido de Android](#).
- Compatibilidad con .NET Standard 2.0 en Windows. Consulte el [inicio rápido de .NET Core](#).
- Experimental: compatibilidad con UWP en Windows (versión 1709 o posterior).
  - Consulte la [Guía de inicio rápido de UWP](#).
  - Nota: Las aplicaciones de UWP creadas con el SDK de Voz no pasan aún el Kit para la certificación de aplicaciones en Windows (WACK).
- Compatibilidad con el reconocimiento de ejecución prolongada con reconexión automática.

## Cambios funcionales

- `StartContinuousRecognitionAsync()` admite reconocimiento de ejecución prolongada.
- El resultado del reconocimiento contiene más campos. Tienen un desplazamiento desde el principio del audio y la duración (ambos en tics) del texto reconocido y valores adicionales que representan el estado de reconocimiento, por ejemplo, `InitialSilenceTimeout` e `InitialBabbleTimeout`.
- Compatibilidad con `AuthorizationToken` para la creación de instancias de fábrica.

## Cambios importantes

- Eventos de reconocimiento: el tipo de evento `NoMatch` se combina con el evento `Error`.
- `SpeechOutputFormat` en C# se llama ahora `OutputFormat` para concordar con C++.
- El tipo de valor devuelto de algunos métodos de la interfaz `AudioInputStream` se ha modificado ligeramente:
  - En Java, el método `read` ahora devuelve `long` en lugar de `int`.
  - En C#, el método `Read` ahora devuelve `uint` en lugar de `int`.
  - En C++, los métodos `Read` y `GetFormat` ahora devuelven `size_t` en lugar de `int`.
- C++: las instancias de secuencias de entrada de audio ahora solo se pueden pasar como un valor `shared_ptr`.

## Correcciones de errores

- Se han corregido los valores devueltos incorrectos cuando se agota el tiempo de espera de `RecognizeAsync()`.
- Se ha eliminado la dependencia de las bibliotecas de Media Foundation en Windows. El SDK ahora usa las API de audio básicas.
- Corrección de la documentación: se ha agregado una página de [regiones](#) para describir cuáles son las regiones admitidas.

## Problema conocido

- SDK de Voz para Android no informa de los resultados de la síntesis de voz para la traducción. Este problema se solucionará en la próxima versión.

# SDK de Voz 0.4.0 de Cognitive Services: Versión de junio de 2018

## Cambios funcionales

- `AudioInputStream`

Un reconocedor ahora puede consumir una secuencia como origen de audio. Para más información, consulte la [guía de procedimientos](#) relacionada.

- Formato de salida detallado

Al crear un elemento `SpeechRecognizer`, puede solicitar el formato de salida `Detailed` o `Simple`.

`DetailedSpeechRecognitionResult` contiene una puntuación de confianza, texto reconocido, formato léxico sin formato, formato normalizado y formato normalizado con palabras soeces enmascaradas.

## Cambio importante

- En C# se cambia de `SpeechRecognitionResult.RecognizedText` a `SpeechRecognitionResult.Text`.

## Correcciones de errores

- Se ha corregido un posible problema de devolución de llamada en la capa USP durante el apagado.
- Si un reconocedor usaba un archivo de entrada de audio, mantenía el identificador de archivo más tiempo del necesario.
- Se han eliminado varios interbloqueos entre el suministro de mensajes y el reconocedor.
- Se desencadena un resultado `NoMatch` cuando se agota la respuesta del servicio.
- Las bibliotecas de Media Foundation en Windows son de carga retrasada. Esta biblioteca solo es necesaria para la entrada del micrófono.
- La velocidad de carga de los datos de audio se limita al doble de la velocidad de audio original.
- En Windows, los ensamblados .NET de C# ahora son de nombre seguro.
- Corrección de la documentación: `Region` necesita información para crear un reconocedor.

Se han agregado más ejemplos y se actualizan constantemente. Para obtener el conjunto más reciente de ejemplos, consulte el [repositorio de GitHub de ejemplos de SDK de Voz](#).

## SDK de Voz 0.2.12733 de Cognitive Services: Versión de mayo de 2018

Esta versión es la primera versión preliminar pública de SDK de Voz de Cognitive Services.

# ¿Qué es la traducción de voz?

13/01/2020 • 4 minutes to read • [Edit Online](#)

La traducción de voz del servicio de voz permite la traducción de voz a voz y voz a texto de secuencias de audio en varios idiomas en tiempo real. Con el SDK de voz, sus aplicaciones, herramientas y los dispositivos tienen acceso a las transcripciones de origen y a las salidas de traducción del audio proporcionadas. Se devuelven resultados provisionales de transcripción y traducción cuando se detecta la voz y los resultados finales se pueden convertir en voz sintetizada.

El motor de traducción de Microsoft usa tecnología de dos enfoques diferentes: traducción automática estadística (SMT) y traducción automática neuronal (NMT). SMT usa análisis estadísticos avanzados para estimar las mejores traducciones posibles dado el contexto de unas pocas palabras. Con NMT, las redes neuronales se utilizan para proporcionar traducciones más precisas y naturales mediante el contexto completo de las oraciones para traducir palabras.

En la actualidad, Microsoft utiliza NMT para la traducción a los idiomas más populares. Todos los [idiomas disponibles para la traducción de voz a voz](#) cuentan con la tecnología de NMT. La traducción de voz a texto puede utilizar SMT o NMT, según el par de idiomas. Si el idioma de destino admite NMT, la traducción completa se realiza con NMT. Si el idioma de destino no admite NMT, la traducción es un híbrido de NMT y SMT, con el idioma inglés como "enlace" entre los dos idiomas.

## Características principales

Estas son las características disponibles en el SDK de voz y las API REST de Speech Services:

CASO DE USO	SDK	REST
Traducción de voz a texto con resultados de reconocimiento.	Sí	Sin
Traducción de voz a voz.	Sí	Sin
Resultados de reconocimiento y traducción provisionales.	Sí	Sin

## Introducción a la traducción de voz

Le ofrecemos inicios rápidos diseñados para que ejecute el código en menos de 10 minutos. Esta tabla incluye una lista de inicios rápidos de traducción de voz ordenados por idioma.

GUÍA DE INICIO RÁPIDO	PLATAFORMA	REFERENCIA DE API
<a href="#">C#, .NET Core</a>	Windows	<a href="#">Browse</a>
<a href="#">C#, .NET Framework</a>	Windows	<a href="#">Browse</a>
<a href="#">C#, UWP</a>	Windows	<a href="#">Browse</a>
<a href="#">C++</a>	Windows	<a href="#">Browse</a>

GUÍA DE INICIO RÁPIDO	PLATAFORMA	REFERENCIA DE API
Java	Windows, Linux, macOS	<a href="#">Browse</a>

## Código de ejemplo

Hay un ejemplo de código para el SDK de voz disponible en GitHub. En estos ejemplos se tratan escenarios comunes como la lectura de audio de un archivo o secuencia, el reconocimiento o traducción continuos y de una sola emisión, y el trabajo con modelos personalizados.

- [Ejemplos de conversión de voz a texto y de traducción \(SDK\)](#)

## Guías de migración

Si sus aplicaciones, herramientas o productos usan [Translator Speech API](#), hemos creado guías que le ayudarán a migrar al servicio de voz.

- [Migración de Translator Speech API al servicio de voz](#)

## Documentos de referencia

- [Acerca del SDK de Voz](#)
- [Speech Devices SDK](#)
- [API REST: Speech-to-text](#) (API de REST: Voz a texto)
- [API REST: Text-to-speech](#) (API de REST: Texto a voz)
- [API REST: Batch transcription and customization](#) (API de REST: Transcripción y personalización de Azure Batch)

## Pasos siguientes

- [Obtenga una clave de suscripción gratuita a los servicios de Voz](#)
- [Obtención del SDK de voz](#)

# Inicio rápido: Traducción de voz a texto

15/01/2020 • 21 minutes to read • [Edit Online](#)

En este inicio rápido, va a usar el SDK de [Voz](#) para traducir interactivamente la voz de un idioma a texto en otro idioma. Una vez que se cumplen los requisitos previos, para la traducción de voz a texto solo son necesarios cinco pasos:

- Cree un objeto `SpeechConfig` a partir de la clave y la región de suscripción.
- Actualice el objeto `SpeechConfig` para especificar los idiomas de origen y destino.
- Cree un objeto `TranslationRecognizer` con el objeto `SpeechConfig` anterior.
- Con el objeto `TranslationRecognizer`, inicie el proceso de reconocimiento de una única expresión.
- Inspeccione el objeto `TranslationRecognitionResult` devuelto.

Si prefiere ponerse a trabajar de inmediato, vea o descargue todos los [ejemplos para C# del SDK de Voz](#) en GitHub. Si no, vamos a comenzar.

## Selección del entorno de destino

- [.NET](#)
- [.NET Core](#)

## Prerequisites

Antes de comenzar, compruebe lo siguiente:

- [Ha creado un recurso de Voz de Azure](#)
- [Ha configurado el entorno de desarrollo](#)
- [Ha creado un proyecto de ejemplo vacío](#)

## Incorporación de código de ejemplo

1. Abra el archivo **Program.cs** y sustituya todo el código existente por el siguiente.

```
using System;
using System.Threading.Tasks;
using Microsoft.CognitiveServices.Speech;
using Microsoft.CognitiveServices.Speech.Translation;

namespace helloworld
{
    class Program
    {
        public static async Task TranslateSpeechToText()
        {
            // Creates an instance of a speech translation config with specified subscription key
            // and service region.
            // Replace with your own subscription key and service region (e.g., "westus").
            var config = SpeechTranslationConfig.FromSubscription("YourSubscriptionKey",
            "YourServiceRegion");

            // Sets source and target languages.
            // Replace with the languages of your choice, from list found here:
            https://aka.ms/speech/sttt-languages
            string fromLanguage = "en-US";
            string toLanguage = "de";
```

```

config.SpeechRecognitionLanguage = fromLanguage;
config.AddTargetLanguage(toLanguage);

// Creates a translation recognizer using the default microphone audio input device.
using (var recognizer = new TranslationRecognizer(config))
{
    // Starts translation, and returns after a single utterance is recognized. The end
    of a
    // single utterance is determined by listening for silence at the end or until a
    maximum of 15
    // seconds of audio is processed. The task returns the recognized text as well as
    the translation.
    // Note: Since RecognizeOnceAsync() returns only a single utterance, it is suitable
    only for single
    // shot recognition like command or query.
    // For long-running multi-utterance recognition, use
    StartContinuousRecognitionAsync() instead.
    Console.WriteLine("Say something...");
    var result = await recognizer.RecognizeOnceAsync();

    // Checks result.
    if (result.Reason == ResultReason.TranslatedSpeech)
    {
        Console.WriteLine($"RECOGNIZED '{fromLanguage}': {result.Text}");
        Console.WriteLine($"TRANSLATED into '{toLanguage}'":
{result.Translations[toLanguage]}");
    }
    else if (result.Reason == ResultReason.RecognizedSpeech)
    {
        Console.WriteLine($"RECOGNIZED '{fromLanguage}': {result.Text} (text could not
be translated)");
    }
    else if (result.Reason == ResultReason.NoMatch)
    {
        Console.WriteLine($"NOMATCH: Speech could not be recognized.");
    }
    else if (result.Reason == ResultReason.Canceled)
    {
        var cancellation = CancellationDetails.FromResult(result);
        Console.WriteLine($"CANCELED: Reason={cancellation.Reason}");

        if (cancellation.Reason == CancellationReason.Error)
        {
            Console.WriteLine($"CANCELED: ErrorCode={cancellation.ErrorCode}");
            Console.WriteLine($"CANCELED: ErrorDetails={cancellation.ErrorDetails}");
            Console.WriteLine($"CANCELED: Did you update the subscription info?");
        }
    }
}

static void Main(string[] args)
{
    TranslateSpeechToText().Wait();
}
}
}

```

2. En el mismo archivo, reemplace la cadena `YourSubscriptionKey` por la clave de suscripción.
3. Reemplace la cadena `YourServiceRegion` por la [región](#) asociada a sus suscripción (por ejemplo, `westus` para la suscripción de evaluación gratuita).
4. En la barra de menús, elija **Archivo > Guardar todo**.

## Compilación y ejecución de la aplicación

1. En la barra de menús, seleccione **Compilar** > **Compilar solución** para compilar la aplicación. El código se debería compilar sin errores ahora.
2. Elija **Depurar** > **Iniciar depuración** o presione **F5** para iniciar la aplicación **HelloWorld**.
3. Diga una oración o frase en inglés. La aplicación transmite la voz al servicio de voz, que la traduce (alemán en este caso) y la transcribe en texto. Después, el servicio de voz envía el texto de nuevo a la aplicación para mostrarlo.

```
Say something...
RECOGNIZED 'en-US': What's the weather in Seattle?
TRANSLATED into 'de': Wie ist das Wetter in Seattle?
```

## Pasos siguientes

### Exploración de ejemplos de C# en GitHub

En este inicio rápido, va a usar el SDK de Voz para traducir interactivamente la voz de un idioma a texto en otro idioma. Una vez que se cumplen los requisitos previos, para la traducción de voz a texto solo son necesarios cinco pasos:

- Cree un objeto `SpeechConfig` a partir de la clave y la región de suscripción.
- Actualice el objeto `SpeechConfig` para especificar los idiomas de origen y destino.
- Cree un objeto `TranslationRecognizer` con el objeto `SpeechConfig` anterior.
- Con el objeto `TranslationRecognizer`, inicie el proceso de reconocimiento de una única expresión.
- Inspeccione el objeto `TranslationRecognitionResult` devuelto.

Si prefiere ponerse a trabajar de inmediato, vea o descargue todos los [ejemplos para C++ del SDK de Voz](#) en GitHub. Si no, vamos a comenzar.

## Prerequisites

Antes de comenzar, compruebe lo siguiente:

- [Ha creado un recurso de Voz de Azure](#)
- [Ha configurado el entorno de desarrollo](#)
- [Ha creado un proyecto de ejemplo vacío](#)

## Incorporación de código de ejemplo

1. Abra el archivo de origen **helloworld.cpp**.
2. Reemplace todo el código por el fragmento siguiente:

```
#include <iostream>
#include <vector>
#include <speechapi_cxx.h>

using namespace std;
using namespace Microsoft::CognitiveServices::Speech;
using namespace Microsoft::CognitiveServices::Speech::Translation;

void TranslateSpeechToText()
{
    // Creates an instance of a speech translation config with specified subscription key and
    // service region.
    // Replace with your own subscription key and service region (e.g., "westus").
```

```

        auto config = SpeechTranslationConfig::FromSubscription("YourSubscriptionKey",
"YourServiceRegion");

        // Sets source and target languages.
        // Replace with the languages of your choice, from list found here: https://aka.ms/speech/sttt-
languages
        auto fromLanguage = "en-US";
        auto toLanguage = "de";
        config->SetSpeechRecognitionLanguage(fromLanguage);
        config->AddTargetLanguage(toLanguage);

        // Creates a translation recognizer using the default microphone audio input device.
        auto recognizer = TranslationRecognizer::FromConfig(config);

        // Starts translation, and returns after a single utterance is recognized. The end of a
        // single utterance is determined by listening for silence at the end or until a maximum of 15
        // seconds of audio is processed. The task returns the recognized text as well as the
        translation.
        // Note: Since RecognizeOnceAsync() returns only a single utterance, it is suitable only for
single
        // shot recognition like command or query.
        // For long-running multi-utterance recognition, use StartContinuousRecognitionAsync() instead.
        cout << "Say something...\n";
        auto result = recognizer->RecognizeOnceAsync().get();

        // Checks result.
        if (result->Reason == ResultReason::TranslatedSpeech)
        {
            cout << "RECOGNIZED '" << fromLanguage << "'": " << result->Text << std::endl;
            cout << "TRANSLATED into '" << toLanguage << "'": " << result->Translations.at(toLanguage) <<
std::endl;
        }
        else if (result->Reason == ResultReason::RecognizedSpeech)
        {
            cout << "RECOGNIZED '" << fromLanguage << "' " << result->Text << " (text could not be
translated)" << std::endl;
        }
        else if (result->Reason == ResultReason::NoMatch)
        {
            cout << "NOMATCH: Speech could not be recognized." << std::endl;
        }
        else if (result->Reason == ResultReason::Canceled)
        {
            auto cancellation = CancellationDetails::FromResult(result);
            cout << "CANCELED: Reason=" << (int)cancellation->Reason << std::endl;

            if (cancellation->Reason == CancellationReason::Error)
            {
                cout << "CANCELED: ErrorCode=" << (int)cancellation->ErrorCode << std::endl;
                cout << "CANCELED: ErrorDetails=" << cancellation->ErrorDetails << std::endl;
                cout << "CANCELED: Did you update the subscription info?" << std::endl;
            }
        }
    }

    int wmain()
    {
        TranslateSpeechToText();
        return 0;
    }
}

```

3. En el mismo archivo, reemplace la cadena `YourSubscriptionKey` por la clave de suscripción.

4. Reemplace la cadena `YourServiceRegion` por la [región](#) asociada a sus suscripción (por ejemplo, `westus` para la suscripción de evaluación gratuita).

5. En la barra de menús, elija **Archivo > Guardar todo**.

# Compilación y ejecución de la aplicación

1. En la barra de menús, seleccione **Compilar > Compilar solución** para compilar la aplicación. El código se debería compilar sin errores ahora.
2. Elija **Depurar > Iniciar depuración** o presione **F5** para iniciar la aplicación **HelloWorld**.
3. Diga una oración o frase en inglés. La aplicación transmite la voz al servicio de voz, que la traduce (alemán en este caso) y la transcribe en texto. Después, el servicio de voz envía el texto de nuevo a la aplicación para mostrarlo.

```
Say something...
RECOGNIZED 'en-US': What's the weather in Seattle?
TRANSLATED into 'de': Wie ist das Wetter in Seattle?
```

## Pasos siguientes

### [Exploración de ejemplos de C++ en GitHub](#)

En este inicio rápido, va a usar el SDK de [Voz](#) para traducir interactivamente la voz de un idioma a texto en otro idioma. Una vez que se cumplen los requisitos previos, para la traducción de voz a texto solo son necesarios cinco pasos:

- Cree un objeto `SpeechConfig` a partir de la clave y la región de suscripción.
- Actualice el objeto `SpeechConfig` para especificar los idiomas de origen y destino.
- Cree un objeto `TranslationRecognizer` con el objeto `SpeechConfig` anterior.
- Con el objeto `TranslationRecognizer`, inicie el proceso de reconocimiento de una única expresión.
- Inspeccione el objeto `TranslationRecognitionResult` devuelto.

Si prefiere ponerse a trabajar de inmediato, vea o descargue todos los [ejemplos para Java del SDK de Voz](#) en GitHub. Si no, vamos a comenzar.

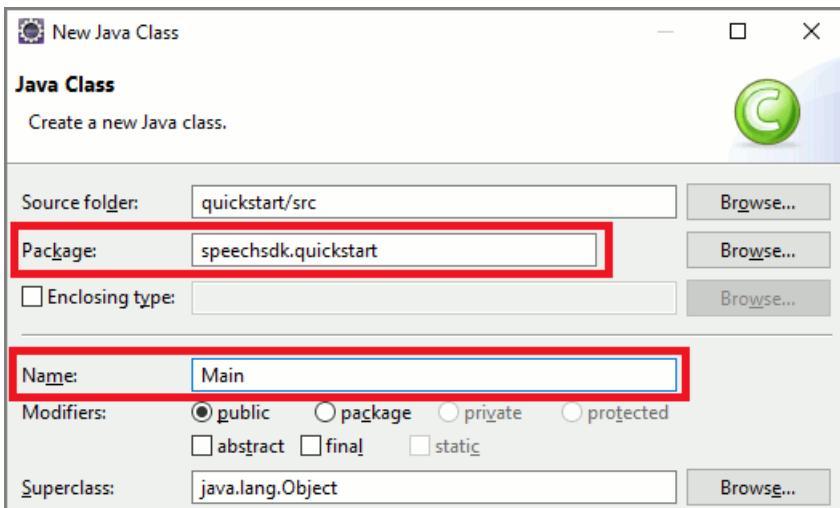
## Prerequisites

Antes de comenzar, compruebe lo siguiente:

- [Ha creado un recurso de Voz de Azure](#)
- [Ha configurado el entorno de desarrollo](#)
- [Ha creado un proyecto de ejemplo vacío](#)

## Incorporación de código de ejemplo

1. Para agregar una nueva clase vacía al proyecto de Java, seleccione **File (Archivo) > New (Nuevo) > Class (Clase)**.
2. En la ventana **New Java Class** (Nueva clase de Java) escriba **speechsdk.quickstart** en el campo **Package** (Paquete) y **Main** en el campo **Name** (Nombre).



3. Reemplace el código en `Main.java` con el siguiente fragmento de código:

```
package quickstart;

import java.io.IOException;
import java.util.concurrent.Future;
import java.util.concurrent.ExecutionException;
import com.microsoft.cognitiveservices.speech.*;
import com.microsoft.cognitiveservices.speech.translation.*;

public class Main {

    public static void translationWithMicrophoneAsync() throws InterruptedException,
ExecutionException, IOException
    {
        // Creates an instance of a speech translation config with specified
        // subscription key and service region. Replace with your own subscription key
        // and service region (e.g., "westus").

        int exitCode = 1;
        SpeechTranslationConfig config =
SpeechTranslationConfig.fromSubscription("YourSubscriptionKey", "YourServiceRegion");
        assert(config != null);

        // Sets source and target languages.
        String fromLanguage = "en-US";
        String toLanguage = "de";
        config.setSpeechRecognitionLanguage(fromLanguage);
        config.addTargetLanguage(toLanguage);

        // Creates a translation recognizer using the default microphone audio input device.
        TranslationRecognizer recognizer = new TranslationRecognizer(config);
        assert(recognizer != null);

        System.out.println("Say something...");

        // Starts translation, and returns after a single utterance is recognized. The end of a
        // single utterance is determined by listening for silence at the end or until a maximum of
15
        // seconds of audio is processed. The task returns the recognized text as well as the
        // translation.
        // Note: Since recognizeOnceAsync() returns only a single utterance, it is suitable only for
single
        // shot recognition like command or query.
        // For long-running multi-utterance recognition, use startContinuousRecognitionAsync()
instead.
        Future<TranslationRecognitionResult> task = recognizer.recognizeOnceAsync();
        assert(task != null);

        TranslationRecognitionResult result = task.get();
```

```

        assert(result != null);

        if (result.getReason() == ResultReason.TranslatedSpeech) {
            System.out.println("RECOGNIZED '" + fromLanguage + "': " + result.getText());
            System.out.println("TRANSLATED into '" + toLanguage + "': " +
result.getTranslations().get(toLanguage));
            exitCode = 0;
        }
        else if (result.getReason() == ResultReason.RecognizedSpeech) {
            System.out.println("RECOGNIZED '" + fromLanguage + "': " + result.getText() + "(text
could not be translated)");
            exitCode = 0;
        }
        else if (result.getReason() == ResultReason.NoMatch) {
            System.out.println("NOMATCH: Speech could not be recognized.");
        }
        else if (result.getReason() == ResultReason.Canceled) {
            CancellationDetails cancellation = CancellationDetails.fromResult(result);
            System.out.println("CANCELED: Reason=" + cancellation.getReason());

            if (cancellation.getReason() == CancellationReason.Error) {
                System.out.println("CANCELED: ErrorCode=" + cancellation.getErrorCode());
                System.out.println("CANCELED: ErrorDetails=" + cancellation.getErrorDetails());
                System.out.println("CANCELED: Did you update the subscription info?");
            }
        }
    }

    recognizer.close();

    System.exit(exitCode);
}

public static void main(String[] args) {
    try {
        translationWithMicrophoneAsync();
    } catch (Exception ex) {
        System.out.println("Unexpected exception: " + ex.getMessage());
        assert(false);
        System.exit(1);
    }
}
}

```

4. Reemplace la cadena `YourSubscriptionKey` por la clave de suscripción.

5. Reemplace la cadena `YourServiceRegion` por la [región](#) asociada a sus suscripción (por ejemplo, `westus` para la suscripción de evaluación gratuita).

6. Guarde los cambios en el proyecto.

## Compilación y ejecución de la aplicación

Presione F11, o seleccione **Run (Ejecutar) > Debug (Depurar)**.

- Diga una oración o frase en inglés. La aplicación transmite la voz al servicio de voz, que la traduce (a alemán en este caso) y la transcribe en texto. Después, el servicio de voz envía el texto de nuevo a la aplicación para mostrarlo.

```

Say something...
RECOGNIZED 'en-US': What's the weather in Seattle?
TRANSLATED into 'de': Wie ist das Wetter in Seattle?

```

# Pasos siguientes

## Exploración de ejemplos de Java en GitHub

En este inicio rápido, va a usar el SDK de [Voz](#) para traducir interactivamente la voz de un idioma a texto en otro idioma. Una vez que se cumplen los requisitos previos, para la traducción de voz a texto solo son necesarios cinco pasos:

- Cree un objeto `SpeechConfig` a partir de la clave y la región de suscripción.
- Actualice el objeto `SpeechConfig` para especificar los idiomas de origen y destino.
- Cree un objeto `TranslationRecognizer` con el objeto `SpeechConfig` anterior.
- Con el objeto `TranslationRecognizer`, inicie el proceso de reconocimiento de una única expresión.
- Inspeccione el objeto `TranslationRecognitionResult` devuelto.

Si prefiere ponerse a trabajar de inmediato, vea o descargue todos los [ejemplos para Python del SDK de Voz](#) en GitHub. Si no, vamos a comenzar.

## Prerequisites

Antes de comenzar, compruebe lo siguiente:

- [Ha creado un recurso de Voz de Azure](#)
- [Ha configurado el entorno de desarrollo](#)
- [Ha creado un proyecto de ejemplo vacío](#)

## Incorporación de código de ejemplo

1. Abra `quickstart.py` y reemplace todo el código que contiene por lo siguiente.

```

import azure.cognitiveservices.speech as speechsdk

speech_key, service_region = "YourSubscriptionKey", "YourServiceRegion"

def translate_speech_to_text():

    # Creates an instance of a speech translation config with specified subscription key and service
    region.
    # Replace with your own subscription key and service region (e.g., "westus").
    translation_config = speechsdk.translation.SpeechTranslationConfig(subscription=speech_key,
    region=service_region)

    # Sets source and target languages.
    # Replace with the languages of your choice, from list found here: https://aka.ms/speech/sttt-
    languages
    fromLanguage = 'en-US'
    toLanguage = 'de'
    translation_config.speech_recognition_language = fromLanguage
    translation_config.add_target_language(toLanguage)

    # Creates a translation recognizer using and audio file as input.
    recognizer = speechsdk.translation.TranslationRecognizer(translation_config=translation_config)

    # Starts translation, and returns after a single utterance is recognized. The end of a
    # single utterance is determined by listening for silence at the end or until a maximum of 15
    # seconds of audio is processed. It returns the recognized text as well as the translation.
    # Note: Since recognize_once() returns only a single utterance, it is suitable only for single
    # shot recognition like command or query.
    # For long-running multi-utterance recognition, use start_continuous_recognition() instead.
    print("Say something...")
    result = recognizer.recognize_once()

    # Check the result
    if result.reason == speechsdk.ResultReason.TranslatedSpeech:
        print("RECOGNIZED '{}': {}".format(fromLanguage, result.text))
        print("TRANSLATED into {}: {}".format(toLanguage, result.translations['de']))
    elif result.reason == speechsdk.ResultReason.RecognizedSpeech:
        print("RECOGNIZED: {} (text could not be translated)".format(result.text))
    elif result.reason == speechsdk.ResultReason.NoMatch:
        print("NOMATCH: Speech could not be recognized: {}".format(result.no_match_details))
    elif result.reason == speechsdk.ResultReason.Canceled:
        print("CANCELED: Reason={}".format(result.cancellation_details.reason))
        if result.cancellation_details.reason == speechsdk.CancellationReason.Error:
            print("CANCELED: ErrorDetails={}".format(result.cancellation_details.error_details))

translate_speech_to_text()

```

2. En el mismo archivo, reemplace la cadena `YourSubscriptionKey` por la clave de suscripción.
3. Reemplace la cadena `YourServiceRegion` por la **región** asociada a sus suscripción (por ejemplo, `westus` para la suscripción de evaluación gratuita).
4. Guarde los cambios realizados en `quickstart.py`.

## Compilación y ejecución de la aplicación

1. Ejecute el ejemplo desde la consola o en el IDE:

```
python quickstart.py
```

2. Diga una oración o frase en inglés. La aplicación transmite la voz al servicio de voz, que la traduce (alemán en este caso) y la transcribe en texto. Despues, el servicio de voz envía el texto de nuevo a la

aplicación para mostrarlo.

```
Say something...
RECOGNIZED 'en-US': What's the weather in Seattle?
TRANSLATED into 'de': Wie ist das Wetter in Seattle?
```

## Pasos siguientes

[Exploración de los ejemplos de Python en GitHub](#)

Vea o descargue todos los [ejemplos del SDK de Voz](#) en GitHub.

## Compatibilidad con plataformas y lenguajes adicionales

Si ha hecho clic en esta pestaña, es probable que no vea un inicio rápido en su lenguaje de programación favorito. No se preocupe, tenemos materiales de inicio rápido y ejemplos de código adicionales disponibles en GitHub. Use la tabla para encontrar el ejemplo correcto para su lenguaje de programación y combinación de plataforma y sistema operativo.

IDIOMA	EJEMPLOS DE CÓDIGO
C++	<a href="#">Inicios rápidos</a> , <a href="#">Ejemplos</a>
C#	<a href="#">.NET Framework</a> , <a href="#">.NET Core</a> , <a href="#">UWP</a> , <a href="#">Unity</a> , <a href="#">Xamarin</a>
Java	<a href="#">Android</a> , <a href="#">JRE</a>
JavaScript	<a href="#">Browser</a>
Node.js	<a href="#">Windows</a> , <a href="#">Linux</a> , <a href="#">macOS</a>
Objective-C	<a href="#">iOS</a> , <a href="#">macOS</a>
Python	<a href="#">Windows</a> , <a href="#">Linux</a> , <a href="#">macOS</a>
Swift	<a href="#">iOS</a> , <a href="#">macOS</a>

# Inicio rápido: Traducción de voz a varios idiomas

15/01/2020 • 22 minutes to read • [Edit Online](#)

En este inicio rápido, va a usar el SDK de [Voz](#) para traducir interactivamente la voz de un idioma a voz en otro idioma. Una vez que se cumplen los requisitos previos, para la traducción de voz a texto en varios idiomas solo son necesarios seis pasos:

- Cree un objeto `SpeechTranslationConfig` a partir de la clave y la región de suscripción.
- Actualice el objeto `SpeechTranslationConfig` para especificar el idioma de origen del reconocimiento de voz.
- Actualice el objeto `SpeechTranslationConfig` para especificar varios idiomas de destino para la traducción.
- Cree un objeto `TranslationRecognizer` con el objeto `SpeechTranslationConfig` anterior.
- Con el objeto `TranslationRecognizer`, inicie el proceso de reconocimiento de una única expresión.
- Inspeccione el objeto `TranslationRecognitionResult` devuelto.

Si prefiere ponerse a trabajar de inmediato, vea o descargue todos los [ejemplos para C# del SDK de Voz](#) en GitHub. Si no, vamos a comenzar.

## Selección del entorno de destino

- [.NET](#)
- [.NET Core](#)

## Prerequisites

Antes de comenzar, compruebe lo siguiente:

- [Ha creado un recurso de Voz de Azure](#)
- [Ha configurado el entorno de desarrollo](#)
- [Ha creado un proyecto de ejemplo vacío](#)

## Incorporación de código de ejemplo

1. Abra el archivo **Program.cs** y sustituya todo el código existente por el siguiente.

```
using System;
using System.Threading.Tasks;
using Microsoft.CognitiveServices.Speech;
using Microsoft.CognitiveServices.Speech.Translation;

namespace helloworld
{
    class Program
    {
        public static async Task TranslateSpeechToText()
        {
            // Creates an instance of a speech translation config with specified subscription key and
            service region.
            // Replace with your own subscription key and service region (e.g., "westus").
            var config = SpeechTranslationConfig.FromSubscription("YourSubscriptionKey",
            "YourServiceRegion");

            // Sets source and target languages.
            // Replace with the languages of your choice, from list found here:
            https://aka.ms/speech/sttt-languages
            string fromLanguage = "en-US";
```

```

        config.SpeechRecognitionLanguage = fromLanguage;
        config.AddTargetLanguage("de");
        config.AddTargetLanguage("fr");

        // Creates a translation recognizer using the default microphone audio input device.
        using (var recognizer = new TranslationRecognizer(config))
        {
            // Starts translation, and returns after a single utterance is recognized. The end of a
            // single utterance is determined by listening for silence at the end or until a maximum
            of 15
            // seconds of audio is processed. The task returns the recognized text as well as the
            translation.
            // Note: Since RecognizeOnceAsync() returns only a single utterance, it is suitable only
            for single
            // shot recognition like command or query.
            // For long-running multi-utterance recognition, use StartContinuousRecognitionAsync()
            instead.

            Console.WriteLine("Say something...");
            var result = await recognizer.RecognizeOnceAsync();

            // Checks result.
            if (result.Reason == ResultReason.TranslatedSpeech)
            {
                Console.WriteLine($"RECOGNIZED '{fromLanguage}': {result.Text}");
                foreach (var element in result.Translations)
                {
                    Console.WriteLine($"TRANSLATED into '{element.Key}': {element.Value}");
                }
            }
            else if (result.Reason == ResultReason.RecognizedSpeech)
            {
                Console.WriteLine($"RECOGNIZED '{fromLanguage}': {result.Text} (text could not be
translated)");
            }
            else if (result.Reason == ResultReason.NoMatch)
            {
                Console.WriteLine($"NOMATCH: Speech could not be recognized.");
            }
            else if (result.Reason == ResultReason.Canceled)
            {
                var cancellation = CancellationDetails.FromResult(result);
                Console.WriteLine($"CANCELED: Reason={cancellation.Reason}");

                if (cancellation.Reason == CancellationReason.Error)
                {
                    Console.WriteLine($"CANCELED: ErrorCode={cancellation.ErrorCode}");
                    Console.WriteLine($"CANCELED: ErrorDetails={cancellation.ErrorDetails}");
                    Console.WriteLine($"CANCELED: Did you update the subscription info?");
                }
            }
        }

        static void Main(string[] args)
        {
            TranslateSpeechToText().Wait();
        }
    }
}

```

2. En el mismo archivo, reemplace la cadena `YourSubscriptionKey` por la clave de suscripción.
3. Reemplace la cadena `YourServiceRegion` por la [región](#) asociada a sus suscripción (por ejemplo, `westus` para la suscripción de evaluación gratuita).
4. En la barra de menús, elija **Archivo > Guardar todo**.

# Compilación y ejecución de la aplicación

1. En la barra de menús, seleccione **Compilar > Compilar solución** para compilar la aplicación. El código se debería compilar sin errores ahora.
2. Elija **Depurar > Iniciar depuración** o presione **F5** para iniciar la aplicación **HelloWorld**.
3. Diga una oración o frase en inglés. La aplicación transmite la voz al servicio de voz, que la traduce (a francés y alemán en este caso) y la transcribe en texto. Después, el servicio de voz envía el texto de nuevo a la aplicación para mostrarlo.

```
Say something...
RECOGNIZED 'en-US': What's the weather in Seattle?
TRANSLATED into 'de': Wie ist das Wetter in Seattle?
TRANSLATED into 'fr': Quel temps fait-il à Seattle ?
```

## Pasos siguientes

### Exploración de ejemplos de C# en GitHub

En este inicio rápido, va a usar el SDK de [Voz](#) para traducir interactivamente la voz de un idioma a voz en otro idioma. Una vez que se cumplen los requisitos previos, para la traducción de voz a texto en varios idiomas solo son necesarios seis pasos:

- Cree un objeto `SpeechTranslationConfig` a partir de la clave y la región de suscripción.
- Actualice el objeto `SpeechTranslationConfig` para especificar el idioma de origen del reconocimiento de voz.
- Actualice el objeto `SpeechTranslationConfig` para especificar varios idiomas de destino para la traducción.
- Cree un objeto `TranslationRecognizer` con el objeto `SpeechTranslationConfig` anterior.
- Con el objeto `TranslationRecognizer`, inicie el proceso de reconocimiento de una única expresión.
- Inspeccione el objeto `TranslationRecognitionResult` devuelto.

Si prefiere ponerse a trabajar de inmediato, vea o descargue todos los [ejemplos para C++ del SDK de Voz](#) en GitHub. Si no, vamos a comenzar.

## Prerequisites

Antes de comenzar, compruebe lo siguiente:

- [Ha creado un recurso de Voz de Azure](#)
- [Ha configurado el entorno de desarrollo](#)
- [Ha creado un proyecto de ejemplo vacío](#)

## Incorporación de código de ejemplo

1. Abra el archivo de origen **helloworld.cpp**.
2. Reemplace todo el código por el fragmento siguiente:

```
#include <iostream>
#include <vector>
#include <speechapi_cxx.h>

using namespace std;
using namespace Microsoft::CognitiveServices::Speech;
using namespace Microsoft::CognitiveServices::Speech::Translation;
```

```

void TranslateSpeechToText()
{
    // Creates an instance of a speech translation config with specified subscription key and service
    // region.
    // Replace with your own subscription key and service region (e.g., "westus").
    auto config = SpeechTranslationConfig::FromSubscription("YourSubscriptionKey", "YourServiceRegion");

    // Sets source and target languages.
    // Replace with the languages of your choice, from list found here: https://aka.ms/speech/sttt-
    languages
    auto fromLanguage = "en-US";
    config->SetSpeechRecognitionLanguage(fromLanguage);
    config->AddTargetLanguage("de");
    config->AddTargetLanguage("fr");

    // Creates a translation recognizer using the default microphone audio input device.
    auto recognizer = TranslationRecognizer::FromConfig(config);

    // Starts translation, and returns after a single utterance is recognized. The end of a
    // single utterance is determined by listening for silence at the end or until a maximum of 15
    // seconds of audio is processed. The task returns the recognized text as well as the translation.
    // Note: Since RecognizeOnceAsync() returns only a single utterance, it is suitable only for single
    // shot recognition like command or query.
    // For long-running multi-utterance recognition, use StartContinuousRecognitionAsync() instead.
    cout << "Say something...\n";
    auto result = recognizer->RecognizeOnceAsync().get();

    // Checks result.
    if (result->Reason == ResultReason::TranslatedSpeech)
    {
        cout << "RECOGNIZED '" << fromLanguage << "'": " << result->Text << std::endl;
        for (const auto& it : result->Translations)
        {
            cout << "TRANSLATED into '" << it.first.c_str() << "'": " << it.second.c_str() << std::endl;
        }
    }
    else if (result->Reason == ResultReason::RecognizedSpeech)
    {
        cout << "RECOGNIZED '" << fromLanguage << "' " << result->Text << " (text could not be
        translated)" << std::endl;
    }
    else if (result->Reason == ResultReason::NoMatch)
    {
        cout << "NOMATCH: Speech could not be recognized." << std::endl;
    }
    else if (result->Reason == ResultReason::Canceled)
    {
        auto cancellation = CancellationDetails::FromResult(result);
        cout << "CANCELED: Reason=" << (int)cancellation->Reason << std::endl;

        if (cancellation->Reason == CancellationReason::Error)
        {
            cout << "CANCELED: ErrorCode=" << (int)cancellation->ErrorCode << std::endl;
            cout << "CANCELED: ErrorDetails=" << cancellation->ErrorDetails << std::endl;
            cout << "CANCELED: Did you update the subscription info?" << std::endl;
        }
    }
}

int wmain()
{
    TranslateSpeechToText();
    return 0;
}

```

3. En el mismo archivo, reemplace la cadena `YourSubscriptionKey` por la clave de suscripción.
4. Reemplace la cadena `YourServiceRegion` por la **región** asociada a sus suscripción (por ejemplo, `westus`) para

la suscripción de evaluación gratuita).

5. En la barra de menús, elija **Archivo** > **Guardar todo**.

## Compilación y ejecución de la aplicación

1. En la barra de menús, seleccione **Compilar** > **Compilar solución** para compilar la aplicación. El código se debería compilar sin errores ahora.
2. Elija **Depurar** > **Iniciar depuración** o presione **F5** para iniciar la aplicación **HelloWorld**.
3. Diga una oración o frase en inglés. La aplicación transmite la voz al servicio de voz, que la traduce (a francés y alemán en este caso) y la transcribe en texto. Después, el servicio de voz envía el texto de nuevo a la aplicación para mostrarlo.

```
Say something...
RECOGNIZED 'en-US': What's the weather in Seattle?
TRANSLATED into 'de': Wie ist das Wetter in Seattle?
TRANSLATED into 'fr': Quel temps fait-il à Seattle ?
```

## Pasos siguientes

### Exploración de ejemplos de C++ en GitHub

En este inicio rápido, va a usar el SDK de [Voz](#) para traducir interactivamente la voz de un idioma a voz en otro idioma. Una vez que se cumplen los requisitos previos, para la traducción de voz a texto en varios idiomas solo son necesarios seis pasos:

- Cree un objeto `SpeechTranslationConfig` a partir de la clave y la región de suscripción.
- Actualice el objeto `SpeechTranslationConfig` para especificar el idioma de origen del reconocimiento de voz.
- Actualice el objeto `SpeechTranslationConfig` para especificar varios idiomas de destino para la traducción.
- Cree un objeto `TranslationRecognizer` con el objeto `SpeechTranslationConfig` anterior.
- Con el objeto `TranslationRecognizer`, inicie el proceso de reconocimiento de una única expresión.
- Inspeccione el objeto `TranslationRecognitionResult` devuelto.

Si prefiere ponerse a trabajar de inmediato, vea o descargue todos los [ejemplos para Java del SDK de Voz](#) en GitHub. Si no, vamos a comenzar.

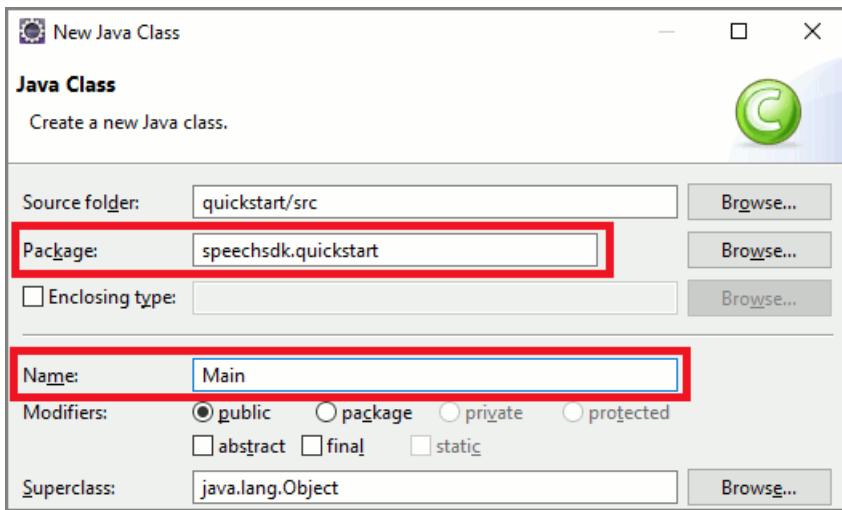
## Prerequisites

Antes de comenzar, compruebe lo siguiente:

- [Ha creado un recurso de Voz de Azure](#)
- [Ha configurado el entorno de desarrollo](#)
- [Ha creado un proyecto de ejemplo vacío](#)

## Incorporación de código de ejemplo

1. Para agregar una nueva clase vacía al proyecto de Java, seleccione **File (Archivo)** > **New (Nuevo)** > **Class (Clase)**.
2. En la ventana **New Java Class** (Nueva clase de Java) escriba **speechsdk.quickstart** en el campo **Package** (Paquete) y **Main** en el campo **Name** (Nombre).



3. Reemplace el código en `Main.java` con el siguiente fragmento de código:

```
package quickstart;

import java.io.IOException;
import java.util.concurrent.Future;
import java.util.Map;
import java.util.concurrent.ExecutionException;
import com.microsoft.cognitiveservices.speech.*;
import com.microsoft.cognitiveservices.speech.translation.*;

public class Main {

    public static void translationWithMicrophoneAsync() throws InterruptedException, ExecutionException,
IOException
    {
        // Creates an instance of a speech translation config with specified
        // subscription key and service region. Replace with your own subscription key
        // and service region (e.g., "westus").

        int exitCode = 1;
        SpeechTranslationConfig config = SpeechTranslationConfig.fromSubscription("YourSubscriptionKey",
"YourServiceRegion");
        assert(config != null);

        // Sets source and target languages.
        String fromLanguage = "en-US";
        config.setSpeechRecognitionLanguage(fromLanguage);
        config.addTargetLanguage("de");
        config.addTargetLanguage("fr");

        // Creates a translation recognizer using the default microphone audio input device.
        TranslationRecognizer recognizer = new TranslationRecognizer(config);
        assert(recognizer != null);

        System.out.println("Say something...");

        // Starts translation, and returns after a single utterance is recognized. The end of a
        // single utterance is determined by listening for silence at the end or until a maximum of 15
        // seconds of audio is processed. The task returns the recognized text as well as the
        translation.
        // Note: Since recognizeOnceAsync() returns only a single utterance, it is suitable only for
single
        // shot recognition like command or query.
        // For long-running multi-utterance recognition, use startContinuousRecognitionAsync() instead.
        Future<TranslationRecognitionResult> task = recognizer.recognizeOnceAsync();
        assert(task != null);

        TranslationRecognitionResult result = task.get();
        assert(result != null);
    }
}
```

```

        if (result.getReason() == ResultReason.TranslatedSpeech) {
            System.out.println("RECOGNIZED '" + fromLanguage + "': " + result.getText());

            Map<String, String> map = result.getTranslations();
            for(String toLanguage: map.keySet()) {
                System.out.println("TRANSLATED into '" + toLanguage + "': " + map.get(toLanguage));
            }
            exitCode = 0;
        }
        else if (result.getReason() == ResultReason.RecognizedSpeech) {
            System.out.println("RECOGNIZED '" + fromLanguage + "': " + result.getText() + "(text could
not be translated)");
            exitCode = 0;
        }
        else if (result.getReason() == ResultReason.NoMatch) {
            System.out.println("NOMATCH: Speech could not be recognized.");
        }
        else if (result.getReason() == ResultReason.Canceled) {
            CancellationDetails cancellation = CancellationDetails.fromResult(result);
            System.out.println("CANCELED: Reason=" + cancellation.getReason());

            if (cancellation.getReason() == CancellationReason.Error) {
                System.out.println("CANCELED: ErrorCode=" + cancellation.getErrorCode());
                System.out.println("CANCELED: ErrorDetails=" + cancellation.getErrorDetails());
                System.out.println("CANCELED: Did you update the subscription info?");
            }
        }
    }

    recognizer.close();

    System.exit(exitCode);
}

public static void main(String[] args) {
    try {
        translationWithMicrophoneAsync();
    } catch (Exception ex) {
        System.out.println("Unexpected exception: " + ex.getMessage());
        assert(false);
        System.exit(1);
    }
}
}

```

4. Reemplace la cadena `YourSubscriptionKey` por la clave de suscripción.
5. Reemplace la cadena `YourServiceRegion` por la [región](#) asociada a sus suscripción (por ejemplo, `westus` para la suscripción de evaluación gratuita).
6. Guarde los cambios en el proyecto.

## Compilación y ejecución de la aplicación

Presione F11, o seleccione **Run (Ejecutar) > Debug (Depurar)**.

1. Diga una oración o frase en inglés. La aplicación transmite la voz al servicio de voz, que la traduce (a francés y alemán en este caso) y la transcribe en texto. Después, el servicio de voz envía el texto de nuevo a la aplicación para mostrarlo.

```
Say something...
RECOGNIZED 'en-US': What's the weather in Seattle?
TRANSLATED into 'de': Wie ist das Wetter in Seattle?
TRANSLATED into 'fr': Quel temps fait-il à Seattle ?
```

## Pasos siguientes

### [Exploración de ejemplos de Java en GitHub](#)

En este inicio rápido, va a usar el SDK de [Voz](#) para traducir interactivamente la voz de un idioma a voz en otro idioma. Una vez que se cumplen los requisitos previos, para la traducción de voz a texto en varios idiomas solo son necesarios seis pasos:

- Cree un objeto `SpeechTranslationConfig` a partir de la clave y la región de suscripción.
- Actualice el objeto `SpeechTranslationConfig` para especificar el idioma de origen del reconocimiento de voz.
- Actualice el objeto `SpeechTranslationConfig` para especificar varios idiomas de destino para la traducción.
- Cree un objeto `TranslationRecognizer` con el objeto `SpeechTranslationConfig` anterior.
- Con el objeto `TranslationRecognizer`, inicie el proceso de reconocimiento de una única expresión.
- Inspeccione el objeto `TranslationRecognitionResult` devuelto.

Si prefiere ponerse a trabajar de inmediato, vea o descargue todos los [ejemplos para Python del SDK de Voz](#) en GitHub. Si no, vamos a comenzar.

## Prerequisites

Antes de comenzar, compruebe lo siguiente:

- [Ha creado un recurso de Voz de Azure](#)
- [Ha configurado el entorno de desarrollo](#)
- [Ha creado un proyecto de ejemplo vacío](#)

## Incorporación de código de ejemplo

1. Abra `quickstart.py` y reemplace todo el código que contiene por lo siguiente.

```

import azure.cognitiveservices.speech as speechsdk

speech_key, service_region = "YourSubscriptionKey", "YourServiceRegion"

def translate_speech_to_text():

    # Creates an instance of a speech translation config with specified subscription key and service
    region.
    # Replace with your own subscription key and service region (e.g., "westus").
    translation_config = speechsdk.translation.SpeechTranslationConfig(subscription=speech_key,
    region=service_region)

    # Sets source and target languages.
    # Replace with the languages of your choice, from list found here: https://aka.ms/speech/sttt-
    languages
    fromLanguage = 'en-US'
    translation_config.speech_recognition_language = fromLanguage
    translation_config.add_target_language('de')
    translation_config.add_target_language('fr')

    # Creates a translation recognizer using an audio file as input.
    recognizer = speechsdk.translation.TranslationRecognizer(translation_config=translation_config)

    # Starts translation, and returns after a single utterance is recognized. The end of a
    # single utterance is determined by listening for silence at the end or until a maximum of 15
    # seconds of audio is processed. It returns the recognized text as well as the translation.
    # Note: Since recognize_once() returns only a single utterance, it is suitable only for single
    # shot recognition like command or query.
    # For long-running multi-utterance recognition, use start_continuous_recognition() instead.
    print("Say something...")
    result = recognizer.recognize_once()

    # Check the result
    if result.reason == speechsdk.ResultReason.TranslatedSpeech:
        print("RECOGNIZED '{}': {}".format(fromLanguage, result.text))
        print("TRANSLATED into {}: {}".format('de', result.translations['de']))
        print("TRANSLATED into {}: {}".format('fr', result.translations['fr']))
    elif result.reason == speechsdk.ResultReason.RecognizedSpeech:
        print("RECOGNIZED: {} (text could not be translated)".format(result.text))
    elif result.reason == speechsdk.ResultReason.NoMatch:
        print("NOMATCH: Speech could not be recognized: {}".format(result.no_match_details))
    elif result.reason == speechsdk.ResultReason.Canceled:
        print("CANCELED: Reason={}".format(result.cancellation_details.reason))
        if result.cancellation_details.reason == speechsdk.CancellationReason.Error:
            print("CANCELED: ErrorDetails={}".format(result.cancellation_details.error_details))

translate_speech_to_text()

```

2. En el mismo archivo, reemplace la cadena `YourSubscriptionKey` por la clave de suscripción.
3. Reemplace la cadena `YourServiceRegion` por la [región](#) asociada a sus suscripciones (por ejemplo, `westus` para la suscripción de evaluación gratuita).
4. Guarde los cambios realizados en `quickstart.py`.

## Compilación y ejecución de la aplicación

1. Ejecute el ejemplo desde la consola o en el IDE:

```
python quickstart.py
```

2. Diga una oración o frase en inglés. La aplicación transmite la voz al servicio de voz, que la traduce (a francés y alemán en este caso) y la transcribe en texto. Después, el servicio de voz envía el texto de nuevo a la

aplicación para mostrarlo.

```
Say something...
RECOGNIZED 'en-US': What's the weather in Seattle?
TRANSLATED into 'de': Wie ist das Wetter in Seattle?
TRANSLATED into 'fr': Quel temps fait-il à Seattle ?
```

## Pasos siguientes

[Exploración de los ejemplos de Python en GitHub](#)

Vea o descargue todos los [ejemplos del SDK de Voz](#) en GitHub.

## Compatibilidad con plataformas y lenguajes adicionales

Si ha hecho clic en esta pestaña, es probable que no vea un inicio rápido en su lenguaje de programación favorito. No se preocupe, tenemos materiales de inicio rápido y ejemplos de código adicionales disponibles en GitHub. Use la tabla para encontrar el ejemplo correcto para su lenguaje de programación y combinación de plataforma y sistema operativo.

IDIOMA	EJEMPLOS DE CÓDIGO
C++	<a href="#">Inicios rápidos</a> , <a href="#">Ejemplos</a>
C#	<a href="#">.NET Framework</a> , <a href="#">.NET Core</a> , <a href="#">UWP</a> , <a href="#">Unity</a> , <a href="#">Xamarin</a>
Java	<a href="#">Android</a> , <a href="#">JRE</a>
JavaScript	<a href="#">Browser</a>
Node.js	<a href="#">Windows</a> , <a href="#">Linux</a> , <a href="#">macOS</a>
Objective-C	<a href="#">iOS</a> , <a href="#">macOS</a>
Python	<a href="#">Windows</a> , <a href="#">Linux</a> , <a href="#">macOS</a>
Swift	<a href="#">iOS</a> , <a href="#">macOS</a>

# Inicio rápido: Traducción de voz a voz

15/01/2020 • 23 minutes to read • [Edit Online](#)

En este inicio rápido, va a usar el SDK de [Voz](#) para traducir interactivamente la voz de un idioma a voz en otro idioma. Una vez que se cumplen los requisitos previos, para la traducción de voz a voz solo son necesarios seis pasos:

- Cree un objeto `SpeechTranslationConfig` a partir de la clave y la región de suscripción.
- Actualice el objeto `SpeechTranslationConfig` para especificar los idiomas de origen y destino.
- Actualice el objeto `SpeechTranslationConfig` para especificar el nombre de la voz de la salida de voz.
- Cree un objeto `TranslationRecognizer` con el objeto `SpeechTranslationConfig` anterior.
- Con el objeto `TranslationRecognizer`, inicie el proceso de reconocimiento de una única expresión.
- Inspeccione el objeto `TranslationRecognitionResult` devuelto.

Si prefiere ponerse a trabajar de inmediato, vea o descargue todos los [ejemplos para C# del SDK de Voz](#) en GitHub. Si no, vamos a comenzar.

## Selección del entorno de destino

- [.NET](#)
- [.NET Core](#)

## Prerequisites

Antes de comenzar, compruebe lo siguiente:

- [Ha creado un recurso de Voz de Azure](#)
- [Ha configurado el entorno de desarrollo](#)
- [Ha creado un proyecto de ejemplo vacío](#)

## Incorporación de código de ejemplo

1. Abra el archivo **Program.cs** y sustituya todo el código existente por el siguiente.

```
using System;
using System.Threading.Tasks;
using Microsoft.CognitiveServices.Speech;
using Microsoft.CognitiveServices.Speech.Translation;

namespace helloworld
{
    class Program
    {
        public static async Task TranslateSpeechToSpeech()
        {
            // Creates an instance of a speech translation config with specified subscription key and
            service region.
            // Replace with your own subscription key and service region (e.g., "westus").
            var config = SpeechTranslationConfig.FromSubscription("YourSubscriptionKey",
            "YourServiceRegion");

            // Sets source and target languages.
            // Replace with the languages of your choice, from list found here:
            https://aka.ms/speech/sttt-languages
            string fromLanguage = "en-US";
```

```

        string toLanguage = "de";
        config.SpeechRecognitionLanguage = fromLanguage;
        config.AddTargetLanguage(toLanguage);

        // Sets the synthesis output voice name.
        // Replace with the languages of your choice, from list found here:
https://aka.ms/speech/tts-languages
        config.VoiceName = "de-DE-Hedda";

        // Creates a translation recognizer using the default microphone audio input device.
        using (var recognizer = new TranslationRecognizer(config))
        {
            // Prepare to handle the synthesized audio data.
            recognizer.Synthesizing += (s, e) =>
            {
                var audio = e.Result.GetAudio();
                Console.WriteLine(audio.Length != 0
                    ? $"AUDIO SYNTHESIZED: {audio.Length} byte(s)"
                    : $"AUDIO SYNTHESIZED: {audio.Length} byte(s) (COMPLETE)");
            };

            // Starts translation, and returns after a single utterance is recognized. The end of a
            // single utterance is determined by listening for silence at the end or until a maximum
            of 15
            // seconds of audio is processed. The task returns the recognized text as well as the
            translation.
            // Note: Since RecognizeOnceAsync() returns only a single utterance, it is suitable only
            for single
            // shot recognition like command or query.
            // For long-running multi-utterance recognition, use StartContinuousRecognitionAsync()
            instead.
            Console.WriteLine("Say something...");
            var result = await recognizer.RecognizeOnceAsync();

            // Checks result.
            if (result.Reason == ResultReason.TranslatedSpeech)
            {
                Console.WriteLine($"RECOGNIZED '{fromLanguage}': {result.Text}");
                Console.WriteLine($"TRANSLATED into '{toLanguage}'":
{result.Translations[toLanguage]}");
            }
            else if (result.Reason == ResultReason.RecognizedSpeech)
            {
                Console.WriteLine($"RECOGNIZED '{fromLanguage}': {result.Text} (text could not be
translated)");
            }
            else if (result.Reason == ResultReason.NoMatch)
            {
                Console.WriteLine($"NOMATCH: Speech could not be recognized.");
            }
            else if (result.Reason == ResultReason.Canceled)
            {
                var cancellation = CancellationDetails.FromResult(result);
                Console.WriteLine($"CANCELED: Reason={cancellation.Reason}");

                if (cancellation.Reason == CancellationReason.Error)
                {
                    Console.WriteLine($"CANCELED: ErrorCode={cancellation.ErrorCode}");
                    Console.WriteLine($"CANCELED: ErrorDetails={cancellation.ErrorDetails}");
                    Console.WriteLine($"CANCELED: Did you update the subscription info?");
                }
            }
        }

        static void Main(string[] args)
        {
            TranslateSpeechToSpeech().Wait();
        }
    
```

```
        }  
    }
```

2. En el mismo archivo, reemplace la cadena `YourSubscriptionKey` por la clave de suscripción.
3. Reemplace la cadena `YourServiceRegion` por la **región** asociada a sus suscripción (por ejemplo, `westus` para la suscripción de evaluación gratuita).
4. En la barra de menús, elija **Archivo > Guardar todo**.

## Compilación y ejecución de la aplicación

1. En la barra de menús, seleccione **Compilar > Compilar solución** para compilar la aplicación. El código se debería compilar sin errores ahora.
2. Elija **Depurar > Iniciar depuración** o presione **F5** para iniciar la aplicación **HelloWorld**.
3. Diga una oración o frase en inglés. La aplicación transmite la voz al servicio de voz, que la traduce (a alemán en este caso) y la transcribe en texto. Después, el servicio de voz envía el audio sintetizado y el texto de nuevo a la aplicación para mostrarlo.

```
Say something...  
AUDIO SYNTHESIZED: 76784 byte(s)  
AUDIO SYNTHESIZED: 0 byte(s)(COMPLETE)  
RECOGNIZED 'en-US': What's the weather in Seattle?  
TRANSLATED into 'de': Wie ist das Wetter in Seattle?
```

## Pasos siguientes

### Exploración de ejemplos de C# en GitHub

En este inicio rápido, va a usar el SDK de [Voz](#) para traducir interactivamente la voz de un idioma a voz en otro idioma. Una vez que se cumplen los requisitos previos, para la traducción de voz a voz solo son necesarios seis pasos:

- Cree un objeto `SpeechTranslationConfig` a partir de la clave y la región de suscripción.
- Actualice el objeto `SpeechTranslationConfig` para especificar los idiomas de origen y destino.
- Actualice el objeto `SpeechTranslationConfig` para especificar el nombre de la voz de la salida de voz.
- Cree un objeto `TranslationRecognizer` con el objeto `SpeechTranslationConfig` anterior.
- Con el objeto `TranslationRecognizer`, inicie el proceso de reconocimiento de una única expresión.
- Inspeccione el objeto `TranslationRecognitionResult` devuelto.

Si prefiere ponerse a trabajar de inmediato, vea o descargue todos los [ejemplos para C++ del SDK de Voz](#) en GitHub. Si no, vamos a comenzar.

## Prerequisites

Antes de comenzar, compruebe lo siguiente:

- [Ha creado un recurso de Voz de Azure](#)
- [Ha configurado el entorno de desarrollo](#)
- [Ha creado un proyecto de ejemplo vacío](#)

## Incorporación de código de ejemplo

1. Abra el archivo de origen **helloworld.cpp**.

2. Reemplace todo el código por el fragmento siguiente:

```
#include <iostream>
#include <vector>
#include <speechapi_cxx.h>

using namespace std;
using namespace Microsoft::CognitiveServices::Speech;
using namespace Microsoft::CognitiveServices::Speech::Translation;

void TranslateSpeechToSpeech()
{
    // Creates an instance of a speech translation config with specified subscription key and service
    // region.
    // Replace with your own subscription key and service region (e.g., "westus").
    auto config = SpeechTranslationConfig::FromSubscription("YourSubscriptionKey", "YourServiceRegion");

    // Sets source and target languages.
    // Replace with the languages of your choice, from list found here: https://aka.ms/speech/sttt-
    languages
    auto fromLanguage = "en-US";
    auto toLanguage = "de";
    config->SetSpeechRecognitionLanguage(fromLanguage);
    config->AddTargetLanguage(toLanguage);

    // Sets the synthesis output voice name.
    // Replace with the languages of your choice, from list found here: https://aka.ms/speech/tts-
    languages
    config->SetVoiceName("de-DE-Hedda");

    // Creates a translation recognizer using the default microphone audio input device.
    auto recognizer = TranslationRecognizer::FromConfig(config);

    // Prepare to handle the synthesized audio data.
    recognizer->Synthesizing.Connect([](const TranslationSynthesisEventArgs& e)
    {
        auto size = e.Result->Audio.size();
        cout << "AUDIO SYNTHESIZED: " << size << " byte(s)" << (size == 0 ? "(COMPLETE)" : "") <<
        std::endl;
    });

    // Starts translation, and returns after a single utterance is recognized. The end of a
    // single utterance is determined by listening for silence at the end or until a maximum of 15
    // seconds of audio is processed. The task returns the recognized text as well as the translation.
    // Note: Since RecognizeOnceAsync() returns only a single utterance, it is suitable only for single
    // shot recognition like command or query.
    // For long-running multi-utterance recognition, use StartContinuousRecognitionAsync() instead.
    cout << "Say something...\n";
    auto result = recognizer->RecognizeOnceAsync().get();

    // Checks result.
    if (result->Reason == ResultReason::TranslatedSpeech)
    {
        cout << "RECOGNIZED '" << fromLanguage << "' : " << result->Text << std::endl;
        cout << "TRANSLATED into '" << toLanguage << "' : " << result->Translations.at(toLanguage) <<
        std::endl;
    }
    else if (result->Reason == ResultReason::RecognizedSpeech)
    {
        cout << "RECOGNIZED '" << fromLanguage << "' " << result->Text << " (text could not be
        translated)" << std::endl;
    }
    else if (result->Reason == ResultReason::NoMatch)
    {
        cout << "NOMATCH: Speech could not be recognized." << std::endl;
    }
}
```

```

        ,
    else if (result->Reason == ResultReason::Canceled)
    {
        auto cancellation = CancellationDetails::FromResult(result);
        cout << "CANCELED: Reason=" << (int)cancellation->Reason << std::endl;

        if (cancellation->Reason == CancellationReason::Error)
        {
            cout << "CANCELED: ErrorCode=" << (int)cancellation->ErrorCode << std::endl;
            cout << "CANCELED: ErrorDetails=" << cancellation->ErrorDetails << std::endl;
            cout << "CANCELED: Did you update the subscription info?" << std::endl;
        }
    }

    int wmain()
    {
        TranslateSpeechToSpeech();
        return 0;
    }
}

```

3. En el mismo archivo, reemplace la cadena `YourSubscriptionKey` por la clave de suscripción.
4. Reemplace la cadena `YourServiceRegion` por la [región](#) asociada a sus suscripción (por ejemplo, `westus` para la suscripción de evaluación gratuita).
5. En la barra de menús, elija **Archivo > Guardar todo**.

## Compilación y ejecución de la aplicación

1. En la barra de menús, seleccione **Compilar > Compilar solución** para compilar la aplicación. El código se debería compilar sin errores ahora.
2. Elija **Depurar > Iniciar depuración** o presione **F5** para iniciar la aplicación **HelloWorld**.
3. Diga una oración o frase en inglés. La aplicación transmite la voz al servicio de voz, que la traduce (a alemán en este caso) y la transcribe en texto. Después, el servicio de voz envía el audio sintetizado y el texto de nuevo a la aplicación para mostrarlo.

```

Say something...
AUDIO SYNTHESIZED: 76784 byte(s)
AUDIO SYNTHESIZED: 0 byte(s)(COMPLETE)
RECOGNIZED 'en-US': What's the weather in Seattle?
TRANSLATED into 'de': Wie ist das Wetter in Seattle?

```

## Pasos siguientes

[Exploración de ejemplos de C++ en GitHub](#)

En este inicio rápido, va a usar el SDK de [Voz](#) para traducir interactivamente la voz de un idioma a voz en otro idioma. Una vez que se cumplen los requisitos previos, para la traducción de voz a voz solo son necesarios seis pasos:

- Cree un objeto `SpeechTranslationConfig` a partir de la clave y la región de suscripción.
- Actualice el objeto `SpeechTranslationConfig` para especificar los idiomas de origen y destino.
- Actualice el objeto `SpeechTranslationConfig` para especificar el nombre de la voz de la salida de voz.
- Cree un objeto `TranslationRecognizer` con el objeto `SpeechTranslationConfig` anterior.
- Con el objeto `TranslationRecognizer`, inicie el proceso de reconocimiento de una única expresión.
- Inspeccione el objeto `TranslationRecognitionResult` devuelto.

Si prefiere ponerse a trabajar de inmediato, vea o descargue todos los [ejemplos para Java del SDK de Voz](#) en GitHub. Si no, vamos a comenzar.

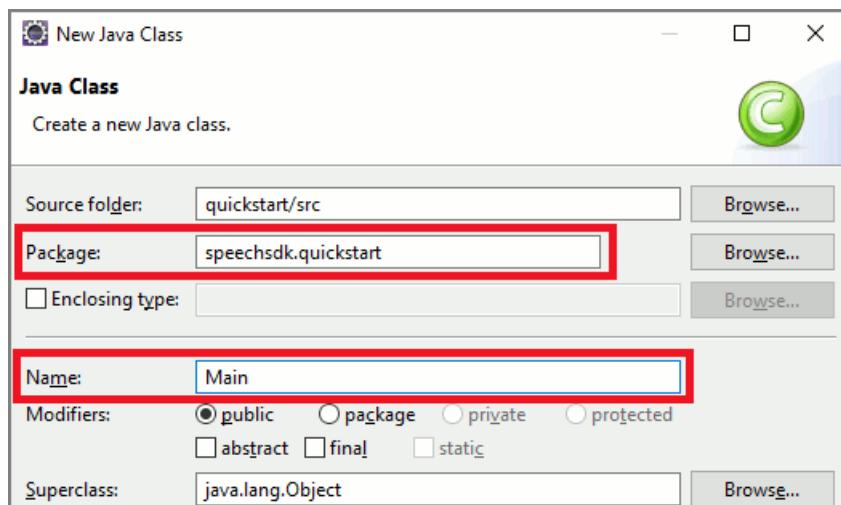
## Prerequisites

Antes de comenzar, compruebe lo siguiente:

- [Ha creado un recurso de Voz de Azure](#)
- [Ha configurado el entorno de desarrollo](#)
- [Ha creado un proyecto de ejemplo vacío](#)

## Incorporación de código de ejemplo

1. Para agregar una nueva clase vacía al proyecto de Java, seleccione **File (Archivo) > New (Nuevo) > Class (Clase)**.
2. En la ventana **New Java Class** (Nueva clase de Java) escriba **speechsdk.quickstart** en el campo **Package** (Paquete) y **Main** en el campo **Name** (Nombre).



3. Reemplace el código en **Main.java** con el siguiente fragmento de código:

```
package quickstart;

import java.io.IOException;
import java.util.concurrent.Future;
import java.util.concurrent.ExecutionException;
import com.microsoft.cognitiveservices.speech.*;
import com.microsoft.cognitiveservices.speech.translation.*;

public class Main {

    public static void translateSpeechToSpeech() throws InterruptedException, ExecutionException,
    IOException
    {
        // Creates an instance of a speech translation config with specified
        // subscription key and service region. Replace with your own subscription key
        // and service region (e.g., "westus").

        int exitCode = 1;
        SpeechTranslationConfig config = SpeechTranslationConfig.fromSubscription("YourSubscriptionKey",
        "YourServiceRegion");
        assert(config != null);

        // Sets source and target languages.
        String fromLanguage = "en-US";
        String toLanguage = "de";
    }
}
```

```

config.setSpeechRecognitionLanguage(fromLanguage);
config.addTargetLanguage(toLanguage);

// Sets the synthesis output voice name.
// Replace with the languages of your choice, from list found here: https://aka.ms/speech/tts-
languages
config.setVoiceName("de-DE-Hedda");

// Creates a translation recognizer using the default microphone audio input device.
TranslationRecognizer recognizer = new TranslationRecognizer(config);
assert(recognizer != null);

// Prepare to handle the synthesized audio data.
recognizer.synthesizing.addEventListen((s, e) -> {
    int size = e.getResult().getAudio().length;
    System.out.println(size != 0
        ? "AUDIO SYNTHESIZED: " + size + " byte(s)"
        : "AUDIO SYNTHESIZED: " + size + " byte(s) (COMPLETE)");
});

System.out.println("Say something...");

// Starts translation, and returns after a single utterance is recognized. The end of a
// single utterance is determined by listening for silence at the end or until a maximum of 15
// seconds of audio is processed. The task returns the recognized text as well as the
translation.
// Note: Since recognizeOnceAsync() returns only a single utterance, it is suitable only for
single
// shot recognition like command or query.
// For long-running multi-utterance recognition, use startContinuousRecognitionAsync() instead.
Future<TranslationRecognitionResult> task = recognizer.recognizeOnceAsync();
assert(task != null);

TranslationRecognitionResult result = task.get();
assert(result != null);

if (result.getReason() == ResultReason.TranslatedSpeech) {
    System.out.println("RECOGNIZED '" + fromLanguage + "':: " + result.getText());
    System.out.println("TRANSLATED into '" + toLanguage + "':: "
result.getTranslations().get(toLanguage));
    exitCode = 0;
}
else if (result.getReason() == ResultReason.RecognizedSpeech) {
    System.out.println("RECOGNIZED '" + fromLanguage + "':: " + result.getText() + "(text could
not be translated)");
    exitCode = 0;
}
else if (result.getReason() == ResultReason.NoMatch) {
    System.out.println("NOMATCH: Speech could not be recognized.");
}
else if (result.getReason() == ResultReason.Canceled) {
    CancellationDetails cancellation = CancellationDetails.fromResult(result);
    System.out.println("CANCELED: Reason=" + cancellation.getReason());

    if (cancellation.getReason() == CancellationReason.Error) {
        System.out.println("CANCELED: ErrorCode=" + cancellation.getErrorCode());
        System.out.println("CANCELED: ErrorDetails=" + cancellation.getErrorDetails());
        System.out.println("CANCELED: Did you update the subscription info?");
    }
}

recognizer.close();

System.exit(exitCode);
}

public static void main(String[] args) {
try {
    translateSpeechToSpeech();
}

```

```

        } catch (Exception ex) {
            System.out.println("Unexpected exception: " + ex.getMessage());
            assert(false);
            System.exit(1);
        }
    }
}

```

4. Reemplace la cadena `YourSubscriptionKey` por la clave de suscripción.
5. Reemplace la cadena `YourServiceRegion` por la [región](#) asociada a sus suscripción (por ejemplo, `westus` para la suscripción de evaluación gratuita).
6. Guarde los cambios en el proyecto.

## Compilación y ejecución de la aplicación

Presione F11, o seleccione **Run (Ejecutar) > Debug (Depurar)**.

1. Diga una oración o frase en inglés. La aplicación transmite la voz al servicio de voz, que la traduce (a alemán en este caso) y la transcribe en texto. Después, el servicio de voz envía el audio sintetizado y el texto de nuevo a la aplicación para mostrarlo.

```

Say something...
AUDIO SYNTHESIZED: 76784 byte(s)
AUDIO SYNTHESIZED: 0 byte(s)(COMPLETE)
RECOGNIZED 'en-US': What's the weather in Seattle?
TRANSLATED into 'de': Wie ist das Wetter in Seattle?

```

## Pasos siguientes

### [Exploración de ejemplos de Java en GitHub](#)

En este inicio rápido, va a usar el SDK de [Voz](#) para traducir interactivamente la voz de un idioma a voz en otro idioma. Una vez que se cumplen los requisitos previos, para la traducción de voz a voz solo son necesarios seis pasos:

- Cree un objeto `SpeechTranslationConfig` a partir de la clave y la región de suscripción.
- Actualice el objeto `SpeechTranslationConfig` para especificar los idiomas de origen y destino.
- Actualice el objeto `SpeechTranslationConfig` para especificar el nombre de la voz de la salida de voz.
- Cree un objeto `TranslationRecognizer` con el objeto `SpeechTranslationConfig` anterior.
- Con el objeto `TranslationRecognizer`, inicie el proceso de reconocimiento de una única expresión.
- Inspeccione el objeto `TranslationRecognitionResult` devuelto.

Si prefiere ponerse a trabajar de inmediato, vea o descargue todos los [ejemplos para Python del SDK de Voz](#) en GitHub. Si no, vamos a comenzar.

## Prerequisites

Antes de comenzar, compruebe lo siguiente:

- [Ha creado un recurso de Voz de Azure](#)
- [Ha configurado el entorno de desarrollo](#)
- [Ha creado un proyecto de ejemplo vacío](#)

# Incorporación de código de ejemplo

1. Abra `quickstart.py` y reemplace todo el código que contiene por lo siguiente.

```
import azure.cognitiveservices.speech as speechsdk

speech_key, service_region = "YourSubscriptionKey", "YourServiceRegion"

def translate_speech_to_speech():

    # Creates an instance of a speech translation config with specified subscription key and service
    # region.
    # Replace with your own subscription key and service region (e.g., "westus").
    translation_config = speechsdk.translation.SpeechTranslationConfig(subscription=speech_key,
    region=service_region)

    # Sets source and target languages.
    # Replace with the languages of your choice, from list found here: https://aka.ms/speech/stt-languages
    fromLanguage = 'en-US'
    toLanguage = 'de'
    translation_config.speech_recognition_language = fromLanguage
    translation_config.add_target_language(toLanguage)

    # Sets the synthesis output voice name.
    # Replace with the languages of your choice, from list found here: https://aka.ms/speech/tts-languages
    translation_config.voice_name = "de-DE-Hedda"

    # Creates a translation recognizer using and audio file as input.
    recognizer = speechsdk.translation.TranslationRecognizer(translation_config=translation_config)

    # Prepare to handle the synthesized audio data.
    def synthesis_callback(evt):
        size = len(evt.result.audio)
        print('AUDIO SYNTHESIZED: {} byte(s) {}'.format(size, '(COMPLETED)' if size == 0 else ''))

    recognizer.synthesizing.connect(synthesis_callback)

    # Starts translation, and returns after a single utterance is recognized. The end of a
    # single utterance is determined by listening for silence at the end or until a maximum of 15
    # seconds of audio is processed. It returns the recognized text as well as the translation.
    # Note: Since recognize_once() returns only a single utterance, it is suitable only for single
    # shot recognition like command or query.
    # For long-running multi-utterance recognition, use start_continuous_recognition() instead.
    print("Say something...")
    result = recognizer.recognize_once()

    # Check the result
    if result.reason == speechsdk.ResultReason.TranslatedSpeech:
        print("RECOGNIZED '{}': {}".format(fromLanguage, result.text))
        print("TRANSLATED into {}: {}".format(toLanguage, result.translations['de']))
    elif result.reason == speechsdk.ResultReason.RecognizedSpeech:
        print("RECOGNIZED: {} (text could not be translated)".format(result.text))
    elif result.reason == speechsdk.ResultReason.NoMatch:
        print("NOMATCH: Speech could not be recognized: {}".format(result.no_match_details))
    elif result.reason == speechsdk.ResultReason.Canceled:
        print("CANCELED: Reason={}".format(result.cancellation_details.reason))
        if result.cancellation_details.reason == speechsdk.CancellationReason.Error:
            print("CANCELED: ErrorDetails={}".format(result.cancellation_details.error_details))

translate_speech_to_speech()
```

2. En el mismo archivo, reemplace la cadena `YourSubscriptionKey` por la clave de suscripción.

3. Reemplace la cadena `YourServiceRegion` por la **región** asociada a sus suscripción (por ejemplo, `westus`) para

la suscripción de evaluación gratuita).

4. Guarde los cambios realizados en `quickstart.py`.

## Compilación y ejecución de la aplicación

1. Ejecute el ejemplo desde la consola o en el IDE:

```
python quickstart.py
```

2. Diga una oración o frase en inglés. La aplicación transmite la voz al servicio de voz, que la traduce (a alemán en este caso) y la transcribe en texto. Después, el servicio de voz envía el audio sintetizado y el texto de nuevo a la aplicación para mostrarlo.

```
Say something...
AUDIO SYNTHESIZED: 76784 byte(s)
AUDIO SYNTHESIZED: 0 byte(s) (COMPLETE)
RECOGNIZED 'en-US': What's the weather in Seattle?
TRANSLATED into 'de': Wie ist das Wetter in Seattle?
```

## Pasos siguientes

[Exploración de los ejemplos de Python en GitHub](#)

Vea o descargue todos los [ejemplos del SDK de Voz](#) en GitHub.

## Compatibilidad con plataformas y lenguajes adicionales

Si ha hecho clic en esta pestaña, es probable que no vea un inicio rápido en su lenguaje de programación favorito. No se preocupe, tenemos materiales de inicio rápido y ejemplos de código adicionales disponibles en GitHub. Use la tabla para encontrar el ejemplo correcto para su lenguaje de programación y combinación de plataforma y sistema operativo.

IDIOMA	EJEMPLOS DE CÓDIGO
C++	<a href="#">Inicios rápidos</a> , <a href="#">Ejemplos</a>
C#	<a href="#">.NET Framework</a> , <a href="#">.NET Core</a> , <a href="#">UWP</a> , <a href="#">Unity</a> , <a href="#">Xamarin</a>
Java	<a href="#">Android</a> , <a href="#">JRE</a>
JavaScript	<a href="#">Browser</a>
Node.js	<a href="#">Windows</a> , <a href="#">Linux</a> , <a href="#">macOS</a>
Objective-C	<a href="#">iOS</a> , <a href="#">macOS</a>
Python	<a href="#">Windows</a> , <a href="#">Linux</a> , <a href="#">macOS</a>
Swift	<a href="#">iOS</a> , <a href="#">macOS</a>

# Tutorial: Compilación de una aplicación de Flask con Azure Cognitive Services

13/01/2020 • 40 minutes to read • [Edit Online](#)

En este tutorial, compilará una aplicación web de Flask que usa Azure Cognitive Services para traducir texto, realizar análisis de opinión y sintetizar texto traducido a voz. Aunque se prestará especial atención al código de Python y las rutas de Flask que habilitan nuestra aplicación, también se verá el código HTML y JavaScript que conforma la aplicación. Si experimenta algún problema, utilice el botón de comentarios de la parte inferior para informarnos.

Este tutorial abarca lo siguiente:

- Obtención de las claves de suscripción a Azure
- Configuración del entorno de desarrollo e instalación de las dependencias
- Creación de una aplicación de Flask
- Uso de Translator Text API para traducir texto
- Uso de Text Analytics para analizar opiniones positivas o negativas de texto de entrada y traducciones
- Uso de Speech Services para convertir texto traducido en voz sintetizada
- Ejecución local de la aplicación de Flask

## TIP

Si prefiere ver directamente todo el código, en [GitHub](#) se encuentra disponible el ejemplo completo junto con las instrucciones de compilación.

## ¿Qué es Flask?

Flask es un marco minimalista para crear aplicaciones web. Esto significa que Flask proporciona las herramientas, bibliotecas y tecnologías necesarias para compilar una aplicación web. Dicha aplicación web puede ser un conjunto de páginas web, un blog, una wiki o incluso una aplicación de calendario basada en web o un sitio web comercial.

Si desea obtener información detallada después de este tutorial, consulte estos vínculos útiles:

- [Documentación de Flask](#)
- [Guía de Flask para principiantes](#)

## Requisitos previos

Para este tutorial, se necesita el software y las claves de suscripción siguientes.

- [Python 3.5.2 o versiones posteriores](#)
- [Herramientas de Git](#)
- Un editor de texto o IDE, como [Visual Studio Code](#) o [Atom](#)
- [Chrome](#) o [Firefox](#)
- Una clave de suscripción de **Translator Text** (no es obligatorio seleccionar una región)
- Una clave de suscripción de **Text Analytics** en la región **Oeste de EE. UU.**
- Una clave de suscripción de **Speech Services** en la región **Oeste de EE. UU.**

# Creación de una cuenta y suscripción a recursos

Como se ha indicado anteriormente, se necesitarán tres claves de suscripción para este tutorial. Esto significa que necesita crear un recurso en su cuenta de Azure para:

- Translator Text
- Text Analytics
- Speech Services

Use [Creación de una cuenta de Cognitive Services en Azure Portal](#) para obtener instrucciones paso a paso para crear recursos.

## IMPORTANT

Para este tutorial, cree los recursos en la región Oeste de EE. UU. Si usa una región distinta, deberá ajustar la URL base en cada uno de los archivos de Python.

# Configuración del entorno de desarrollo

Antes de compilar la aplicación web de Flask, deberá crear un directorio de trabajo para el proyecto e instalar algunos paquetes de Python.

## Creación de un directorio de trabajo

1. Abra terminal (macOS o Linux) o la línea de comandos (Windows). A continuación, cree un directorio de trabajo y los subdirectorios para el proyecto:

```
mkdir -p flask-cog-services/static/scripts && mkdir flask-cog-services/templates
```

2. Cámbielo al directorio de trabajo del proyecto:

```
cd flask-cog-services
```

## Creación y activación de un entorno virtual con `virtualenv`

Ahora creará un entorno virtual para la aplicación de Flask con `virtualenv`. El uso de un entorno virtual garantiza que dispone de un entorno limpio desde el que trabajar.

1. En el directorio de trabajo, ejecute este comando para crear un entorno virtual: **macOS/Linux:**

```
virtualenv venv --python=python3
```

Se ha indicado explícitamente que el entorno virtual debe usar Python 3. Esto permite garantizar que los usuarios con varias instalaciones de Python usen la versión correcta.

## CMD de Windows/Bash de Windows:

```
virtualenv venv
```

Para simplificar las cosas, se denominará al entorno virtual `venv`.

2. Los comandos para activar el entorno virtual variarán según la plataforma o shell:

PLATAFORMA	SHELL	GET-HELP
macOS/Linux	bash/zsh	<code>source venv/bin/activate</code>
Windows	Bash	<code>source venv/Scripts/activate</code>
	Línea de comandos	<code>venv\Scripts\activate.bat</code>
	PowerShell	<code>venv\Scripts\Activate.ps1</code>

Después de ejecutar este comando, la sesión de línea de comandos o terminal debe ir precedida por `venv`.

3. Puede desactivar la sesión en cualquier momento escribiendo lo siguiente en la línea de comandos o terminal: `deactivate`.

#### NOTE

Python cuenta con una amplia documentación para crear y administrar entornos virtuales; consulte [virtualenv](#).

## Instalación de Requests

Requests es un módulo popular que se usa para enviar solicitudes HTTP 1.1. No es necesario agregar manualmente cadenas de consulta a las direcciones URL ni formar y codificar los datos POST.

1. Para instalar Requests, ejecute lo siguiente:

```
pip install requests
```

#### NOTE

Si desea más información sobre Requests, consulte [Requests: HTTP for Humans](#) (Requests: HTTP para humanos).

## Instalación y configuración de Flask

A continuación, se debe instalar Flask. Flask controla el enrutamiento de nuestra aplicación web y permite realizar llamadas de servidor a servidor que ocultan nuestras claves de suscripción al usuario final.

1. Para instalar Flask, ejecute lo siguiente:

```
pip install Flask
```

Asegúrese de que se haya instalado Flask. Ejecutar:

```
flask --version
```

La versión se debe imprimir en terminal. Si no es así, algo no ha funcionado.

2. Para ejecutar la aplicación de Flask, puede usar el comando flask o modificador -m de Python con Flask. Antes de eso, es necesario indicar a terminal con qué aplicación trabajar mediante la exportación de la variable de entorno `FLASK_APP`:

**macOS/Linux:**

```
export FLASK_APP=app.py
```

## Windows:

```
set FLASK_APP=app.py
```

# Creación de la aplicación de Flask

En esta sección, va a crear una aplicación de Flask esencial que devuelve un archivo HTML cuando los usuarios llegan a la raíz de la aplicación. No dedique demasiado tiempo a analizar en detalle el código, ya que más adelante regresaremos sobre este archivo para actualizarlo.

## ¿Qué es una ruta de Flask?

A continuación, se explica brevemente en qué consisten las "rutas". El enrutamiento se usa para enlazar una dirección URL a una función específica. Flask usa elementos Decorator de ruta para registrar funciones en direcciones URL específicas. Por ejemplo, cuando un usuario navega a la raíz (/) de nuestra aplicación web, se representa index.html .

```
@app.route('/')
def index():
    return render_template('index.html')
```

Veamos otro ejemplo para dejarlo claro.

```
@app.route('/about')
def about():
    return render_template('about.html')
```

Este código garantiza que cuando un usuario navega a http://your-web-app.com/about , se representa el archivo about.html .

Aunque estos ejemplos ilustran cómo representar páginas HTML para un usuario, también se pueden usar rutas para llamar a API al presionar un botón o para realizar una serie de acciones sin necesidad de salir de la página principal. Cuando cree rutas para traducción, opinión y síntesis de voz verá un ejemplo de esto en acción.

## Primeros pasos

1. Abra el proyecto en el entorno de desarrollo integrado y, después, cree un archivo denominado app.py en la raíz de su directorio de trabajo. A continuación, copie este código en app.py y guarde:

```
from flask import Flask, render_template, url_for, jsonify, request

app = Flask(__name__)
app.config['JSON_AS_ASCII'] = False

@app.route('/')
def index():
    return render_template('index.html')
```

Este bloque de código indica a la aplicación que muestre index.html cada vez que un usuario navega a la raíz de la aplicación web (/).

2. A continuación, vamos a crear la interfaz de usuario de la aplicación web. Cree un archivo denominado index.html en el directorio templates . A continuación, copie este código en templates/index.html .

```

<!doctype html>
<html lang="en">
  <head>
    <!-- Required metadata tags -->
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
    <meta name="description" content="Translate and analyze text with Azure Cognitive Services.">
    <!-- Bootstrap CSS -->
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"
integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm"
crossorigin="anonymous">
    <title>Translate and analyze text with Azure Cognitive Services</title>
  </head>
  <body>
    <div class="container">
      <h1>Translate, synthesize, and analyze text with Azure</h1>
      <p>This simple web app uses Azure for text translation, text-to-speech conversion, and sentiment analysis of input text and translations. Learn more about <a href="https://docs.microsoft.com/azure/cognitive-services/">Azure Cognitive Services</a>.</p>
      <!-- HTML provided in the following sections goes here. -->
      <!-- End -->
    </div>

    <!-- Required Javascript for this tutorial -->
    <script src="https://code.jquery.com/jquery-3.2.1.slim.min.js" integrity="sha384-KJ3o2DKtIkVYIK3UENzmM7KCkRr/rE9/Qpg6aAZGJwFDMVNA/GpGFF93hXpG5KkN" crossorigin="anonymous"></script>
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.js"
integrity="sha384-ApNbgh9B+y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvfa0b4Q"
crossorigin="anonymous"></script>
    <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js" integrity="sha384-JZR6Spejh4U02d8j0t6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVCmYl"
crossorigin="anonymous"></script>
    <script type="text/javascript" src="static/scripts/main.js"></script>
  </body>
</html>

```

3. Vamos a probar la aplicación de Flask. Desde terminal, ejecute:

```
flask run
```

4. Abra un explorador y vaya a la dirección URL proporcionada. Deberá ver la aplicación de página única.

Presione **Ctrl + C** para terminar la aplicación.

## Traducir texto

Ahora que tiene una idea de cómo funciona una aplicación sencilla de Flask, realizará lo siguiente:

- Escribir código Python para llamar a Translator Text API y devolver una respuesta
- Crear una ruta de Flask para llamar a su código Python
- Actualizar el código HTML con un área de entrada de texto y traducción, un selector de idioma y un botón de traducción
- Escribir código JavaScript que permita a los usuarios interactuar con la aplicación de Flask desde el código HTML

### Llamar a Translator Text API

Lo primero que necesita hacer es escribir una función para llamar a Translator Text API. Esta función tendrá dos argumentos: `text_input` y `language_output`. Se llama a esta función cada vez que un usuario presiona el botón de traducción de la aplicación. El área de texto del código HTML se envía como `text_input`, y el valor de selección de

idioma del código HTML se envía como `language_output`.

1. Para empezar, cree un archivo denominado `translate.py` en la raíz de su directorio de trabajo.
2. A continuación, agregue este código a `translate.py`. Esta función tiene dos argumentos: `text_input` y `language_output`.

```
import os, requests, uuid, json

# Don't forget to replace with your Cog Services subscription key!
# If you prefer to use environment variables, see Extra Credit for more info.
subscription_key = 'YOUR_TRANSLATOR_TEXT_SUBSCRIPTION_KEY'

# Don't forget to replace with your Cog Services location!
# Our Flask route will supply two arguments: text_input and language_output.
# When the translate text button is pressed in our Flask app, the Ajax request
# will grab these values from our web app, and use them in the request.
# See main.js for Ajax calls.

def get_translation(text_input, language_output):
    base_url = 'https://api.cognitive.microsofttranslator.com'
    path = '/translate?api-version=3.0'
    params = '&to=' + language_output
    constructed_url = base_url + path + params

    headers = {
        'Ocp-Apim-Subscription-Key': subscription_key,
        'Ocp-Apim-Subscription-Region': 'location',
        'Content-type': 'application/json',
        'X-ClientTraceId': str(uuid.uuid4())
    }

    # You can pass more than one object in body.
    body = [{ 
        'text' : text_input
    }]
    response = requests.post(constructed_url, headers=headers, json=body)
    return response.json()
```

3. Agregue su clave de suscripción de Translator Text y guarde.

### Agregar una ruta a `app.py`

A continuación, deberá crear una ruta en la aplicación de Flask que llame a `translate.py`. Se llamará a esta ruta cada vez que un usuario presione el botón de traducción en la aplicación.

Para esta aplicación, la ruta aceptará solicitudes `POST`. Esto se debe a que la función espera el texto que traducir y un idioma de salida para la traducción.

Flask proporciona funciones auxiliares para ayudarle a analizar y administrar cada solicitud. En el código proporcionado, `get_json()` devuelve los datos de la solicitud `POST` como JSON. A continuación, mediante `data['text']` y `data['to']`, los valores de texto e idioma de salida se pasan a la función `get_translation()` disponible desde `translate.py`. El último paso es devolver la respuesta como JSON, ya que necesitará mostrar estos datos en la aplicación web.

En las secciones siguientes, deberá repetir este proceso a medida que cree rutas para la síntesis de voz y el análisis de opinión.

1. Abra `app.py` y busque la instrucción de importación en la parte superior de `app.py` y agregue la siguiente línea:

```
import translate
```

Ahora la aplicación de Flask puede usar el método disponible mediante `translate.py`.

2. Copie este código al final de `app.py` y guarde:

```
@app.route('/translate-text', methods=['POST'])
def translate_text():
    data = request.get_json()
    text_input = data['text']
    translation_output = data['to']
    response = translate.get_translation(text_input, translation_output)
    return jsonify(response)
```

**Actualizar** `index.html`

Ahora que tiene una función para traducir texto y una ruta en la aplicación de Flask para llamarla, el siguiente paso es empezar a compilar el código HTML de la aplicación. El código HTML siguiente realiza varias cosas:

- Proporciona un área de texto donde los usuarios pueden escribir el texto que se traducirá.
- Incluye un selector de idioma.
- Incluye los elementos HTML para representar el idioma detectado y las puntuaciones de confianza que se devuelven durante la traducción.
- Proporciona un área de texto de solo lectura donde se muestra la salida de traducción.
- Incluye marcadores de posición para análisis de opinión y síntesis de voz que agregarán a este archivo más adelante en el tutorial.

Ahora actualizaremos `index.html`.

1. Abra `index.html` y busque estos comentarios en el código:

```
<!-- HTML provided in the following sections goes here. -->
<!-- End -->
```

2. Reemplace los comentarios del código por este bloque HTML:

```

<div class="row">
  <div class="col">
    <form>
      <!-- Enter text to translate. -->
      <div class="form-group">
        <label for="text-to-translate"><strong>Enter the text you'd like to translate:</strong></label>
        <textarea class="form-control" id="text-to-translate" rows="5"></textarea>
      </div>
      <!-- Select output language. -->
      <div class="form-group">
        <label for="select-language"><strong>Translate to:</strong></label>
        <select class="form-control" id="select-language">
          <option value="ar">Arabic</option>
          <option value="ca">Catalan</option>
          <option value="zh-Hans">Chinese (Simplified)</option>
          <option value="zh-Hant">Chinese (Traditional)</option>
          <option value="hr">Croatian</option>
          <option value="en">English</option>
          <option value="fr">French</option>
          <option value="de">German</option>
          <option value="el">Greek</option>
          <option value="he">Hebrew</option>
          <option value="hi">Hindi</option>
          <option value="it">Italian</option>
          <option value="ja">Japanese</option>
          <option value="ko">Korean</option>
          <option value="pt">Portuguese</option>
          <option value="ru">Russian</option>
          <option value="es">Spanish</option>
          <option value="th">Thai</option>
          <option value="tr">Turkish</option>
          <option value="vi">Vietnamese</option>
        </select>
      </div>
      <button type="submit" class="btn btn-primary mb-2" id="translate">Translate text</button><br>
      <div id="detected-language" style="display: none">
        <strong>Detected language:</strong> <span id="detected-language-result"></span><br />
        <strong>Detection confidence:</strong> <span id="confidence"></span><br /><br />
      </div>

      <!-- Start sentiment code-->
      <!-- End sentiment code -->
    </form>
  </div>
  <div class="col">
    <!-- Translated text returned by the Translate API is rendered here. -->
    <form>
      <div class="form-group" id="translator-text-response">
        <label for="translation-result"><strong>Translated text:</strong></label>
        <textarea readonly class="form-control" id="translation-result" rows="5"></textarea>
      </div>
      <!-- Start voice font selection code -->
      <!-- End voice font selection code -->
    </form>
    <!-- Add Speech Synthesis button and audio element -->
    <!-- End Speech Synthesis button -->
  </div>
</div>

```

El siguiente paso es escribir código JavaScript. Este es el puente entre el código HTML y la ruta de Flask.

### Cree `main.js`

El archivo `main.js` es el puente entre el código HTML y la ruta de Flask. La aplicación usará una combinación de jQuery, Ajax y XMLHttpRequest para representar el contenido y realizar solicitudes `POST` a las rutas de Flask.

En el código siguiente, el contenido del código HTML se usa para construir una solicitud a la ruta de Flask. En concreto, el contenido del área de texto y el selector de idioma se asigna a variables y, a continuación, se pasa en la solicitud a `translate-text`.

El código procesa una iteración en la respuesta y actualiza el código HTML con la traducción, el idioma detectado y la puntuación de confianza.

1. Desde el entorno de desarrollo integrado, cree un archivo denominado `main.js` en el directorio

`static/scripts`.

2. Copie este código en `static/scripts/main.js`:

```
//Initiate jQuery on load.
$(function() {
    //Translate text with flask route
    $("#translate").on("click", function(e) {
        e.preventDefault();
        var translateVal = document.getElementById("text-to-translate").value;
        var languageVal = document.getElementById("select-language").value;
        var translateRequest = { 'text': translateVal, 'to': languageVal }

        if (translateVal !== "") {
            $.ajax({
                url: '/translate-text',
                method: 'POST',
                headers: {
                    'Content-Type': 'application/json'
                },
                dataType: 'json',
                data: JSON.stringify(translateRequest),
                success: function(data) {
                    for (var i = 0; i < data.length; i++) {
                        document.getElementById("translation-result").textContent = data[i].translations[0].text;
                        document.getElementById("detected-language-result").textContent =
                            data[i].detectedLanguage.language;
                        if (document.getElementById("detected-language-result").textContent !== ""){
                            document.getElementById("detected-language").style.display = "block";
                        }
                        document.getElementById("confidence").textContent = data[i].detectedLanguage.score;
                    }
                }
            });
        }
    });
});
// In the following sections, you'll add code for sentiment analysis and
// speech synthesis here.
})
```

### Probar la traducción

Ahora probaremos la función de traducción de la aplicación.

```
flask run
```

Vaya a la dirección del servidor proporcionada. Escriba texto en el área de entrada, seleccione un idioma y presione el botón de traducción. Deberá obtener una traducción. Si no funciona, asegúrese de que ha agregado su clave de

suscripción.

#### TIP

Si no se muestran los cambios realizados o la aplicación no funciona de la forma esperada, pruebe a borrar la caché o a abrir una ventana privada o de incógnito.

Presione **CTRL + C** para terminar la aplicación y, después, vaya a la sección siguiente.

## Análisis de opinión

Text Analytics API puede usarse para realizar análisis de opinión, extraer frases clave del texto o detectar el idioma de origen. En esta aplicación, vamos a usar análisis de opinión para determinar si el texto proporcionado es negativo, neutral o positivo. La API devuelve una puntuación numérica entre 0 y 1. Las puntuaciones próximas a 1 indican una opinión positiva y las puntuaciones próximas a 0 indican una opinión negativa.

En esta sección, realizará lo siguiente:

- Escribir código Python para llamar a Text Analytics API para realizar análisis de opinión y devolver una respuesta
- Crear una ruta de Flask para llamar a su código Python
- Actualizar el código HTML con un área para las puntuaciones de opinión y un botón para realizar el análisis
- Escribir código JavaScript que permita a los usuarios interactuar con la aplicación de Flask desde el código HTML

### Llamada a la API de Text Analytics

Ahora escribirá una función para llamar a Text Analytics API. Esta función tendrá cuatro argumentos: `input_text`, `input_language`, `output_text` y `output_language`. Se llama a esta función cada vez que un usuario presiona el botón de ejecución de análisis de opinión de la aplicación. Con cada solicitud se proporcionan los datos especificados por el usuario en el área de texto y el selector de idioma, así como el idioma detectado y la salida de traducción. El objeto de respuesta incluye las puntuaciones de opinión para el origen y la traducción. En las secciones siguientes, escribirá código JavaScript para analizar la respuesta y usarlo en la aplicación. Por ahora, nos centraremos en la llamada a Text Analytics API.

1. Cree un archivo denominado `sentiment.py` en la raíz de su directorio de trabajo.
2. A continuación, agregue este código a `sentiment.py`.

```

import os, requests, uuid, json

# Don't forget to replace with your Cog Services subscription key!
subscription_key = 'YOUR_TEXT_ANALYTICS_SUBSCRIPTION_KEY'

# Our Flask route will supply four arguments: input_text, input_language,
# output_text, output_language.
# When the run sentiment analysis button is pressed in our Flask app,
# the Ajax request will grab these values from our web app, and use them
# in the request. See main.js for Ajax calls.

def get_sentiment(input_text, input_language, output_text, output_language):
    base_url = 'https://westus.api.cognitive.microsoft.com/text/analytics'
    path = '/v2.0/sentiment'
    constructed_url = base_url + path

    headers = {
        'Ocp-Apim-Subscription-Key': subscription_key,
        'Content-type': 'application/json',
        'X-ClientTraceId': str(uuid.uuid4())
    }

    # You can pass more than one object in body.
    body = {
        'documents': [
            {
                'language': input_language,
                'id': '1',
                'text': input_text
            },
            {
                'language': output_language,
                'id': '2',
                'text': output_text
            }
        ]
    }
    response = requests.post(constructed_url, headers=headers, json=body)
    return response.json()

```

### 3. Agregue su clave de suscripción de Text Analytics y guarde.

#### Agregar una ruta a `app.py`

Ahora creará una ruta en la aplicación de Flask que llame a `sentiment.py`. Se llamará a esta ruta cada vez que un usuario presione el botón de ejecución de análisis de opinión de la aplicación. Al igual que la ruta para traducción, esta ruta aceptará solicitudes `POST`, ya que la función espera argumentos.

#### 1. Abra `app.py`, busque la instrucción de importación en la parte superior de `app.py` y actualícela:

```
import translate, sentiment
```

Ahora la aplicación de Flask puede usar el método disponible mediante `sentiment.py`.

#### 2. Copie este código al final de `app.py` y guarde:

```
@app.route('/sentiment-analysis', methods=['POST'])
def sentiment_analysis():
    data = request.get_json()
    input_text = data['inputText']
    input_lang = data['inputLanguage']
    output_text = data['outputText']
    output_lang = data['outputLanguage']
    response = sentiment.get_sentiment(input_text, input_lang, output_text, output_lang)
    return jsonify(response)
```

## Actualizar `index.html`

Ahora que tiene una función para ejecutar análisis de opinión y una ruta en la aplicación de Flask para llamarla, el siguiente paso es empezar a escribir el código HTML de la aplicación. El código HTML siguiente realiza varias cosas:

- Agrega un botón a la aplicación para ejecutar análisis de opinión
- Agrega un elemento que explica la puntuación de opiniones
- Agrega un elemento que muestra las puntuaciones de opinión

1. Abra `index.html` y busque estos comentarios en el código:

```
<!-- Start sentiment code-->

<!-- End sentiment code -->
```

2. Reemplace los comentarios del código por este bloque HTML:

```
<button type="submit" class="btn btn-primary mb-2" id="sentiment-analysis">Run sentiment
analysis</button><br>
<div id="sentiment" style="display: none">
    <p>Sentiment scores are provided on a 1 point scale. The closer the sentiment score is to 1,
    indicates positive sentiment. The closer it is to 0, indicates negative sentiment.</p>
    <strong>Sentiment score for input:</strong> <span id="input-sentiment"></span><br />
    <strong>Sentiment score for translation:</strong> <span id="translation-sentiment"></span>
</div>
```

## Actualizar `main.js`

En el código siguiente, el contenido del código HTML se usa para construir una solicitud a la ruta de Flask. En concreto, el contenido del área de texto y el selector de idioma se asigna a variables y, a continuación, se pasa en la solicitud a la ruta `sentiment-analysis`.

El código procesa una iteración en la respuesta y actualiza el código HTML con las puntuaciones de opinión.

1. Desde el entorno de desarrollo integrado, cree un archivo denominado `main.js` en el directorio `static`.
2. Copie este código en `static/scripts/main.js`:

```

//Run sentiment analysis on input and translation.
$("#sentiment-analysis").on("click", function(e) {
  e.preventDefault();
  var inputText = document.getElementById("text-to-translate").value;
  var inputLanguage = document.getElementById("detected-language-result").innerHTML;
  var outputText = document.getElementById("translation-result").value;
  var outputLanguage = document.getElementById("select-language").value;

  var sentimentRequest = { "inputText": inputText, "inputLanguage": inputLanguage, "outputText": outputText, "outputLanguage": outputLanguage };

  if (inputText !== "") {
    $.ajax({
      url: "/sentiment-analysis",
      method: "POST",
      headers: {
        "Content-Type": "application/json"
      },
      dataType: "json",
      data: JSON.stringify(sentimentRequest),
      success: function(data) {
        for (var i = 0; i < data.documents.length; i++) {
          if (typeof data.documents[i] !== "undefined"){
            if (data.documents[i].id === "1") {
              document.getElementById("input-sentiment").textContent = data.documents[i].score;
            }
            if (data.documents[i].id === "2") {
              document.getElementById("translation-sentiment").textContent = data.documents[i].score;
            }
          }
        }
        for (var i = 0; i < data.errors.length; i++) {
          if (typeof data.errors[i] !== "undefined"){
            if (data.errors[i].id === "1") {
              document.getElementById("input-sentiment").textContent = data.errors[i].message;
            }
            if (data.errors[i].id === "2") {
              document.getElementById("translation-sentiment").textContent = data.errors[i].message;
            }
          }
        }
        if (document.getElementById("input-sentiment").textContent !== '' &&
        document.getElementById("translation-sentiment").textContent !== ""){
          document.getElementById("sentiment").style.display = "block";
        }
      }
    });
  }
});
// In the next section, you'll add code for speech synthesis here.

```

## Probar el análisis de opinión

Ahora probará el análisis de opinión en la aplicación.

```
flask run
```

Vaya a la dirección del servidor proporcionada. Escriba texto en el área de entrada, seleccione un idioma y presione el botón de traducción. Deberá obtener una traducción. A continuación, presione el botón de ejecución de análisis de opinión. Deberá ver dos puntuaciones. Si no funciona, asegúrese de que ha agregado su clave de suscripción.

#### TIP

Si no se muestran los cambios realizados o la aplicación no funciona de la forma esperada, pruebe a borrar la caché o a abrir una ventana privada o de incógnito.

Presione **CTRL + C** para terminar la aplicación y, después, vaya a la sección siguiente.

## Conversión de texto a voz

[Text-to-Speech API](#) permite a la aplicación convertir texto en voz sintetizada natural similar a la humana. El servicio admite voces neuronales, estándares y personalizadas. Nuestra aplicación de ejemplo usa varias de las voces disponibles; para ver una lista completa, consulte los [idiomas admitidos](#).

En esta sección, realizará lo siguiente:

- Escribir código Python para convertir texto en voz con Text-to-Speech API
- Crear una ruta de Flask para llamar a su código Python
- Actualizar el código HTML con un botón para convertir texto en voz y un elemento para la reproducción de audio
- Escribir código JavaScript que permita a los usuarios interactuar con la aplicación de Flask

### Llamar a Text-to-Speech API

Ahora escribirá una función para convertir texto en voz. Esta función tendrá dos argumentos: `input_text` y `voice_font`. Se llama a esta función cada vez que un usuario presiona el botón de convertir texto en voz de la aplicación. `input_text` es la salida de traducción devuelta por la llamada para traducir texto, `voice_font` es el valor del selector de fuente de voz en el código HTML.

1. Cree un archivo denominado `synthesize.py` en la raíz de su directorio de trabajo.
2. A continuación, agregue este código a `synthesize.py`.

```

import os, requests, time
from xml.etree import ElementTree

class TextToSpeech(object):
    def __init__(self, input_text, voice_font):
        subscription_key = 'YOUR_SPEECH_SERVICES_SUBSCRIPTION_KEY'
        self.subscription_key = subscription_key
        self.input_text = input_text
        self.voice_font = voice_font
        self.timestr = time.strftime('%Y%m%d-%H%M')
        self.access_token = None

    # This function performs the token exchange.
    def get_token(self):
        fetch_token_url = 'https://westus.api.cognitive.microsoft.com/sts/v1.0/issueToken'
        headers = {
            'Ocp-Apim-Subscription-Key': self.subscription_key
        }
        response = requests.post(fetch_token_url, headers=headers)
        self.access_token = str(response.text)

    # This function calls the TTS endpoint with the access token.
    def save_audio(self):
        base_url = 'https://westus.tts.speech.microsoft.com/'
        path = 'cognitiveservices/v1'
        constructed_url = base_url + path
        headers = {
            'Authorization': 'Bearer ' + self.access_token,
            'Content-Type': 'application/ssml+xml',
            'X-Microsoft-OutputFormat': 'riff-24kHz-16bit-mono-pcm',
            'User-Agent': 'YOUR_RESOURCE_NAME',
        }
        # Build the SSML request with ElementTree
        xml_body = ElementTree.Element('speak', version='1.0')
        xml_body.set('{http://www.w3.org/XML/1998/namespace}lang', 'en-us')
        voice = ElementTree.SubElement(xml_body, 'voice')
        voice.set('{http://www.w3.org/XML/1998/namespace}lang', 'en-US')
        voice.set('name', 'Microsoft Server Speech Text to Speech Voice {}'.format(self.voice_font))
        voice.text = self.input_text
        # The body must be encoded as UTF-8 to handle non-ascii characters.
        body = ElementTree.tostring(xml_body, encoding="utf-8")

        #Send the request
        response = requests.post(constructed_url, headers=headers, data=body)

        # Write the response as a wav file for playback. The file is located
        # in the same directory where this sample is run.
        return response.content

```

### 3. Agregue la clave de suscripción de Speech Services y guarde.

#### Agregar una ruta a `app.py`

Ahora creará una ruta en la aplicación de Flask que llame a `synthesize.py`. Se llamará a esta ruta cada vez que un usuario presione el botón de convertir texto en voz de la aplicación. Al igual que las rutas de traducción y análisis de opinión, esta ruta aceptará solicitudes `POST`, ya que la función espera dos argumentos: el texto que sintetizar y la fuente de voz para la reproducción.

1. Abra `app.py`, busque la instrucción de importación en la parte superior de `app.py` y actualícela:

```
import translate, sentiment, synthesize
```

Ahora la aplicación de Flask puede usar el método disponible mediante `synthesize.py`.

2. Copie este código al final de `app.py` y guarde:

```
@app.route('/text-to-speech', methods=['POST'])
def text_to_speech():
    data = request.get_json()
    text_input = data['text']
    voice_font = data['voice']
    tts = synthesize.TextToSpeech(text_input, voice_font)
    tts.get_token()
    audio_response = tts.save_audio()
    return audio_response
```

**Actualizar** `index.html`

Ahora que tiene una función para convertir texto en voz y una ruta en la aplicación de Flask para llamarla, el siguiente paso es empezar a escribir el código HTML de la aplicación. El código HTML siguiente realiza varias cosas:

- Proporciona una lista desplegable de selección de voz
- Agrega un botón para convertir texto a voz
- Agrega un elemento de audio, que se usa para reproducir la voz sintetizada

1. Abra `index.html` y busque estos comentarios en el código:

```
<!-- Start voice font selection code -->

<!-- End voice font selection code -->
```

2. Reemplace los comentarios del código por este bloque HTML:

```

<div class="form-group">
  <label for="select-voice"><strong>Select voice font:</strong></label>
  <select class="form-control" id="select-voice">
    <option value="(ar-SA, Naayf)">Arabic | Male | Naayf</option>
    <option value="(ca-ES, HerenaRUS)">Catalan | Female | HerenaRUS</option>
    <option value="(zh-CN, HuihuiRUS)">Chinese (Mainland) | Female | HuihuiRUS</option>
    <option value="(zh-CN, Kangkang, Apollo)">Chinese (Mainland) | Male | Kangkang, Apollo</option>
    <option value="(zh-HK, Tracy, Apollo)">Chinese (Hong Kong) | Female | Tracy, Apollo</option>
    <option value="(zh-HK, Danny, Apollo)">Chinese (Hong Kong) | Male | Danny, Apollo</option>
    <option value="(zh-TW, Yating, Apollo)">Chinese (Taiwan) | Female | Yaiting, Apollo</option>
    <option value="(zh-TW, Zhiwei, Apollo)">Chinese (Taiwan) | Male | Zhiwei, Apollo</option>
    <option value="(hr-HR, Matej)">Croatian | Male | Matej</option>
    <option value="(en-US, Jessa24kRUS)">English (US) | Female | Jessa24kRUS</option>
    <option value="(en-US, Guy24kRUS)">English (US) | Male | Guy24kRUS</option>
    <option value="(en-IE, Sean)">English (IE) | Male | Sean</option>
    <option value="(fr-FR, Julie, Apollo)">French | Female | Julie, Apollo</option>
    <option value="(fr-FR, HortenseRUS)">French | Female | Julie, HortenseRUS</option>
    <option value="(fr-FR, Paul, Apollo)">French | Male | Paul, Apollo</option>
    <option value="(de-DE, Hedda)">German | Female | Hedda</option>
    <option value="(de-DE, HeddaRUS)">German | Female | HeddaRUS</option>
    <option value="(de-DE, Stefan, Apollo)">German | Male | Apollo</option>
    <option value="(el-GR, Stefanos)">Greek | Male | Stefanos</option>
    <option value="(he-IL, Asaf)">Hebrew (Isreal) | Male | Asaf</option>
    <option value="(hi-IN, Kalpana, Apollo)">Hindi | Female | Kalpana, Apollo</option>
    <option value="(hi-IN, Hemant)">Hindi | Male | Hemant</option>
    <option value="(it-IT, LuciaRUS)">Italian | Female | LuciaRUS</option>
    <option value="(it-IT, Cosimo, Apollo)">Italian | Male | Cosimo, Apollo</option>
    <option value="(ja-JP, Ichiro, Apollo)">Japanese | Male | Ichiro</option>
    <option value="(ja-JP, HarukaRUS)">Japanese | Female | HarukaRUS</option>
    <option value="(ko-KR, HeamiRUS)">Korean | Female | Haemi</option>
    <option value="(pt-BR, HeloisaRUS)">Portuguese (Brazil) | Female | HeloisaRUS</option>
    <option value="(pt-BR, Daniel, Apollo)">Portuguese (Brazil) | Male | Daniel, Apollo</option>
    <option value="(pt-PT, HeliaRUS)">Portuguese (Portugal) | Female | HeliaRUS</option>
    <option value="(ru-RU, Irina, Apollo)">Russian | Female | Irina, Apollo</option>
    <option value="(ru-RU, Pavel, Apollo)">Russian | Male | Pavel, Apollo</option>
    <option value="(ru-RU, EkaterinaRUS)">Russian | Female | EkaterinaRUS</option>
    <option value="(es-ES, Laura, Apollo)">Spanish | Female | Laura, Apollo</option>
    <option value="(es-ES, HelenaRUS)">Spanish | Female | HelenaRUS</option>
    <option value="(es-ES, Pablo, Apollo)">Spanish | Male | Pablo, Apollo</option>
    <option value="(th-TH, Pattara)">Thai | Male | Pattara</option>
    <option value="(tr-TR, SedaRUS)">Turkish | Female | SedaRUS</option>
    <option value="(vi-VN, An)">Vietnamese | Male | An</option>
  </select>
</div>

```

3. A continuación, busque estos comentarios en el código:

```

<!-- Add Speech Synthesis button and audio element --&gt;

<!-- End Speech Synthesis button --&gt;
</pre>

```

4. Reemplace los comentarios del código por este bloque HTML:

```

<button type="submit" class="btn btn-primary mb-2" id="text-to-speech">Convert text-to-speech</button>
<div id="audio-playback">
  <audio id="audio" controls>
    <source id="audio-source" type="audio/mpeg" />
  </audio>
</div>

```

5. Asegúrese de guardar los cambios.

**Actualizar** `main.js`

En el código siguiente, el contenido del código HTML se usa para construir una solicitud a la ruta de Flask. En concreto, la traducción y la fuente de voz se asignan a variables y, a continuación, se pasan en la solicitud a la ruta `text-to-speech`.

El código procesa una iteración en la respuesta y actualiza el código HTML con las puntuaciones de opinión.

1. Desde el entorno de desarrollo integrado, cree un archivo denominado `main.js` en el directorio `static`.
2. Copie este código en `static/scripts/main.js`:

```
// Convert text-to-speech
$("#text-to-speech").on("click", function(e) {
  e.preventDefault();
  var ttsInput = document.getElementById("translation-result").value;
  var ttsVoice = document.getElementById("select-voice").value;
  var ttsRequest = { 'text': ttsInput, 'voice': ttsVoice }

  var xhr = new XMLHttpRequest();
  xhr.open("post", "/text-to-speech", true);
  xhr.setRequestHeader("Content-Type", "application/json");
  xhr.responseType = "blob";
  xhr.onload = function(evt){
    if (xhr.status === 200) {
      audioBlob = new Blob([xhr.response], {type: "audio/mpeg"});
      audioURL = URL.createObjectURL(audioBlob);
      if (audioURL.length > 5){
        var audio = document.getElementById("audio");
        var source = document.getElementById("audio-source");
        source.src = audioURL;
        audio.load();
        audio.play();
      }else{
        console.log("An error occurred getting and playing the audio.");
      }
    }
  }
  xhr.send(JSON.stringify(ttsRequest));
});
// Code for automatic language selection goes here.
```

3. Ya casi ha terminado. Lo último que debe hacer es agregar código a `main.js` para seleccionar automáticamente una fuente de voz según el idioma seleccionado para la traducción. Agregue este bloque de código a `main.js`:

```

// Automatic voice font selection based on translation output.
$('select[id="select-language"]').change(function(e) {
    if ($(this).val() == "ar"){
        document.getElementById("select-voice").value = "(ar-SA, Naayf)";
    }
    if ($(this).val() == "ca"){
        document.getElementById("select-voice").value = "(ca-ES, HerenaRUS)";
    }
    if ($(this).val() == "zh-Hans"){
        document.getElementById("select-voice").value = "(zh-HK, Tracy, Apollo)";
    }
    if ($(this).val() == "zh-Hant"){
        document.getElementById("select-voice").value = "(zh-HK, Tracy, Apollo)";
    }
    if ($(this).val() == "hr"){
        document.getElementById("select-voice").value = "(hr-HR, Matej)";
    }
    if ($(this).val() == "en"){
        document.getElementById("select-voice").value = "(en-US, Jessa24kRUS)";
    }
    if ($(this).val() == "fr"){
        document.getElementById("select-voice").value = "(fr-FR, HortenseRUS)";
    }
    if ($(this).val() == "de"){
        document.getElementById("select-voice").value = "(de-DE, HeddaRUS)";
    }
    if ($(this).val() == "el"){
        document.getElementById("select-voice").value = "(el-GR, Stefanos)";
    }
    if ($(this).val() == "he"){
        document.getElementById("select-voice").value = "(he-IL, Asaf)";
    }
    if ($(this).val() == "hi"){
        document.getElementById("select-voice").value = "(hi-IN, Kalpana, Apollo)";
    }
    if ($(this).val() == "it"){
        document.getElementById("select-voice").value = "(it-IT, LuciaRUS)";
    }
    if ($(this).val() == "ja"){
        document.getElementById("select-voice").value = "(ja-JP, HarukaRUS)";
    }
    if ($(this).val() == "ko"){
        document.getElementById("select-voice").value = "(ko-KR, HeamiRUS)";
    }
    if ($(this).val() == "pt"){
        document.getElementById("select-voice").value = "(pt-BR, HeloisaRUS)";
    }
    if ($(this).val() == "ru"){
        document.getElementById("select-voice").value = "(ru-RU, EkaterinaRUS)";
    }
    if ($(this).val() == "es"){
        document.getElementById("select-voice").value = "(es-ES, HelenaRUS)";
    }
    if ($(this).val() == "th"){
        document.getElementById("select-voice").value = "(th-TH, Pattara)";
    }
    if ($(this).val() == "tr"){
        document.getElementById("select-voice").value = "(tr-TR, SedaRUS)";
    }
    if ($(this).val() == "vi"){
        document.getElementById("select-voice").value = "(vi-VN, An)";
    }
});

```

## Prueba de la aplicación

Ahora probará la síntesis de voz en la aplicación.

```
flask run
```

Vaya a la dirección del servidor proporcionada. Escriba texto en el área de entrada, seleccione un idioma y presione el botón de traducción. Deberá obtener una traducción. A continuación, seleccione una voz y presione el botón de convertir texto en voz. La traducción debe reproducirse como voz sintetizada. Si no funciona, asegúrese de que ha agregado su clave de suscripción.

**TIP**

Si no se muestran los cambios realizados o la aplicación no funciona de la forma esperada, pruebe a borrar la caché o a abrir una ventana privada o de incógnito.

Eso es todo. Ya tiene una aplicación funcional que realiza traducciones, analiza opiniones y genera voz sintetizada. Presione **Ctrl + C** para terminar la aplicación. No olvide consultar los demás recursos de [Azure Cognitive Services](#).

## Obtención del código fuente

El código fuente de este proyecto está disponible en [GitHub](#).

## Pasos siguientes

- [Referencia de Translator Text API](#)
- [Referencia de Text Analytics API](#)
- [Referencia de Text-to-speech API](#)

# Notas de la versión

15/01/2020 • 33 minutes to read • [Edit Online](#)

## SDK de Voz 1.8.0: Versión de noviembre de 2019

### Nuevas características

- Se ha agregado una API `FromHost()` para facilitar su uso con contenedores locales y nubes soberanas.
- Se ha agregado la detección automática de idioma de origen para el reconocimiento de voz (en Java y C++)
- Se ha agregado el objeto `SourceLanguageConfig` para el reconocimiento de voz, que se usa para especificar los idiomas de origen esperados (en Java y C++).
- Se ha agregado compatibilidad con `KeywordRecognizer` en Windows (UWP), Android e iOS mediante los paquetes de Nuget y Unity.
- Se ha agregado la API de Java de conversación remota para realizar la transcripción de conversaciones en lotes asíncronos.

### Cambios importantes

- Las funcionalidades de transcripción de conversaciones se han movido al espacio de nombres `Microsoft.CognitiveServices.Speech.Transcription`.
- Parte de los métodos de transcripción de conversaciones se han movido a la nueva clase `Conversation`.
- Compatibilidad eliminada para iOS de 32 bits (ARMv7 y x86)

### Correcciones de errores

- Se ha corregido un bloqueo si se usa `KeywordRecognizer` local sin una clave de suscripción válida al servicio de voz.

### Muestras

- Ejemplo de Xamarin para `KeywordRecognizer`
- Ejemplo de Unity para `KeywordRecognizer`
- Ejemplos de C++ y Java de detección automática de idioma de origen

## SDK de voz 1.7.0: versión de septiembre de 2019

### Nuevas características

- Compatibilidad con la versión beta agregada para Xamarin en la Plataforma universal de Windows (UWP), Android e iOS
- Compatibilidad con iOS agregada para Unity
- Se ha agregado compatibilidad con entradas `Compressed` para ALaw, Mulaw, FLAC en Android, iOS y Linux.
- Se ha agregado `SendMessageAsync` en la clase `Connection` para enviar un mensaje al servicio.
- Se ha agregado `SetMessageProperty` en la clase `Connection` para establecer la propiedad de un mensaje.
- TTS agregó compatibilidad para Java (JRE y Android), Python, Swift y Objective-C
- TTS agregó compatibilidad de reproducción para macOS, iOS y Android
- Se ha agregado información de "límite de palabras" para TTS

### Correcciones de errores

- Se ha corregido un problema de compilación de IL2CPP en Unity 2019 para Android

- Se ha corregido un problema con los encabezados con formato incorrecto en la entrada de archivo WAV que se procesa de forma incorrecta
- Se ha corregido un problema con UUID que no es único en algunas propiedades de conexión
- Se han corregido algunas advertencias sobre los especificadores de nulabilidad en los enlaces SWIFT (puede que se requieran pequeños cambios en el código)
- Se ha corregido un error que provocaba que las conexiones de WebSocket se cerraran de manera incorrecta en la carga de red
- Se ha corregido un problema en Android que a veces provoca que `DialogServiceConnector` use identificadores de impresión duplicados.
- Se han introducido mejoras en la estabilidad de las conexiones entre interacciones multiproceso y la generación de informes de errores (a través de eventos `Canceled`) cuando se producen con `DialogServiceConnector`.
- Los inicios de sesión de `DialogServiceConnector` ahora proporcionarán eventos correctamente, incluso si se llama a `ListenOnceAsync()` durante una operación `StartKeywordRecognitionAsync()` activa.
- Se ha resuelto un bloqueo asociado a la recepción de actividades `DialogServiceConnector`.

## Muestras

- Inicio rápido para Xamarin
- Se ha actualizado el inicio rápido de CPP con información de ARM64 de Linux
- Se ha actualizado el inicio rápido de Unity con información de iOS

# SDK de Voz 1.6.0: versión de junio de 2019

## Muestras

- Ejemplos de inicio rápido para Texto a voz en UWP y Unity
- Ejemplo de inicio rápido para Swift en iOS
- Ejemplos de Unity para Traducción y Reconocimiento de la intención comunicativa y Voz
- Ejemplos de inicios rápidos actualizados para `DialogServiceConnector`

## Mejoras y cambios

- Espacio de nombres de cuadro de diálogo:
  - El nombre de `SpeechBotConnector` ha cambiado a `DialogServiceConnector`
  - El nombre de `BotConfig` ha cambiado a `DialogServiceConfig`
  - `BotConfig::FromChannelSecret()` se ha reasignado a `DialogServiceConfig::FromBotSecret()`
  - Todos los clientes de Voz de Direct Line existentes siguen siendo compatibles después del cambio de nombre
- Actualización del adaptador REST de TTS para admitir una conexión persistente de proxy
- Un mejor mensaje de error cuando se pasa una región no válida
- Swift/Objective-C:
  - Mejores informes de errores: los métodos que pueden generar un error ahora se encuentran en dos versiones: una que expone un objeto `NSError` para el control de errores y una que genera una excepción. La primera se expone a Swift. Este cambio requiere adaptaciones en el código Swift existente.
  - Mejor control de eventos

## Correcciones de errores

- Corrección de TTS: donde el futuro de `SpeakTextAsync` se devolvió sin esperar al fin de la representación del audio
- Corrección para la serialización de las cadenas en C# para permitir la compatibilidad total con idiomas
- Corrección del problema de las aplicaciones centrales de .NET para cargar la biblioteca principal con un marco

de destino net461 en ejemplos

- Corrección de problemas ocasionales para implementar bibliotecas nativas en la carpeta de salida en los ejemplos
- Corrección para cerrar el socket web de manera confiable
- Corrección de un posible bloqueo al abrir una conexión con una carga muy elevada en Linux
- Corrección de metadatos faltantes en el paquete de marcos para macOS
- Corrección de problemas con `pip install --user` en Windows

## Speech SDK 1.5.1

Se trata de una versión de corrección de errores y solo afecta al SDK nativo o administrado. No afecta a la versión de JavaScript del SDK.

### Correcciones de errores

- Corrección de FromSubscription cuando se usa con la transcripción de la conversación.
- Corrección de errores en la detección de palabras clave para los asistentes por voz.

## Speech SDK 1.5.0 Versión de mayo de 2019

### Nuevas características:

- La detección de palabras clave (KWS) ahora está disponible para Windows y Linux. La funcionalidad KWS podría funcionar con cualquier tipo de micrófono; no obstante, la compatibilidad oficial de KWS está limitada actualmente a las matrices de micrófonos que se encuentran en el hardware de Azure Kinect DK o el SDK de dispositivos de voz.
- La funcionalidad de sugerencia de frases está disponible a través del SDK. Para más información, consulte [esta página](#).
- La funcionalidad de transcripción de conversaciones está disponible a través del SDK. Consulte [aquí](#).
- Compatibilidad agregada con los asistentes por voz mediante el canal Direct Line Speech.

### Muestras

- Se han agregado ejemplos para nuevas características o nuevos servicios admitidos por el SDK.

### Mejoras y cambios

- Se han agregado varias propiedades de reconocimiento para ajustar el comportamiento del servicio o los resultados del servicio (por ejemplo, enmascaramiento de palabras soeces etc.).
- Ahora puede configurar el reconocimiento a través de las propiedades de configuración estándar, incluso si ha creado el valor de `FromEndpoint` del reconocedor.
- Objective-C: se agregó la propiedad `OutputFormat` a `SPXSpeechConfiguration`.
- El SDK ahora admite Debian 9 como una distribución de Linux.

### Correcciones de errores

- Se ha corregido un problema donde el recurso de altavoz se destruía demasiado pronto en la conversión de texto a voz.

## Speech SDK 1.4.2

Se trata de una versión de corrección de errores y solo afecta al SDK nativo o administrado. No afecta a la versión de JavaScript del SDK.

## Speech SDK 1.4.1

Esta es una versión solo para JavaScript. No se agregó ninguna característica. Se realizaron las siguientes correcciones:

- Se impide que el paquete web cargue https-proxy-agent.

## Speech SDK 1.4.0 Versión de abril de 2019

### Nuevas características:

- El SDK admite ahora el servicio de conversión de texto a voz en versión beta. Se admite en Windows y Linux Desktop desde C++ y C#. Para más información, consulte la [información general sobre la conversión de texto a voz](#).
- El SDK ahora admite archivos de audio MP3 y Opus/OGG como archivos de entrada de secuencia. Esta característica solo está disponible en Linux desde C++ y C# y está actualmente en versión beta (más detalles [aquí](#)).
- Speech SDK para Java, .NET Core, C++ y Objective-C ha conseguido compatibilidad con macOS. La compatibilidad de Objective-C con macOS está actualmente en versión beta.
- iOS: Speech SDK para iOS (Objective-C) ahora también se publica como una instancia de CocoaPod.
- JavaScript: compatibilidad con micrófono no predeterminada como dispositivo de entrada.
- JavaScript: compatibilidad con servidores proxy para Node.js.

### Muestras

- se han agregado ejemplos para usar Speech SDK con C++ y con Objective-C en macOS.
- Se han agregado ejemplos que muestran el uso del servicio de conversión de texto a voz.

### Mejoras y cambios

- Python: ahora se exponen propiedades adicionales de los resultados del reconocimiento mediante la propiedad `properties`.
- Para la compatibilidad adicional con el desarrollo y la depuración, puede redirigir la información de registro y diagnóstico del SDK a un archivo de registro (más información [aquí](#)).
- JavaScript: mejora del rendimiento del procesamiento de audio.

### Correcciones de errores

- Mac/iOS: se corrigió un error que daba lugar a una larga espera cuando no se podía establecer una conexión con el servicio de voz.
- Python: mejora del control de errores en los argumentos de las devoluciones de llamada de Python.
- JavaScript: se corrigieron los informes de estado erróneos de la voz que finalizaban en RequestSession.

## Speech SDK 1.3.1 Actualización de febrero de 2019

Se trata de una versión de corrección de errores y solo afecta al SDK nativo o administrado. No afecta a la versión de JavaScript del SDK.

### Corrección de error

- Se ha corregido una fuga de memoria cuando se usa la entrada de micrófono. No afecta a la entrada de archivos o basada en secuencias.

## Speech SDK 1.3.0: versión de febrero de 2019

### Nuevas características

- El SDK de voz admite la selección del micrófono de entrada mediante la clase `AudioConfig`. Esto permite transmitir datos de audio al servicio de voz desde un micrófono no predeterminado. Para más información, consulte la documentación en la que se describe cómo [seleccionar un dispositivo de entrada de audio](#). Esta característica aún no está disponible en JavaScript.
- Speech SDK ahora es compatible con Unity en una versión beta. Proporcione sus comentarios en la sección de problemas en el [repositorio de ejemplos de GitHub](#). Esta versión es compatible con Unity en Windows x86 y x64 (aplicaciones de escritorio o de la Plataforma universal de Windows) y Android (ARM32/64, x86). Puede encontrar más información en nuestra [guía de inicio rápido sobre Unity](#).
- El archivo `Microsoft.CognitiveServices.Speech.csharp.bindings.dll` (incluido en versiones anteriores) ya no es necesario. La funcionalidad está ahora integrada en el SDK principal.

## Muestras

El siguiente contenido nuevo está disponible en nuestro [repositorio de ejemplo](#):

- Ejemplos adicionales para `AudioConfig.FromMicrophoneInput`.
- Ejemplos adicionales de Python para traducción y reconocimiento de intenciones.
- Ejemplos adicionales para usar el objeto `Connection` en iOS.
- Ejemplos adicionales de Java para la traducción con la salida de audio.
- Nuevo ejemplo de uso de la [API de REST de transcripción de lotes](#).

## Mejoras y cambios

- Python
  - Mensajes de error y verificación de parámetros mejorada en `SpeechConfig`.
  - Adición de compatibilidad para el objeto `Connection`.
  - Compatibilidad con Python (x86) de 32 bits en Windows.
  - Speech SDK para Python ya no está disponible como beta.
- iOS
  - El SDK ahora se compila en función de la versión 12.1 del SDK de iOS.
  - El SDK ahora es compatible con las versiones 9.2 y posteriores de iOS.
  - Documentación de referencia mejorada y corrección de varios nombres de propiedad.
- JavaScript
  - Adición de compatibilidad para el objeto `Connection`.
  - Archivos de definición de tipos agregados para JavaScript agrupado.
  - Compatibilidad e implementación iniciales para sugerencias de frases.
  - Colección de propiedades devuelta con JSON del servicio para reconocimiento.
- Los archivos DLL de Windows contienen ahora un recurso de versión.
- Si crea un valor de `FromEndpoint` de reconocedor, puede agregar parámetros directamente a la dirección URL del punto de conexión. Con `FromEndpoint` no puede configurar el reconocedor mediante las propiedades de configuración estándar.

## Correcciones de errores

- La contraseña de proxy y el nombre de usuario de proxy vacíos no se administraron correctamente. Con esta versión, si establece el nombre de usuario de proxy y la contraseña de proxy en una cadena vacía, no se enviarán al conectarse al proxy.
- El identificador de sesión creado por el SDK no siempre es realmente aleatorio para algunos lenguajes o entornos. Se ha agregado la inicialización del generador aleatorio para corregir este problema.
- Control mejorado del token de autorización. Si desea usar un token de autorización, especifíquelo en `SpeechConfig` y deje la clave de suscripción vacía. A continuación, cree el reconocedor como de costumbre.

- En algunos casos, el objeto `Connection` no se publicó correctamente. Ahora se ha corregido.
- Se corrigió el ejemplo de JavaScript para admitir la salida de audio para la síntesis de traducción también en Safari.

## Speech SDK 1.2.1

Esta es una versión solo para JavaScript. No se agregó ninguna característica. Se realizaron las siguientes correcciones:

- Activar el final del flujo en `turn.end`, y no en `speech.end`.
- Corregir error de la bomba de audio por el que no se programaba el siguiente envío en caso de error del envío actual.
- Corregir el reconocimiento continuo con el token de autenticación.
- Corrección de errores de diferentes reconocedores y puntos de conexión.
- Mejoras en la documentación.

## Speech SDK 1.2.0: Versión de diciembre de 2018

### Nuevas características

- Python
  - La versión beta de la compatibilidad con Python (3.5 y versiones posteriores) está disponible con esta versión. Para más información, consulte [aquí](#)(quickstart-python.md).
- JavaScript
  - Speech SDK para JavaScript ha sido de código abierto. El código fuente está disponible en [GitHub](#).
  - Ya se admite Node.js; puede encontrar más información [aquí](#).
  - Se quitó la restricción de longitud para las sesiones de audio; la reconexión se realizará automáticamente en la portada.
- Objeto `Connection`
  - Desde el objeto `Recognizer`, puede acceder al objeto `Connection`. Este objeto le permite iniciar la conexión al servicio y suscribirse para conectar y desconectar eventos explícitamente. (Esta característica no está disponible aún ni en JavaScript ni en Python).
- Compatibilidad con Ubuntu 18.04.
- Android
  - Compatibilidad con ProGuard habilitada durante la generación del APK.

### Mejoras

- Mejoras en el uso de subprocessos internos, lo que reduce el número de subprocessos, bloqueos y exclusiones mutuas.
- Se mejoraron los informes de errores y la información. En algunos casos, los mensajes de error no se propagan totalmente.
- Se actualizaron las dependencias de desarrollo en JavaScript para usar los módulos actualizados.

### Correcciones de errores

- Se han corregido las fugas de causadas por un error de coincidencia de tipos en `RecognizeAsync`.
- En algunos casos, se perdieron excepciones.
- Corrección de las fugas de memoria en los argumentos de eventos de traducción.
- Se ha corregido un problema de bloqueo al volver a conectar en sesiones de larga ejecución.
- Se ha corregido un problema que podría dar lugar a que faltase el resultado final para las traducciones con errores.

- C#: Si no se esperaba una operación `async` en el subprocesso principal, es posible que se pudiese desechar el reconocedor antes de completarse la tarea asíncrona.
- Java: Se ha corregido un problema que provocaba un bloqueo de la VM de Java.
- Objective-C: Se ha corregido la asignación de la enumeración; se devolvió `RecognizedIntent` en lugar de `RecognizingIntent`.
- JavaScript: Se ha establecido el formato de salida predeterminado en "simple" en `SpeechConfig`.
- JavaScript: Se ha quitado una incoherencia entre las propiedades del objeto de configuración en JavaScript y otros lenguajes.

## Muestras

- Se han actualizado y corregido varios ejemplos, como las voces de salida para la traducción, etc.
- Se han agregado ejemplos de Node.js en el [repositorio de ejemplo](#).

# Speech SDK 1.1.0

## Nuevas características

- Compatibilidad con Android x86/x64.
- Compatibilidad con proxy: En el objeto `SpeechConfig`, ahora puede llamar a una función para establecer la información del proxy (nombre de host, puerto, nombre de usuario y contraseña). Esta característica no está disponible aún en iOS.
- Mensajes y códigos de error mejorados. Si un reconocimiento devolvió un error, esto ya ha establecido `Reason` (en el evento cancelado) o `CancellationDetails` (en el resultado del reconocimiento) en `Error`. El evento cancelado ahora contiene dos miembros adicionales, `ErrorCode` y `ErrorDetails`. Si el servidor devolvió información de error adicional con el error notificado, ahora estará disponible en los nuevos miembros.

## Mejoras

- Verificación adicional agregada en la configuración del reconocedor y mensaje de error adicional agregado.
- Control mejorado del silencio prolongado en medio de un archivo de audio.
- Paquete NuGet: para proyectos de .NET Framework, evita la compilación con la configuración de AnyCPU.

## Correcciones de errores

- En los reconocedores se han encontrado varias excepciones corregidas. Además, las excepciones se detectan y se convierten en un evento `Canceled`.
- Corrección de una fuga de memoria en la administración de propiedades.
- Se corrigió el error en el que un archivo de entrada de audio podría bloquear el reconocedor.
- Se corrigió un error donde se podrían recibir eventos después de un evento de detención de la sesión.
- Se corrigieron algunas condiciones de subprocessos.
- Se corrigió un problema de compatibilidad de iOS que podría dar lugar a un bloqueo.
- Mejoras de estabilidad para la compatibilidad del micrófono en Android.
- Se corrigió un error donde un reconocedor en JavaScript ignoraría el lenguaje de reconocimiento.
- Se corrigió un error que impedía establecer el valor `EndpointId` (en algunos casos) en JavaScript.
- Se cambió el orden de los parámetros en `AddIntent` en JavaScript y se agregó la firma de JavaScript `AddIntent` que faltaba.

## Muestras

- Se han agregado ejemplos de C++ y C# para el uso de transmisiones de inserción y extracción en el [repositorio de ejemplos](#).

## Speech SDK 1.0.1

Mejoras en la confiabilidad y correcciones de errores:

- Corrección de un potencial error grave debido a una condición de carrera al desechar un reconocedor.
- Corrección de un potencial error grave en el caso de propiedades sin establecer.
- Comprobación adicional de errores y parámetros.
- Objective-C: corrección de posibles errores graves causados por la invalidación de nombres en NSString.
- Objective-C: ajuste de visibilidad en la API.
- JavaScript: corrección con respecto a los eventos y sus cargas.
- Mejoras en la documentación.

Se ha agregado un nuevo ejemplo de Javascript en nuestro [repositorio de ejemplos](#).

## SDK de Voz 1.0.0 de Cognitive Services: Versión de septiembre de 2018

### Nuevas características:

- Compatibilidad con Objective-C en iOS. Consulte la [Guía de inicio rápido de Objective-C para iOS](#).
- Se admite JavaScript en el explorador. Consulte la [Guía de inicio rápido de JavaScript](#).

### Cambios importantes

- Con esta versión se presentan una serie de cambios importantes. Consulte [esta página](#) para más información.

## SDK de Voz 0.6.0 de Cognitive Services: Versión de agosto de 2018

### Nuevas características:

- Ahora, las aplicaciones de UWP creadas con SDK de Voz superan el Kit para la certificación de aplicaciones en Windows (WACK). Consulte la [Guía de inicio rápido de UWP](#).
- Compatibilidad con .NET Standard 2.0 en Linux (Ubuntu 16.04 x64).
- Experimental: compatibilidad con Java 8 en Windows (64 bits) y Linux (Ubuntu 16.04 x 64). Consulte la [Guía de inicio rápido de Java Runtime Environment](#).

### Cambios funcionales

- Se expone más información detallada sobre los errores de conexión.

### Cambios importantes

- En Java (Android), la función `SpeechFactory.configureNativePlatformBindingWithDefaultCertificate` ya no requiere un parámetro de ruta de acceso. Ahora, la ruta de acceso se detecta automáticamente en todas las plataformas compatibles.
- En Java y C#, se ha quitado el descriptor de acceso get- de la propiedad `EndpointUrl`.

### Correcciones de errores

- En Java, se implementa ahora el resultado de la síntesis de audio en el reconocedor de traducción.
- Se ha corregido un error que podía provocar subprocessos inactivos y un mayor número de sockets abiertos y sin usar.
- Se ha corregido un problema por el que un proceso de reconocimiento de larga ejecución podía terminar en mitad de la transmisión.
- Se ha corregido una condición de carrera en el proceso de apagado del reconocedor.

# SDK de Voz 0.5.0 de Cognitive Services: Versión de julio de 2018

## Nuevas características:

- Compatibilidad con la plataforma Android (API 23: Android Marshmallow 6.0 o posterior). Consulte el [inicio rápido de Android](#).
- Compatibilidad con .NET Standard 2.0 en Windows. Consulte el [inicio rápido de .NET Core](#).
- Experimental: compatibilidad con UWP en Windows (versión 1709 o posterior).
  - Consulte la [Guía de inicio rápido de UWP](#).
  - Nota: Las aplicaciones de UWP creadas con el SDK de Voz no pasan aún el Kit para la certificación de aplicaciones en Windows (WACK).
- Compatibilidad con el reconocimiento de ejecución prolongada con reconexión automática.

## Cambios funcionales

- `StartContinuousRecognitionAsync()` admite reconocimiento de ejecución prolongada.
- El resultado del reconocimiento contiene más campos. Tienen un desplazamiento desde el principio del audio y la duración (ambos en tics) del texto reconocido y valores adicionales que representan el estado de reconocimiento, por ejemplo, `InitialSilenceTimeout` e `InitialBabbleTimeout`.
- Compatibilidad con `AuthorizationToken` para la creación de instancias de fábrica.

## Cambios importantes

- Eventos de reconocimiento: el tipo de evento `NoMatch` se combina con el evento `Error`.
- `SpeechOutputFormat` en C# se llama ahora `OutputFormat` para concordar con C++.
- El tipo de valor devuelto de algunos métodos de la interfaz `AudioInputStream` se ha modificado ligeramente:
  - En Java, el método `read` ahora devuelve `long` en lugar de `int`.
  - En C#, el método `Read` ahora devuelve `uint` en lugar de `int`.
  - En C++, los métodos `Read` y `GetFormat` ahora devuelven `size_t` en lugar de `int`.
- C++: las instancias de secuencias de entrada de audio ahora solo se pueden pasar como un valor `shared_ptr`.

## Correcciones de errores

- Se han corregido los valores devueltos incorrectos cuando se agota el tiempo de espera de `RecognizeAsync()`.
- Se ha eliminado la dependencia de las bibliotecas de Media Foundation en Windows. El SDK ahora usa las API de audio básicas.
- Corrección de la documentación: se ha agregado una página de [regiones](#) para describir cuáles son las regiones admitidas.

## Problema conocido

- SDK de Voz para Android no informa de los resultados de la síntesis de voz para la traducción. Este problema se solucionará en la próxima versión.

# SDK de Voz 0.4.0 de Cognitive Services: Versión de junio de 2018

## Cambios funcionales

- `AudioInputStream`

Un reconocedor ahora puede consumir una secuencia como origen de audio. Para más información, consulte la [guía de procedimientos](#) relacionada.

- Formato de salida detallado

Al crear un elemento `SpeechRecognizer`, puede solicitar el formato de salida `Detailed` o `Simple`.

`DetailedSpeechRecognitionResult` contiene una puntuación de confianza, texto reconocido, formato léxico sin formato, formato normalizado y formato normalizado con palabras soeces enmascaradas.

## Cambio importante

- En C# se cambia de `SpeechRecognitionResult.RecognizedText` a `SpeechRecognitionResult.Text`.

## Correcciones de errores

- Se ha corregido un posible problema de devolución de llamada en la capa USP durante el apagado.
- Si un reconocedor usaba un archivo de entrada de audio, mantenía el identificador de archivo más tiempo del necesario.
- Se han eliminado varios interbloqueos entre el suministro de mensajes y el reconocedor.
- Se desencadena un resultado `NoMatch` cuando se agota la respuesta del servicio.
- Las bibliotecas de Media Foundation en Windows son de carga retrasada. Esta biblioteca solo es necesaria para la entrada del micrófono.
- La velocidad de carga de los datos de audio se limita al doble de la velocidad de audio original.
- En Windows, los ensamblados .NET de C# ahora son de nombre seguro.
- Corrección de la documentación: `Region` necesita información para crear un reconocedor.

Se han agregado más ejemplos y se actualizan constantemente. Para obtener el conjunto más reciente de ejemplos, consulte el [repositorio de GitHub de ejemplos de SDK de Voz](#).

## SDK de Voz 0.2.12733 de Cognitive Services: Versión de mayo de 2018

Esta versión es la primera versión preliminar pública de SDK de Voz de Cognitive Services.

# ¿Qué es la transcripción de conversaciones (versión preliminar)?

13/01/2020 • 7 minutes to read • [Edit Online](#)

La transcripción de conversaciones es una solución de [conversión de voz a texto](#) que combina el reconocimiento de voz, la identificación del hablante y la atribución de oraciones a cada hablante (lo que también se conoce como *diarización*) para proporcionar la transcripción asincrónica o en tiempo real de cualquier conversación. La transcripción de conversaciones distingue a los hablantes de una conversación para determinar quién dijo qué y cuándo, y facilita a los desarrolladores la tarea de agregar conversión de voz a texto a sus aplicaciones que realizan la diarización de varios hablantes.

## Principales características

- **Marcas de tiempo:** cada expresión del hablante tiene una marca de tiempo, por lo que puede encontrar fácilmente cuándo se dijo una frase.
- **Transcripciones legibles:** el formato y la puntuación de las transcripciones se agregan automáticamente para garantizar que el texto coincide lo más posible con lo que se ha dicho.
- **Perfiles de usuario:** los perfiles de usuario se generan mediante la recopilación de muestras de voz de usuarios y el envío de estas a la generación de firmas.
- **Identificación del hablante:** los hablantes se identifican mediante perfiles de usuario; a cada hablante se le asigna un *identificador de hablante*.
- **Diarización de varios hablantes:** determine quién dijo qué mediante la sintetización de la secuencia de audio con cada identificador de hablante.
- **Transcripción en tiempo real:** proporcione transcripciones en directo de quién dice qué y en qué momento mientras tiene lugar la conversación.
- **Transcripción asincrónica:** proporcione transcripciones con una mayor precisión mediante una secuencia de audio de varios canales.

### NOTE

Aunque la transcripción de conversaciones no impone un límite sobre el número de hablantes en la sala, está optimizada para entre 2 y 10 hablantes por sesión.

## Casos de uso

### Reuniones inclusivas

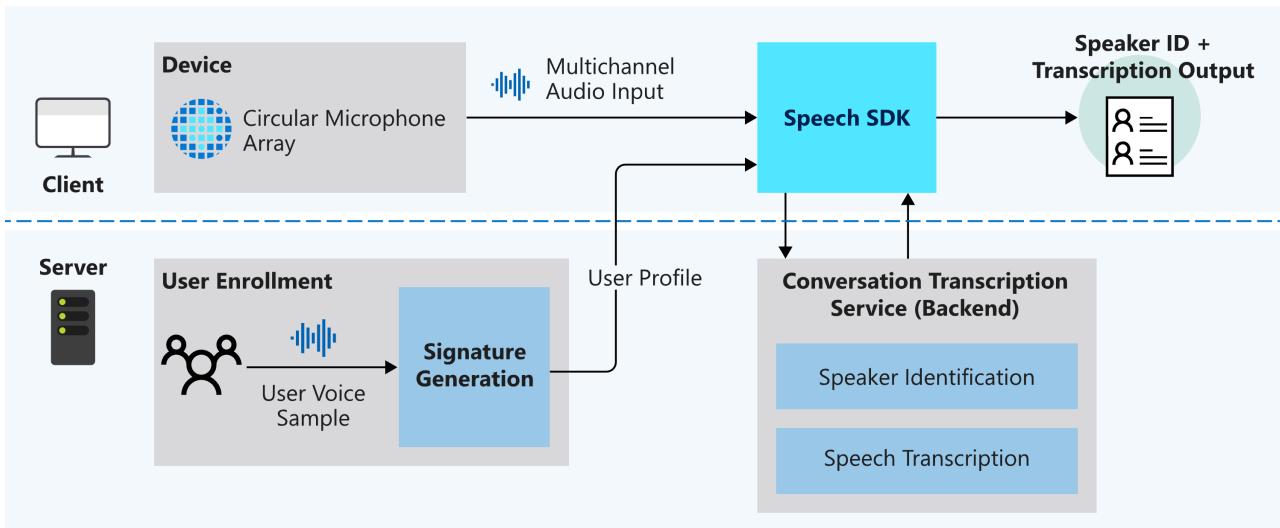
Para que las reuniones sean inclusivas para todo el mundo, por ejemplo, para los participantes sordos y con dificultades auditivas, es importante contar con transcripción en tiempo real. La transcripción de conversaciones en tiempo real toma el audio de una reunión y determina quién dice qué, lo que permite a todos los participantes seguir la transcripción y participar en la reunión sin ningún retraso.

### Mayor eficacia

Los participantes en la reunión pueden centrarse en la reunión y dejar la toma de notas a la transcripción de conversaciones. De este modo, pueden participar activamente en la reunión y realizar un seguimiento rápido de los pasos que vienen a continuación valiéndose de la transcripción en lugar de tomar notas y perderse algo posiblemente durante el evento.

# Cómo funciona

Así es a grandes rasgos cómo funciona la transcripción de conversaciones.



## Entradas esperadas

- **Secuencia de audio de varios canales:** para especificaciones y detalles de diseño, consulte [Micrófono del SDK de dispositivos de voz de Microsoft](#). Para más información o comprar un kit de desarrollo, consulte cómo [obtener Microsoft Speech Devices SDK](#).
- **Muestras de voz de usuarios:** la transcripción de conversaciones necesita perfiles de usuario antes de la conversación. Deberá recopilar grabaciones de audio de cada usuario y, luego, enviarlas al [servicio de generación de firmas](#) para validar el audio y generar los perfiles de usuario.

## Tiempo real frente a asincrónico

La transcripción de conversaciones ofrece tres modos de transcripción:

### Tiempo real

Los datos de audio se procesan en directo para devolver el identificador y la transcripción del hablante. Seleccione este modo si quiere usar la solución de transcripción para proporcionar a los participantes de la conversación una vista de la transcripción en directo de su conversación en curso. Por ejemplo, crear una aplicación para que las reuniones sean más accesibles para los participantes sordos y con dificultades auditivas es un caso de uso idóneo para la transcripción en tiempo real.

### Asincrónica

Los datos de audio se procesan por lotes para devolver el identificador y la transcripción del hablante. Seleccione este modo si quiere usar la solución de transcripción para proporcionar una mayor precisión sin la vista de la transcripción en directo. Por ejemplo, si quiere crear una aplicación para permitir que los participantes de la reunión se pongan al día fácilmente con las reuniones a las que han faltado, use el modo de transcripción asincrónica para obtener resultados de transcripción de alta precisión.

### Tiempo real y asincrónico

Los datos de audio se procesan en directo para devolver el identificador y la transcripción del hablante; además, se crea una solicitud para obtener también una transcripción de alta precisión mediante el procesamiento asincrónico. Seleccione este modo si la aplicación necesita una transcripción en tiempo real, pero también requiere una transcripción de mayor precisión después de la conversación o reunión.

## Compatibilidad con idiomas

Actualmente, la transcripción de conversaciones está disponible para los idiomas "en-US" y "zh-CN" en las siguientes regiones:*centralus* y *eastasia*. Si necesita ayuda adicional con la configuración regional, póngase en contacto con el [equipo de la característica Transcripción de conversaciones](#).

## Pasos siguientes

[Transcripción de conversaciones en tiempo real](#)

# Transcripción de conversaciones en tiempo real (versión preliminar)

13/01/2020 • 7 minutes to read • [Edit Online](#)

La API **ConversationTranscriber** del SDK de voz le permite transcribir reuniones y conversaciones con la posibilidad de agregar, quitar e identificar varios participantes haciendo streaming de audio al servicio de voz mediante `PullStream` o `PushStream`. Este tema requiere que sepa cómo usar la conversión de voz en texto con el SDK de voz (versión 1.8.0 o posterior). Para más información, consulte [Qué es el servicio de voz](#).

## Limitaciones

- Se admite la API ConversationTranscriber para C++, C# y Java en Windows, Linux y Android.
- Actualmente disponible para los idiomas "en-US" y "zh-CN" en las siguientes regiones: *centralus* y *eastasia*.
- Requiere una matriz con varios micrófonos circular de siete micrófonos con un flujo de referencia de reproducción. La matriz de micrófonos debe cumplir [nuestra especificación](#).
- El [SDK de dispositivos de voz](#) proporciona dispositivos adecuados y una aplicación de ejemplo que muestra la transcripción de conversaciones.

## Recursos de código de ejemploopcionales

El SDK de dispositivos de voz proporciona código de ejemplo en Java para la captura de audio en tiempo real mediante ocho canales.

- [ROOBO device sample code](#) (Código de ejemplo de dispositivo ROOBO)
- [Azure Kinect Dev Kit sample code](#) (Código de ejemplo de Azure Kinect Dev Kit)

## Requisitos previos

Una suscripción al servicio de voz. Puede [obtener una suscripción de prueba a Voz](#) si no la tiene.

## Creación de firmas de voz

El primer paso es crear firmas de voz para los participantes de la conversación para una identificación eficaz del hablante.

### Requisitos de entrada de audio

- El archivo WAVE de audio de entrada para la creación de firmas de voz debe aparecer en las muestras de 16 bits, con frecuencia de muestreo de 16 kHz y un formato de canal único (mono).
- La longitud recomendada para cada muestra de audio está entre 30 segundos y dos minutos.

### Código de ejemplo

El ejemplo siguiente muestra dos maneras diferentes de crear la firma de voz mediante la [API de REST](#) de C#. Tenga en cuenta que necesitará sustituir la información real de "YourSubscriptionKey", su nombre de archivo WAVE de "speakerVoice.wav" y su región de `{region}` y "YourServiceRegion" (*centralus* o *eastasia*).

```

class Program
{
    static async Task CreateVoiceSignatureByUsingFormData()
    {
        // Replace with your own region
        var region = "YourServiceRegion";
        // Change the name of the wave file to match yours
        byte[] fileBytes = File.ReadAllBytes(@"speakerVoice.wav");
        var form = new MultipartFormDataContent();
        var content = new ByteArrayContent(fileBytes);
        form.Add(content, "file", "file");
        var client = new HttpClient();
        // Add your subscription key to the header Ocp-Apim-Subscription-Key directly
        // Replace "YourSubscriptionKey" with your own subscription key
        client.DefaultRequestHeaders.Add("Ocp-Apim-Subscription-Key", "YourSubscriptionKey");
        // Edit with your desired region for `'{region}'`
        var response = await client.PostAsync($"https://signature.{region}.cts.speech.microsoft.com/api/v1/Signature/GenerateVoiceSignatureFromFormData", form);
        // A voice signature contains Version, Tag and Data key values from the Signature json structure from the Response body.
        // Voice signature format example: { "Version": <Numeric value>, "Tag": "string", "Data": "string" }
        var jsonData = await response.Content.ReadAsStringAsync();
    }

    static async Task CreateVoiceSignatureByUsingBody()
    {
        // Replace with your own region
        var region = "YourServiceRegion";
        // Change the name of the wave file to match yours
        byte[] fileBytes = File.ReadAllBytes(@"speakerVoice.wav");
        var content = new ByteArrayContent(fileBytes);

        var client = new HttpClient();
        // Add your subscription key to the header Ocp-Apim-Subscription-Key directly
        // Replace "YourSubscriptionKey" with your own subscription key
        client.DefaultRequestHeaders.Add("Ocp-Apim-Subscription-Key", "YourSubscriptionKey");
        // Edit with your desired region for `'{region}'`
        var response = await client.PostAsync($"https://signature.{region}.cts.speech.microsoft.com/api/v1/Signature/GenerateVoiceSignatureFromByteArray", content);
        // A voice signature contains Version, Tag and Data key values from the Signature json structure from the Response body.
        // Voice signature format example: { "Version": <Numeric value>, "Tag": "string", "Data": "string" }
        var jsonData = await response.Content.ReadAsStringAsync();
    }

    static void Main(string[] args)
    {
        CreateVoiceSignatureByUsingFormData().Wait();
        CreateVoiceSignatureByUsingBody().Wait();
    }
}

```

## Transcripción de conversaciones

El siguiente código de ejemplo muestra cómo transcribir conversaciones en tiempo real para tres hablantes. Se da por supuesto que ya ha creado firmas de voz para cada hablante como se muestra arriba. Sustituya la información real de "YourSubscriptionKey" y "YourServiceRegion" al crear el objeto SpeechConfig.

Estos son aspectos destacados de código de ejemplo:

- Creación de un objeto `Conversation` a partir del objeto `SpeechConfig` mediante un identificador de reunión generado con `Guid.NewGuid()`
- Creación de un objeto `ConversationTranscriber` y participación en la conversación con `JoinConversationAsync()`

para iniciar la transcripción

- Registro de los eventos de interés
- Incorporación de participantes a la conversación o eliminación de los mismos mediante el objeto Conversation
- Transmisión del audio

El identificador del hablante y la transcripción vuelven en los eventos registrados.

```
using Microsoft.CognitiveServices.Speech;
using Microsoft.CognitiveServices.Speech.Audio;
using Microsoft.CognitiveServices.Speech.Transcription;

public class MyConversationTranscriber
{
    public static async Task ConversationWithPullAudioStreamAsync()
    {
        // Creates an instance of a speech config with specified subscription key and service region
        // Replace with your own subscription key and region
        var config = SpeechConfig.FromSubscription("YourSubscriptionKey", "YourServiceRegion");
        config.SetProperty("ConversationTranscriptionInRoomAndOnline", "true");

        var stopTranscription = new TaskCompletionSource<int>();

        // Create an audio stream from a wav file or from the default microphone if you want to stream live
        // audio from the supported devices
        // Replace with your own audio file name and Helper class which implements AudioConfig using
        PullAudioInputStreamCallback
        using (var audioInput = Helper.OpenWavFile(@"8channelsOfRecordedPCMAudio.wav"))
        {
            var meetingId = Guid.NewGuid().ToString();
            using (var conversation = new Conversation(config, meetingId))
            {
                // Create a conversation transcriber using audio stream input
                using (var conversationTranscriber = new ConversationTranscriber(audioInput))
                {
                    await conversationTranscriber.JoinConversationAsync(conversation);

                    // Subscribe to events
                    conversationTranscriber.Transcribing += (s, e) =>
                    {
                        Console.WriteLine($"TRANSCRIBING: Text={e.Result.Text}");
                    };

                    conversationTranscriber.Transcribed += (s, e) =>
                    {
                        if (e.Result.Reason == ResultReason.RecognizedSpeech)
                        {
                            Console.WriteLine($"TRANSCRIBED: Text={e.Result.Text}, UserID={e.Result.UserId}");
                        }
                        else if (e.Result.Reason == ResultReason.NoMatch)
                        {
                            Console.WriteLine($"NOMATCH: Speech could not be recognized.");
                        }
                    };
                };

                conversationTranscriber.Canceled += (s, e) =>
                {
                    Console.WriteLine($"CANCELED: Reason={e.Reason}");

                    if (e.Reason == CancellationReason.Error)
                    {
                        Console.WriteLine($"CANCELED: ErrorCode={e.ErrorCode}");
                        Console.WriteLine($"CANCELED: ErrorDetails={e.ErrorDetails}");
                        Console.WriteLine($"CANCELED: Did you update the subscription info?");

                        stopTranscription.TrySetResult(0);
                    }
                };
            };
        };
    }
}
```

## Pasos siguientes

## Transcripción de conversaciones asincrónica

# Transcripción de conversaciones asincrónica (versión preliminar)

13/01/2020 • 6 minutes to read • [Edit Online](#)

En este artículo, se muestra la transcripción de conversaciones asincrónica mediante la API

**RemoteConversationTranscriptionClient**. Si ha configurado la transcripción de conversaciones para realizar la transcripción asincrónica y tiene un `conversationId`, puede obtener la transcripción asociada a ese `conversationId` mediante la API **RemoteConversationTranscriptionClient**.

## Asincrónica frente a la combinación de tiempo real y asincrónica

Con la transcripción asincrónica, se transmite el audio de conversación, pero no es necesario que se devuelva una transcripción en tiempo real. En su lugar, después de enviar el audio, use el `conversationId` de `Conversation` para consultar el estado de la transcripción asincrónica. Cuando la transcripción asincrónica esté lista, obtendrá un `RemoteConversationTranscriptionResult`.

Con una combinación de transcripción en tiempo real y asincrónica, se obtiene la transcripción en tiempo real, pero también se obtiene la transcripción mediante consultas con el `conversationId` (similar a un escenario asincrónico).

Se requieren dos pasos para realizar la transcripción asincrónica. El primer paso consiste en cargar el audio y elegir solo transcripción asincrónica o en tiempo real y asincrónica. El segundo paso es obtener los resultados de la transcripción.

## Carga del audio

Antes de que se pueda realizar la transcripción asincrónica, debe enviar el audio al servicio de transcripción de conversaciones mediante el SDK del cliente de Voz de Microsoft Cognitive (versión 1.8.0 o superior).

En este ejemplo de código se muestra cómo crear un transcriptor de conversaciones para el modo de solo asincrónico. Para transmitir audio al transcriptor, debe agregar código de streaming de audio derivado de la [transcripción de conversaciones en tiempo real con el SDK de Voz](#). Consulte la sección de **limitaciones** de ese tema para ver las API de lenguajes y plataformas admitidas.

```
// Create the speech config object
// Substitute real information for "YourSubscriptionKey" and "Region"
SpeechConfig speechConfig = SpeechConfig.fromSubscription("YourSubscriptionKey", "Region");
speechConfig.setProperty("ConversationTranscriptionInRoomAndOnline", "true");

// Set the property for asynchronous transcription
speechConfig.setServiceProperty("transcriptionMode", "Async", ServicePropertyChannel.UriQueryParameter);

// Set the property for real-time plus asynchronous transcription
//speechConfig.setServiceProperty("transcriptionMode", "RealTimeAndAsync",
//ServicePropertyChannel.UriQueryParameter);

// pick a conversation Id that is a GUID.
String conversationId = UUID.randomUUID().toString();

// Create a Conversation
Conversation conversation = new Conversation(speechConfig, conversationId);

// Create an audio stream from a wav file or from the default microphone if you want to stream live audio from
// the supported devices
```

```

// Replace with your own audio file name and Helper class which implements AudioConfig using
PullAudioInputStreamCallback
PullAudioInputStreamCallback wavfilePullStreamCallback =
Helper.OpenWavFile("16Khz16Bits8channelsOfRecordedPCMAudio.wav");
// Create an audio stream format assuming the file used above is 16Khz, 16 bits and 8 channel pcm wav file
AudioStreamFormat audioStreamFormat = AudioStreamFormat.getWaveFormatPCM((long)16000, (short)16,(short)8);
// Create an input stream
AudioInputStream audioStream = AudioInputStream.createPullStream(wavfilePullStreamCallback,
audioStreamFormat);

// Create a conversation transcriber
ConversationTranscriber transcriber = new ConversationTranscriber(AudioConfig.fromStreamInput(audioStream));

// join a conversation
transcriber.joinConversationAsync(conversation);

// Add the event listener for the realtime events
transcriber.transcribed.addEventListner((o, e) -> {
    System.out.println("Conversation transcriber Recognized:" + e.toString());
});

transcriber.canceled.addEventListner((o, e) -> {
    System.out.println("Conversation transcriber canceled:" + e.toString());
    try {
        transcriber.stopTranscribingAsync().get();
    } catch (InterruptedException ex) {
        ex.printStackTrace();
    } catch (ExecutionException ex) {
        ex.printStackTrace();
    }
});

transcriber.sessionStopped.addEventListner((o, e) -> {
    System.out.println("Conversation transcriber stopped:" + e.toString());

    try {
        transcriber.stopTranscribingAsync().get();
    } catch (InterruptedException ex) {
        ex.printStackTrace();
    } catch (ExecutionException ex) {
        ex.printStackTrace();
    }
});

// start the transcription.
Future<?> future = transcriber.startTranscribingAsync();
...

```

Si quiere que sea en tiempo real y asincrónica, agregue y quite la marca de comentario de las líneas de código adecuadas como se indica a continuación:

```

// Set the property for asynchronous transcription
//speechConfig.setServiceProperty("transcriptionMode", "Async", ServicePropertyChannel.UriQueryParameter);

// Set the property for real-time plus asynchronous transcription
speechConfig.setServiceProperty("transcriptionMode", "RealTimeAndAsync",
ServicePropertyChannel.UriQueryParameter);

```

## Obtención de los resultados de la transcripción

Este paso obtiene los resultados de la transcripción asincrónica pero supone que el procesamiento en tiempo real que haya solicitado se realiza en otro lugar. Para obtener más información, consulte [transcripción de conversaciones en tiempo real con el SDK de Voz](#).

Para el código que se muestra aquí, necesita **remote-conversation versión 1.8.0**, compatible solo con Java (1.8.0 o superior) en Windows, Linux y Android (nivel 26 de API o superior).

## Obtención del SDK de cliente

Puede obtener **remote-conversation** si edita el archivo pom.xml como se indica a continuación.

1. Al final del archivo, antes de la etiqueta de cierre `</project>`, cree un elemento `repositories` con una referencia al repositorio de Maven para el SDK de Voz:

```
<repositories>
<repository>
<id>maven-cognitiveservices-speech</id>
<name>Microsoft Cognitive Services Speech Maven Repository</name>
<url>https://csspeechstorage.blob.core.windows.net/maven/</url>
</repository>
</repositories>
```

2. Agregue también un elemento `dependencies`, con `remoteconversation-client-sdk 1.8.0` como dependencia:

```
<dependencies>
<dependency>
<groupId>com.microsoft.cognitiveservices.speech.remoteconversation</groupId>
<artifactId>remote-conversation</artifactId>
<version>1.8.0</version>
</dependency>
</dependencies>
```

3. Guarde los cambios.

## Código de transcripción de ejemplo

Una vez que tenga `conversationId`, cree cliente de transcripción de conversaciones remoto

**RemoteConversationTranscriptionClient** en la aplicación cliente para consultar el estado de la transcripción asincrónica. Use el método `getTranscriptionOperation` en **RemoteConversationTranscriptionClient** para obtener un objeto `PollerFlux`. El objeto PollerFlux tendrá información sobre el estado de la operación remota

**RemoteConversationTranscriptionOperation** y el resultado final **RemoteConversationTranscriptionResult**.

Una vez finalizada la operación, obtenga **RemoteConversationTranscriptionResult** llamando a `getFinalResult` en una `SyncPoller`. En este código, simplemente se imprime el contenido de los resultados en la salida del sistema.

```

// Create the speech config object
SpeechConfig speechConfig = SpeechConfig.fromSubscription("YourSubscriptionKey", "Region");

// Create a remote Conversation Transcription client
RemoteConversationTranscriptionClient client = new RemoteConversationTranscriptionClient(speechConfig);

// Get the PollerFlux for the remote operation
PollerFlux<RemoteConversationTranscriptionOperation, RemoteConversationTranscriptionResult>
remoteTranscriptionOperation = client.getTranscriptionOperation(conversationId);

// Subscribe to PollerFlux to get the remote operation status
remoteTranscriptionOperation.subscribe(
    pollResponse -> {
        System.out.println("Poll response status : " + pollResponse.getStatus());
        System.out.println("Poll response status : " + pollResponse.getValue().getServiceStatus());
    }
);

// Obtain the blocking operation using getSyncPoller
SyncPoller<RemoteConversationTranscriptionOperation, RemoteConversationTranscriptionResult> blockingOperation
= remoteTranscriptionOperation.getSyncPoller();

// Wait for the operation to finish
blockingOperation.waitForCompletion();

// Get the final result response
RemoteConversationTranscriptionResult resultResponse = blockingOperation.getFinalResult();

// Print the result
if(resultResponse != null) {
    if(resultResponse.getConversationTranscriptionResults() != null) {
        for (int i = 0; i < resultResponse.getConversationTranscriptionResults().size(); i++) {
            ConversationTranscriptionResult result =
resultResponse.getConversationTranscriptionResults().get(i);

System.out.println(result.getProperties().getProperty(PropertyId.SpeechServiceResponse_JsonResult.name()));

System.out.println(result.getProperties().getProperty(PropertyId.SpeechServiceResponse_JsonResult));
        System.out.println(result.getOffset());
        System.out.println(result.getDuration());
        System.out.println(result.getUserId());
        System.out.println(result.getReason());
        System.out.println(result.getResultId());
        System.out.println(result.getText());
        System.out.println(result.toString());
    }
}
}

System.out.println("Operation finished");

```

## Pasos siguientes

[Exploración de ejemplos en GitHub](#)

# Notas de la versión

15/01/2020 • 33 minutes to read • [Edit Online](#)

## SDK de Voz 1.8.0: Versión de noviembre de 2019

### Nuevas características

- Se ha agregado una API `FromHost()` para facilitar su uso con contenedores locales y nubes soberanas.
- Se ha agregado la detección automática de idioma de origen para el reconocimiento de voz (en Java y C++)
- Se ha agregado el objeto `SourceLanguageConfig` para el reconocimiento de voz, que se usa para especificar los idiomas de origen esperados (en Java y C++).
- Se ha agregado compatibilidad con `KeywordRecognizer` en Windows (UWP), Android e iOS mediante los paquetes de Nuget y Unity.
- Se ha agregado la API de Java de conversación remota para realizar la transcripción de conversaciones en lotes asíncronos.

### Cambios importantes

- Las funcionalidades de transcripción de conversaciones se han movido al espacio de nombres `Microsoft.CognitiveServices.Speech.Transcription`.
- Parte de los métodos de transcripción de conversaciones se han movido a la nueva clase `Conversation`.
- Compatibilidad eliminada para iOS de 32 bits (ARMv7 y x86)

### Correcciones de errores

- Se ha corregido un bloqueo si se usa `KeywordRecognizer` local sin una clave de suscripción válida al servicio de voz.

### Muestras

- Ejemplo de Xamarin para `KeywordRecognizer`
- Ejemplo de Unity para `KeywordRecognizer`
- Ejemplos de C++ y Java de detección automática de idioma de origen

## SDK de voz 1.7.0: versión de septiembre de 2019

### Nuevas características

- Compatibilidad con la versión beta agregada para Xamarin en la Plataforma universal de Windows (UWP), Android e iOS
- Compatibilidad con iOS agregada para Unity
- Se ha agregado compatibilidad con entradas `Compressed` para ALaw, Mulaw, FLAC en Android, iOS y Linux.
- Se ha agregado `SendMessageAsync` en la clase `Connection` para enviar un mensaje al servicio.
- Se ha agregado `SetMessageProperty` en la clase `Connection` para establecer la propiedad de un mensaje.
- TTS agregó compatibilidad para Java (JRE y Android), Python, Swift y Objective-C
- TTS agregó compatibilidad de reproducción para macOS, iOS y Android
- Se ha agregado información de "límite de palabras" para TTS

### Correcciones de errores

- Se ha corregido un problema de compilación de IL2CPP en Unity 2019 para Android

- Se ha corregido un problema con los encabezados con formato incorrecto en la entrada de archivo WAV que se procesa de forma incorrecta
- Se ha corregido un problema con UUID que no es único en algunas propiedades de conexión
- Se han corregido algunas advertencias sobre los especificadores de nulabilidad en los enlaces SWIFT (puede que se requieran pequeños cambios en el código)
- Se ha corregido un error que provocaba que las conexiones de WebSocket se cerraran de manera incorrecta en la carga de red
- Se ha corregido un problema en Android que a veces provoca que `DialogServiceConnector` use identificadores de impresión duplicados.
- Se han introducido mejoras en la estabilidad de las conexiones entre interacciones multiproceso y la generación de informes de errores (a través de eventos `Canceled`) cuando se producen con `DialogServiceConnector`.
- Los inicios de sesión de `DialogServiceConnector` ahora proporcionarán eventos correctamente, incluso si se llama a `ListenOnceAsync()` durante una operación `StartKeywordRecognitionAsync()` activa.
- Se ha resuelto un bloqueo asociado a la recepción de actividades `DialogServiceConnector`.

## Muestras

- Inicio rápido para Xamarin
- Se ha actualizado el inicio rápido de CPP con información de ARM64 de Linux
- Se ha actualizado el inicio rápido de Unity con información de iOS

# SDK de Voz 1.6.0: versión de junio de 2019

## Muestras

- Ejemplos de inicio rápido para Texto a voz en UWP y Unity
- Ejemplo de inicio rápido para Swift en iOS
- Ejemplos de Unity para Traducción y Reconocimiento de la intención comunicativa y Voz
- Ejemplos de inicios rápidos actualizados para `DialogServiceConnector`

## Mejoras y cambios

- Espacio de nombres de cuadro de diálogo:
  - El nombre de `SpeechBotConnector` ha cambiado a `DialogServiceConnector`
  - El nombre de `BotConfig` ha cambiado a `DialogServiceConfig`
  - `BotConfig::FromChannelSecret()` se ha reasignado a `DialogServiceConfig::FromBotSecret()`
  - Todos los clientes de Voz de Direct Line existentes siguen siendo compatibles después del cambio de nombre
- Actualización del adaptador REST de TTS para admitir una conexión persistente de proxy
- Un mejor mensaje de error cuando se pasa una región no válida
- Swift/Objective-C:
  - Mejores informes de errores: los métodos que pueden generar un error ahora se encuentran en dos versiones: una que expone un objeto `NSError` para el control de errores y una que genera una excepción. La primera se expone a Swift. Este cambio requiere adaptaciones en el código Swift existente.
  - Mejor control de eventos

## Correcciones de errores

- Corrección de TTS: donde el futuro de `SpeakTextAsync` se devolvió sin esperar al fin de la representación del audio
- Corrección para la serialización de las cadenas en C# para permitir la compatibilidad total con idiomas
- Corrección del problema de las aplicaciones centrales de .NET para cargar la biblioteca principal con un marco

de destino net461 en ejemplos

- Corrección de problemas ocasionales para implementar bibliotecas nativas en la carpeta de salida en los ejemplos
- Corrección para cerrar el socket web de manera confiable
- Corrección de un posible bloqueo al abrir una conexión con una carga muy elevada en Linux
- Corrección de metadatos faltantes en el paquete de marcos para macOS
- Corrección de problemas con `pip install --user` en Windows

## Speech SDK 1.5.1

Se trata de una versión de corrección de errores y solo afecta al SDK nativo o administrado. No afecta a la versión de JavaScript del SDK.

### Correcciones de errores

- Corrección de FromSubscription cuando se usa con la transcripción de la conversación.
- Corrección de errores en la detección de palabras clave para los asistentes por voz.

## Speech SDK 1.5.0 Versión de mayo de 2019

### Nuevas características:

- La detección de palabras clave (KWS) ahora está disponible para Windows y Linux. La funcionalidad KWS podría funcionar con cualquier tipo de micrófono; no obstante, la compatibilidad oficial de KWS está limitada actualmente a las matrices de micrófonos que se encuentran en el hardware de Azure Kinect DK o el SDK de dispositivos de voz.
- La funcionalidad de sugerencia de frases está disponible a través del SDK. Para más información, consulte [esta página](#).
- La funcionalidad de transcripción de conversaciones está disponible a través del SDK. Consulte [aquí](#).
- Compatibilidad agregada con los asistentes por voz mediante el canal Direct Line Speech.

### Muestras

- Se han agregado ejemplos para nuevas características o nuevos servicios admitidos por el SDK.

### Mejoras y cambios

- Se han agregado varias propiedades de reconocimiento para ajustar el comportamiento del servicio o los resultados del servicio (por ejemplo, enmascaramiento de palabras soeces etc.).
- Ahora puede configurar el reconocimiento a través de las propiedades de configuración estándar, incluso si ha creado el valor de `FromEndpoint` del reconocedor.
- Objective-C: se agregó la propiedad `OutputFormat` a `SPXSpeechConfiguration`.
- El SDK ahora admite Debian 9 como una distribución de Linux.

### Correcciones de errores

- Se ha corregido un problema donde el recurso de altavoz se destruía demasiado pronto en la conversión de texto a voz.

## Speech SDK 1.4.2

Se trata de una versión de corrección de errores y solo afecta al SDK nativo o administrado. No afecta a la versión de JavaScript del SDK.

## Speech SDK 1.4.1

Esta es una versión solo para JavaScript. No se agregó ninguna característica. Se realizaron las siguientes correcciones:

- Se impide que el paquete web cargue https-proxy-agent.

## Speech SDK 1.4.0 Versión de abril de 2019

### Nuevas características:

- El SDK admite ahora el servicio de conversión de texto a voz en versión beta. Se admite en Windows y Linux Desktop desde C++ y C#. Para más información, consulte la [información general sobre la conversión de texto a voz](#).
- El SDK ahora admite archivos de audio MP3 y Opus/OGG como archivos de entrada de secuencia. Esta característica solo está disponible en Linux desde C++ y C# y está actualmente en versión beta (más detalles [aquí](#)).
- Speech SDK para Java, .NET Core, C++ y Objective-C ha conseguido compatibilidad con macOS. La compatibilidad de Objective-C con macOS está actualmente en versión beta.
- iOS: Speech SDK para iOS (Objective-C) ahora también se publica como una instancia de CocoaPod.
- JavaScript: compatibilidad con micrófono no predeterminada como dispositivo de entrada.
- JavaScript: compatibilidad con servidores proxy para Node.js.

### Muestras

- se han agregado ejemplos para usar Speech SDK con C++ y con Objective-C en macOS.
- Se han agregado ejemplos que muestran el uso del servicio de conversión de texto a voz.

### Mejoras y cambios

- Python: ahora se exponen propiedades adicionales de los resultados del reconocimiento mediante la propiedad `properties`.
- Para la compatibilidad adicional con el desarrollo y la depuración, puede redirigir la información de registro y diagnóstico del SDK a un archivo de registro (más información [aquí](#)).
- JavaScript: mejora del rendimiento del procesamiento de audio.

### Correcciones de errores

- Mac/iOS: se corrigió un error que daba lugar a una larga espera cuando no se podía establecer una conexión con el servicio de voz.
- Python: mejora del control de errores en los argumentos de las devoluciones de llamada de Python.
- JavaScript: se corrigieron los informes de estado erróneos de la voz que finalizaban en RequestSession.

## Speech SDK 1.3.1 Actualización de febrero de 2019

Se trata de una versión de corrección de errores y solo afecta al SDK nativo o administrado. No afecta a la versión de JavaScript del SDK.

### Corrección de error

- Se ha corregido una fuga de memoria cuando se usa la entrada de micrófono. No afecta a la entrada de archivos o basada en secuencias.

## Speech SDK 1.3.0: versión de febrero de 2019

### Nuevas características

- El SDK de voz admite la selección del micrófono de entrada mediante la clase `AudioConfig`. Esto permite transmitir datos de audio al servicio de voz desde un micrófono no predeterminado. Para más información, consulte la documentación en la que se describe cómo [seleccionar un dispositivo de entrada de audio](#). Esta característica aún no está disponible en JavaScript.
- Speech SDK ahora es compatible con Unity en una versión beta. Proporcione sus comentarios en la sección de problemas en el [repositorio de ejemplos de GitHub](#). Esta versión es compatible con Unity en Windows x86 y x64 (aplicaciones de escritorio o de la Plataforma universal de Windows) y Android (ARM32/64, x86). Puede encontrar más información en nuestra [guía de inicio rápido sobre Unity](#).
- El archivo `Microsoft.CognitiveServices.Speech.csharp.bindings.dll` (incluido en versiones anteriores) ya no es necesario. La funcionalidad está ahora integrada en el SDK principal.

## Muestras

El siguiente contenido nuevo está disponible en nuestro [repositorio de ejemplo](#):

- Ejemplos adicionales para `AudioConfig.FromMicrophoneInput`.
- Ejemplos adicionales de Python para traducción y reconocimiento de intenciones.
- Ejemplos adicionales para usar el objeto `Connection` en iOS.
- Ejemplos adicionales de Java para la traducción con la salida de audio.
- Nuevo ejemplo de uso de la [API de REST de transcripción de lotes](#).

## Mejoras y cambios

- Python
  - Mensajes de error y verificación de parámetros mejorada en `SpeechConfig`.
  - Adición de compatibilidad para el objeto `Connection`.
  - Compatibilidad con Python (x86) de 32 bits en Windows.
  - Speech SDK para Python ya no está disponible como beta.
- iOS
  - El SDK ahora se compila en función de la versión 12.1 del SDK de iOS.
  - El SDK ahora es compatible con las versiones 9.2 y posteriores de iOS.
  - Documentación de referencia mejorada y corrección de varios nombres de propiedad.
- JavaScript
  - Adición de compatibilidad para el objeto `Connection`.
  - Archivos de definición de tipos agregados para JavaScript agrupado.
  - Compatibilidad e implementación iniciales para sugerencias de frases.
  - Colección de propiedades devuelta con JSON del servicio para reconocimiento.
- Los archivos DLL de Windows contienen ahora un recurso de versión.
- Si crea un valor de `FromEndpoint` de reconocedor, puede agregar parámetros directamente a la dirección URL del punto de conexión. Con `FromEndpoint` no puede configurar el reconocedor mediante las propiedades de configuración estándar.

## Correcciones de errores

- La contraseña de proxy y el nombre de usuario de proxy vacíos no se administraron correctamente. Con esta versión, si establece el nombre de usuario de proxy y la contraseña de proxy en una cadena vacía, no se enviarán al conectarse al proxy.
- El identificador de sesión creado por el SDK no siempre es realmente aleatorio para algunos lenguajes o entornos. Se ha agregado la inicialización del generador aleatorio para corregir este problema.
- Control mejorado del token de autorización. Si desea usar un token de autorización, especifíquelo en `SpeechConfig` y deje la clave de suscripción vacía. A continuación, cree el reconocedor como de costumbre.

- En algunos casos, el objeto `Connection` no se publicó correctamente. Ahora se ha corregido.
- Se corrigió el ejemplo de JavaScript para admitir la salida de audio para la síntesis de traducción también en Safari.

## Speech SDK 1.2.1

Esta es una versión solo para JavaScript. No se agregó ninguna característica. Se realizaron las siguientes correcciones:

- Activar el final del flujo en `turn.end`, y no en `speech.end`.
- Corregir error de la bomba de audio por el que no se programaba el siguiente envío en caso de error del envío actual.
- Corregir el reconocimiento continuo con el token de autenticación.
- Corrección de errores de diferentes reconocedores y puntos de conexión.
- Mejoras en la documentación.

## Speech SDK 1.2.0: Versión de diciembre de 2018

### Nuevas características

- Python
  - La versión beta de la compatibilidad con Python (3.5 y versiones posteriores) está disponible con esta versión. Para más información, consulte [aquí](#)(quickstart-python.md).
- JavaScript
  - Speech SDK para JavaScript ha sido de código abierto. El código fuente está disponible en [GitHub](#).
  - Ya se admite Node.js; puede encontrar más información [aquí](#).
  - Se quitó la restricción de longitud para las sesiones de audio; la reconexión se realizará automáticamente en la portada.
- Objeto `Connection`
  - Desde el objeto `Recognizer`, puede acceder al objeto `Connection`. Este objeto le permite iniciar la conexión al servicio y suscribirse para conectar y desconectar eventos explícitamente. (Esta característica no está disponible aún ni en JavaScript ni en Python).
- Compatibilidad con Ubuntu 18.04.
- Android
  - Compatibilidad con ProGuard habilitada durante la generación del APK.

### Mejoras

- Mejoras en el uso de subprocessos internos, lo que reduce el número de subprocessos, bloqueos y exclusiones mutuas.
- Se mejoraron los informes de errores y la información. En algunos casos, los mensajes de error no se propagan totalmente.
- Se actualizaron las dependencias de desarrollo en JavaScript para usar los módulos actualizados.

### Correcciones de errores

- Se han corregido las fugas de causadas por un error de coincidencia de tipos en `RecognizeAsync`.
- En algunos casos, se perdieron excepciones.
- Corrección de las fugas de memoria en los argumentos de eventos de traducción.
- Se ha corregido un problema de bloqueo al volver a conectar en sesiones de larga ejecución.
- Se ha corregido un problema que podría dar lugar a que faltase el resultado final para las traducciones con errores.

- C#: Si no se esperaba una operación `async` en el subprocesso principal, es posible que se pudiese desechar el reconocedor antes de completarse la tarea asíncrona.
- Java: Se ha corregido un problema que provocaba un bloqueo de la VM de Java.
- Objective-C: Se ha corregido la asignación de la enumeración; se devolvió `RecognizedIntent` en lugar de `RecognizingIntent`.
- JavaScript: Se ha establecido el formato de salida predeterminado en "simple" en `SpeechConfig`.
- JavaScript: Se ha quitado una incoherencia entre las propiedades del objeto de configuración en JavaScript y otros lenguajes.

## Muestras

- Se han actualizado y corregido varios ejemplos, como las voces de salida para la traducción, etc.
- Se han agregado ejemplos de Node.js en el [repositorio de ejemplo](#).

# Speech SDK 1.1.0

## Nuevas características

- Compatibilidad con Android x86/x64.
- Compatibilidad con proxy: En el objeto `SpeechConfig`, ahora puede llamar a una función para establecer la información del proxy (nombre de host, puerto, nombre de usuario y contraseña). Esta característica no está disponible aún en iOS.
- Mensajes y códigos de error mejorados. Si un reconocimiento devolvió un error, esto ya ha establecido `Reason` (en el evento cancelado) o `CancellationDetails` (en el resultado del reconocimiento) en `Error`. El evento cancelado ahora contiene dos miembros adicionales, `ErrorCode` y `ErrorDetails`. Si el servidor devolvió información de error adicional con el error notificado, ahora estará disponible en los nuevos miembros.

## Mejoras

- Verificación adicional agregada en la configuración del reconocedor y mensaje de error adicional agregado.
- Control mejorado del silencio prolongado en medio de un archivo de audio.
- Paquete NuGet: para proyectos de .NET Framework, evita la compilación con la configuración de AnyCPU.

## Correcciones de errores

- En los reconocedores se han encontrado varias excepciones corregidas. Además, las excepciones se detectan y se convierten en un evento `Canceled`.
- Corrección de una fuga de memoria en la administración de propiedades.
- Se corrigió el error en el que un archivo de entrada de audio podría bloquear el reconocedor.
- Se corrigió un error donde se podrían recibir eventos después de un evento de detención de la sesión.
- Se corrigieron algunas condiciones de subprocessos.
- Se corrigió un problema de compatibilidad de iOS que podría dar lugar a un bloqueo.
- Mejoras de estabilidad para la compatibilidad del micrófono en Android.
- Se corrigió un error donde un reconocedor en JavaScript ignoraría el lenguaje de reconocimiento.
- Se corrigió un error que impedía establecer el valor `EndpointId` (en algunos casos) en JavaScript.
- Se cambió el orden de los parámetros en `AddIntent` en JavaScript y se agregó la firma de JavaScript `AddIntent` que faltaba.

## Muestras

- Se han agregado ejemplos de C++ y C# para el uso de transmisiones de inserción y extracción en el [repositorio de ejemplos](#).

## Speech SDK 1.0.1

Mejoras en la confiabilidad y correcciones de errores:

- Corrección de un potencial error grave debido a una condición de carrera al desechar un reconocedor.
- Corrección de un potencial error grave en el caso de propiedades sin establecer.
- Comprobación adicional de errores y parámetros.
- Objective-C: corrección de posibles errores graves causados por la invalidación de nombres en NSString.
- Objective-C: ajuste de visibilidad en la API.
- JavaScript: corrección con respecto a los eventos y sus cargas.
- Mejoras en la documentación.

Se ha agregado un nuevo ejemplo de Javascript en nuestro [repositorio de ejemplos](#).

## SDK de Voz 1.0.0 de Cognitive Services: Versión de septiembre de 2018

### Nuevas características:

- Compatibilidad con Objective-C en iOS. Consulte la [Guía de inicio rápido de Objective-C para iOS](#).
- Se admite JavaScript en el explorador. Consulte la [Guía de inicio rápido de JavaScript](#).

### Cambios importantes

- Con esta versión se presentan una serie de cambios importantes. Consulte [esta página](#) para más información.

## SDK de Voz 0.6.0 de Cognitive Services: Versión de agosto de 2018

### Nuevas características:

- Ahora, las aplicaciones de UWP creadas con SDK de Voz superan el Kit para la certificación de aplicaciones en Windows (WACK). Consulte la [Guía de inicio rápido de UWP](#).
- Compatibilidad con .NET Standard 2.0 en Linux (Ubuntu 16.04 x64).
- Experimental: compatibilidad con Java 8 en Windows (64 bits) y Linux (Ubuntu 16.04 x 64). Consulte la [Guía de inicio rápido de Java Runtime Environment](#).

### Cambios funcionales

- Se expone más información detallada sobre los errores de conexión.

### Cambios importantes

- En Java (Android), la función `SpeechFactory.configureNativePlatformBindingWithDefaultCertificate` ya no requiere un parámetro de ruta de acceso. Ahora, la ruta de acceso se detecta automáticamente en todas las plataformas compatibles.
- En Java y C#, se ha quitado el descriptor de acceso get- de la propiedad `EndpointUrl`.

### Correcciones de errores

- En Java, se implementa ahora el resultado de la síntesis de audio en el reconocedor de traducción.
- Se ha corregido un error que podía provocar subprocessos inactivos y un mayor número de sockets abiertos y sin usar.
- Se ha corregido un problema por el que un proceso de reconocimiento de larga ejecución podía terminar en mitad de la transmisión.
- Se ha corregido una condición de carrera en el proceso de apagado del reconocedor.

# SDK de Voz 0.5.0 de Cognitive Services: Versión de julio de 2018

## Nuevas características:

- Compatibilidad con la plataforma Android (API 23: Android Marshmallow 6.0 o posterior). Consulte el [inicio rápido de Android](#).
- Compatibilidad con .NET Standard 2.0 en Windows. Consulte el [inicio rápido de .NET Core](#).
- Experimental: compatibilidad con UWP en Windows (versión 1709 o posterior).
  - Consulte la [Guía de inicio rápido de UWP](#).
  - Nota: Las aplicaciones de UWP creadas con el SDK de Voz no pasan aún el Kit para la certificación de aplicaciones en Windows (WACK).
- Compatibilidad con el reconocimiento de ejecución prolongada con reconexión automática.

## Cambios funcionales

- `StartContinuousRecognitionAsync()` admite reconocimiento de ejecución prolongada.
- El resultado del reconocimiento contiene más campos. Tienen un desplazamiento desde el principio del audio y la duración (ambos en tics) del texto reconocido y valores adicionales que representan el estado de reconocimiento, por ejemplo, `InitialSilenceTimeout` e `InitialBabbleTimeout`.
- Compatibilidad con `AuthorizationToken` para la creación de instancias de fábrica.

## Cambios importantes

- Eventos de reconocimiento: el tipo de evento `NoMatch` se combina con el evento `Error`.
- `SpeechOutputFormat` en C# se llama ahora `OutputFormat` para concordar con C++.
- El tipo de valor devuelto de algunos métodos de la interfaz `AudioInputStream` se ha modificado ligeramente:
  - En Java, el método `read` ahora devuelve `long` en lugar de `int`.
  - En C#, el método `Read` ahora devuelve `uint` en lugar de `int`.
  - En C++, los métodos `Read` y `GetFormat` ahora devuelven `size_t` en lugar de `int`.
- C++: las instancias de secuencias de entrada de audio ahora solo se pueden pasar como un valor `shared_ptr`.

## Correcciones de errores

- Se han corregido los valores devueltos incorrectos cuando se agota el tiempo de espera de `RecognizeAsync()`.
- Se ha eliminado la dependencia de las bibliotecas de Media Foundation en Windows. El SDK ahora usa las API de audio básicas.
- Corrección de la documentación: se ha agregado una página de [regiones](#) para describir cuáles son las regiones admitidas.

## Problema conocido

- SDK de Voz para Android no informa de los resultados de la síntesis de voz para la traducción. Este problema se solucionará en la próxima versión.

# SDK de Voz 0.4.0 de Cognitive Services: Versión de junio de 2018

## Cambios funcionales

- `AudioInputStream`

Un reconocedor ahora puede consumir una secuencia como origen de audio. Para más información, consulte la [guía de procedimientos](#) relacionada.

- Formato de salida detallado

Al crear un elemento `SpeechRecognizer`, puede solicitar el formato de salida `Detailed` o `Simple`.

`DetailedSpeechRecognitionResult` contiene una puntuación de confianza, texto reconocido, formato léxico sin formato, formato normalizado y formato normalizado con palabras soeces enmascaradas.

## Cambio importante

- En C# se cambia de `SpeechRecognitionResult.RecognizedText` a `SpeechRecognitionResult.Text`.

## Correcciones de errores

- Se ha corregido un posible problema de devolución de llamada en la capa USP durante el apagado.
- Si un reconocedor usaba un archivo de entrada de audio, mantenía el identificador de archivo más tiempo del necesario.
- Se han eliminado varios interbloqueos entre el suministro de mensajes y el reconocedor.
- Se desencadena un resultado `NoMatch` cuando se agota la respuesta del servicio.
- Las bibliotecas de Media Foundation en Windows son de carga retrasada. Esta biblioteca solo es necesaria para la entrada del micrófono.
- La velocidad de carga de los datos de audio se limita al doble de la velocidad de audio original.
- En Windows, los ensamblados .NET de C# ahora son de nombre seguro.
- Corrección de la documentación: `Region` necesita información para crear un reconocedor.

Se han agregado más ejemplos y se actualizan constantemente. Para obtener el conjunto más reciente de ejemplos, consulte el [repositorio de GitHub de ejemplos de SDK de Voz](#).

## SDK de Voz 0.2.12733 de Cognitive Services: Versión de mayo de 2018

Esta versión es la primera versión preliminar pública de SDK de Voz de Cognitive Services.

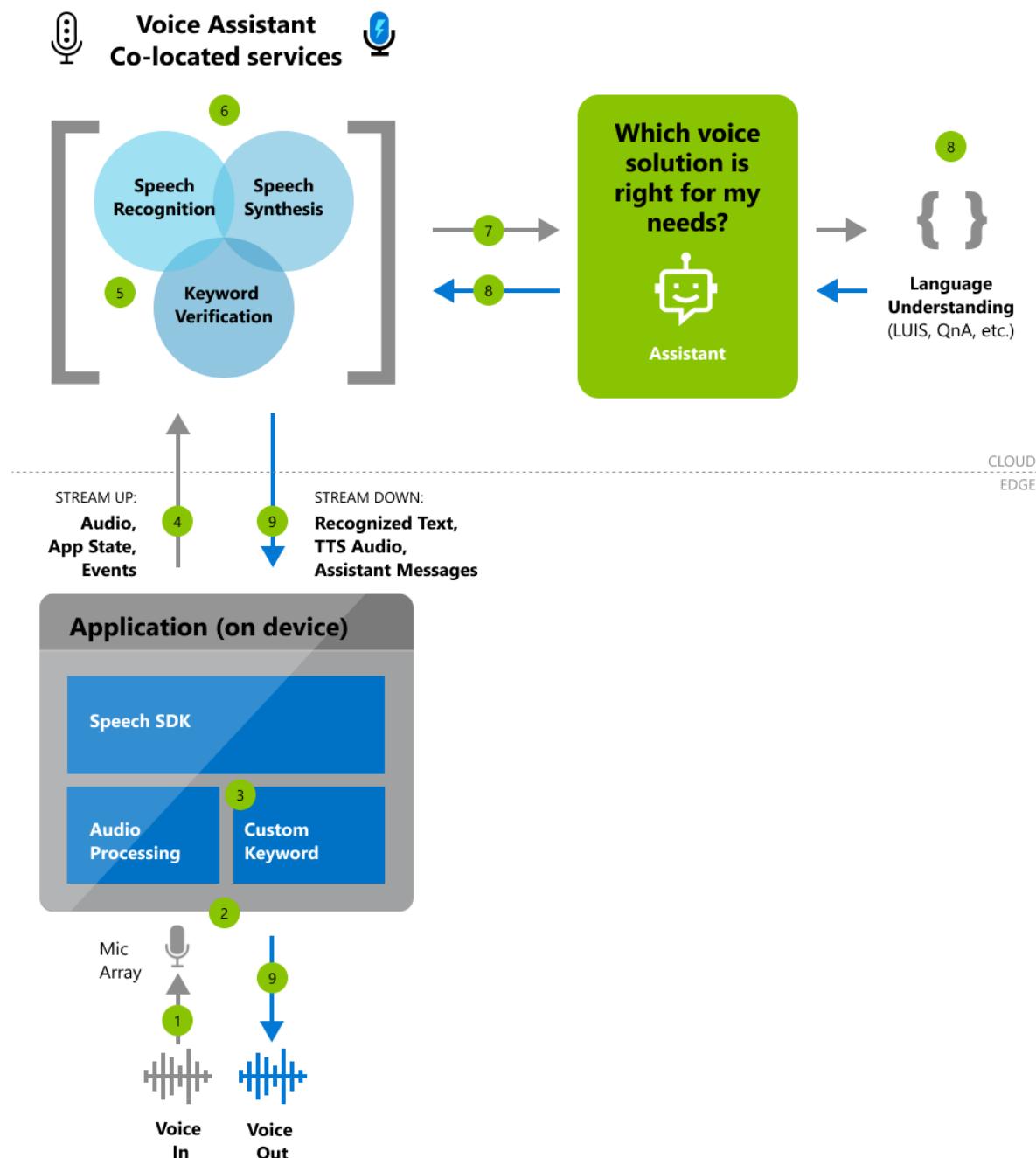
# Acerca de los asistentes de voz

13/01/2020 • 7 minutes to read • [Edit Online](#)

Los asistentes de voz que utilizan el servicio de voz permiten a los desarrolladores crear interfaces de conversación naturales, similares a la humana, para sus aplicaciones y experiencias.

El servicio del asistente de voz proporciona una interacción rápida y confiable entre un dispositivo y una implementación de asistente que usa (1) el canal de voz de Direct Line Speech de Bot Framework o (2) el servicio integrado de comandos personalizados (versión preliminar) para la finalización de tareas.

Las aplicaciones se conectan al servicio del asistente de voz con el kit de desarrollo de software (SDK) de Voz.



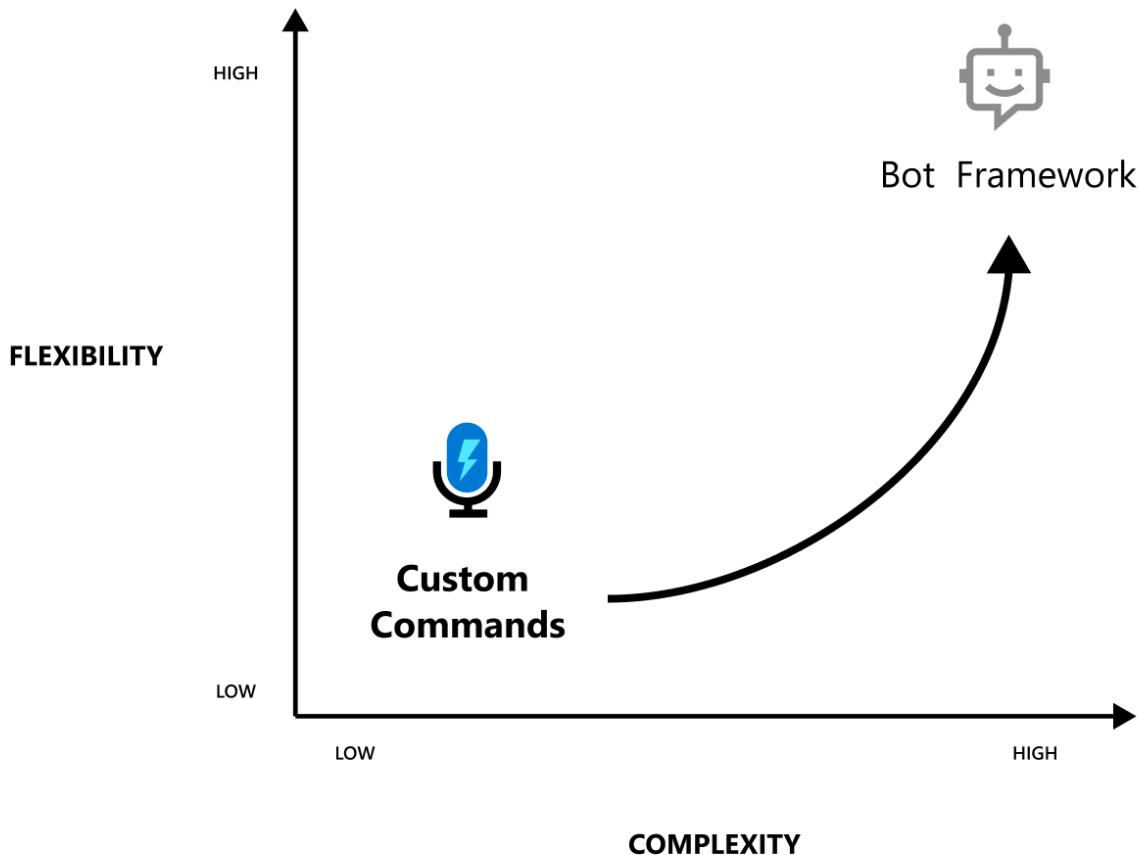
## Elección de una solución de asistente

El primer paso para crear un asistente de voz es decidir qué se debe hacer. El servicio de voz proporciona varias soluciones complementarias para diseñar las interacciones de los asistentes. Si desea la flexibilidad y la versatilidad que proporciona el canal [Direct Line Speech](#) de Bot Framework o la simplicidad de [Custom Commands \(versión preliminar\)](#) para escenarios sencillos, la selección de las herramientas adecuadas le ayudará a comenzar.

SI QUIERE...	CONSIDERE:	POR EJEMPLO...
Conversación abierta con integración de sólidas aptitudes y control completo de la implementación	El canal <a href="#">Direct Line Speech</a> de Bot Framework	<ul style="list-style-type: none"> <li>• "Tengo que ir a Seattle"</li> <li>• "¿Qué tipo de pizza puedo pedir?"</li> </ul>
Conversación orientada a tareas o controles y comandos con la creación y el hospedaje simplificados	<a href="#">Comandos personalizados (versión preliminar)</a>	<ul style="list-style-type: none"> <li>• "Encender la luz superior"</li> <li>• "Subir la temperatura 5 grados"</li> </ul>

Se recomienda [Direct Line Speech](#) como mejor opción predeterminada si aún no está seguro de lo que le gustaría que controlara el asistente. Ofrece integración con un completo conjunto de herramientas y ayudas de creación, como la [solución de asistente virtual y de plantilla de empresa](#) y el servicio de [QnA Maker](#) para crear patrones comunes y usar los orígenes de conocimiento existentes.

[Custom Commands \(versión preliminar\)](#) proporciona una experiencia de creación y hospedaje simplificada específicamente adaptada a escenarios de control y comandos de lenguaje natural.



## Características principales

Tanto si elige [Direct Line Speech](#) o [Custom Commands \(versión preliminar\)](#) para crear sus interacciones con el asistente, puede usar un amplio conjunto de características de personalización para personalizar el asistente para

la marca, el producto y la personalidad.

CATEGORY	CARACTERÍSTICAS
Palabra clave personalizada	Los usuarios pueden iniciar conversaciones con los asistentes mediante una palabra clave personalizada como "Hola, Contoso". Una aplicación lo lleva a cabo con un motor de palabras clave personalizadas en el SDK de Voz, que puede configurarse con una palabra clave <a href="#">que puede generar aquí</a> . Los asistentes de voz pueden utilizar la comprobación de palabras clave del lado del servicio para mejorar la precisión de la activación de palabras clave (frente al dispositivo por sí solo).
Speech to Text	Los asistentes de voz convierten audio en tiempo real en texto reconocido mediante la <a href="#">Conversión de voz en texto</a> del servicio de voz. Este texto está disponible, a medida que se escribe, tanto para la implementación del asistente como para la aplicación cliente.
Texto a voz	Las respuestas textuales desde el asistente se sintetizan mediante <a href="#">Texto a voz</a> del servicio de voz. A continuación, esta síntesis se pone a disposición de la aplicación cliente como una secuencia de audio. Microsoft ofrece la posibilidad de crear su propia voz TTS neuronal personalizada de alta calidad que le pone voz a su marca. Para obtener más información, <a href="#">póngase en contacto con nosotros</a> .

## Introducción a los asistentes de voz

Le ofrecemos inicios rápidos diseñados para que ejecute el código en menos de 10 minutos. Esta tabla incluye una lista de inicios rápidos para asistente de voz ordenados por idioma.

GUÍA DE INICIO RÁPIDO	PLATAFORMA	REFERENCIA DE API
C#, UWP	Windows	<a href="#">Browse</a>
Java	Windows, macOS, Linux	<a href="#">Browse</a>
Java	Android	<a href="#">Browse</a>

## Código de ejemplo

El código de ejemplo para crear un asistente de voz está disponible en GitHub. Estos ejemplos abarcan la aplicación cliente para conectarse al asistente en varios lenguajes de programación conocidos.

- [Ejemplos del asistente de voz \(SDK\)](#)
- [Tutorial: Habilitación del asistente de voz mediante el SDK de Voz, C#](#)

## Tutorial

Tutorial sobre cómo [habilitar la voz para el asistente mediante el SDK de Voz y el canal Direct Line Speech](#).

## Personalización

Los asistentes de voz creados mediante el servicio de voz pueden usar la amplia variedad de opciones de

personalización disponibles para la [conversión de voz en texto](#), la [conversión de texto en voz](#) y la [selección de palabras clave personalizadas](#).

**NOTE**

Las opciones de personalización varían según el idioma o la configuración regional (consulte los [idiomas admitidos](#)).

## Documentos de referencia

- [Acerca del SDK de Voz](#)
- [Azure Bot Service](#)

## Pasos siguientes

- [Obtenga una clave de suscripción gratuita a los servicios de Voz](#)
- [Obtención del SDK de voz](#)
- [Más información sobre los comandos personalizados \(versión preliminar\)](#)
- [Más información sobre Direct Line Speech](#)

# Comandos personalizados (versión preliminar)

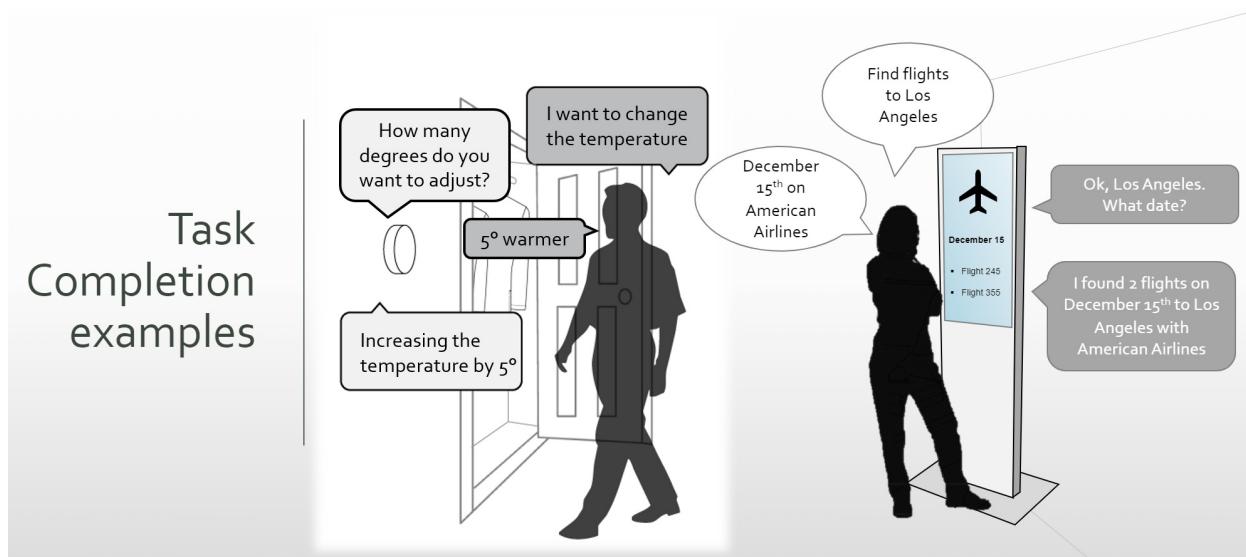
13/01/2020 • 4 minutes to read • [Edit Online](#)

Los asistentes de voz escuchan a los usuarios y realizan una acción en respuesta, lo que a menudo implica una respuesta. Utilizan [conversión de voz en texto](#) para transcribir la voz del usuario y, a continuación, realizan una acción en función de su comprensión del lenguaje natural del texto. Esta acción suele incluir una respuesta hablada del asistente generada con la [conversión de texto en voz](#). Los dispositivos se conectan con los asistentes mediante el objeto `DialogServiceConnector` del SDK de Voz.

**Comandos personalizados (versión preliminar)** es una solución simplificada para crear un asistente de voz. Proporciona una experiencia de creación unificada, un modelo de hospedaje automático y una complejidad relativamente inferior en comparación con otras opciones de creación de asistentes como [Direct Line Speech](#). Sin embargo, esta simplificación conlleva una reducción en la flexibilidad. Por lo tanto, los comandos personalizados (versión preliminar) son más adecuados para escenarios de finalización de tareas o de comando y control. Es especialmente adecuado para los dispositivos de Internet de las cosas (IoT) y sin periféricos.

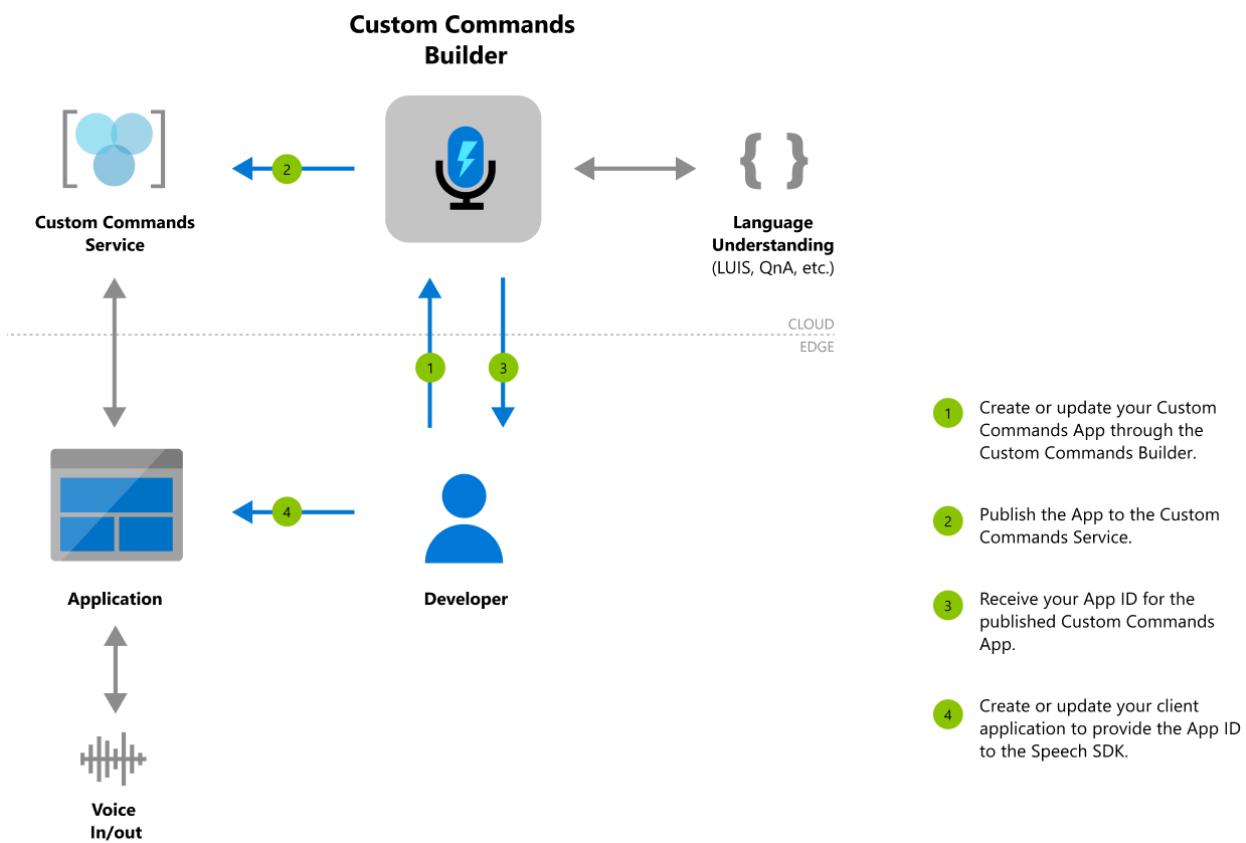
Para una interacción e integración de conversaciones complejas con otras soluciones como la [solución de Virtual Assistant y la plantilla empresarial](#), se recomienda usar Direct Line Speech.

Los candidatos adecuados para los comandos personalizados (versión preliminar) tienen un vocabulario fijo con conjuntos de variables bien definidos. Por ejemplo, las tareas de automatización del hogar, como el control de un termostato, son ideales.



## Introducción a los comandos personalizados (versión preliminar)

El primer paso para usar comandos personalizados (versión preliminar) para realizar un asistente de voz es [obtener una clave de suscripción de voz](#) y acceder al generador de comandos personalizados (versión preliminar) en [Speech Studio](#). Desde allí, puede crear una nueva aplicación de comandos personalizados (versión preliminar) y publicarla, tras lo cual una aplicación en el dispositivo podrá comunicarse con ella mediante el SDK de Voz.



Le ofrecemos inicios rápidos diseñados para que ejecute el código en menos de 10 minutos.

- [Creación de una aplicación de comandos personalizados \(versión preliminar\)](#)
- [Creación de una aplicación de comandos personalizados \(versión preliminar\) con parámetros](#)
- [Conexión a una aplicación de comandos personalizados \(versión preliminar\) con el SDK de Voz, C#](#)

## Código de ejemplo

El código de ejemplo para crear un asistente de voz con los comandos personalizados (versión preliminar) está disponible en GitHub.

- [Ejemplos del asistente de voz \(SDK\)](#)

## Personalización

Los asistentes de voz creados mediante el servicio de voz pueden usar la amplia variedad de opciones de personalización disponibles para la [conversión de voz en texto](#), la [conversión de texto en voz](#) y la [selección de palabras clave personalizadas](#).

### NOTE

Las opciones de personalización varían según el idioma o la configuración regional (consulte los [idiomas admitidos](#)).

## Documentos de referencia

- [Speech SDK](#)

## Pasos siguientes

- [Obtenga una clave de suscripción gratuita a los servicios de Voz](#)

- Obtención del SDK de voz

# Acerca de Direct Line Speech

13/01/2020 • 4 minutes to read • [Edit Online](#)

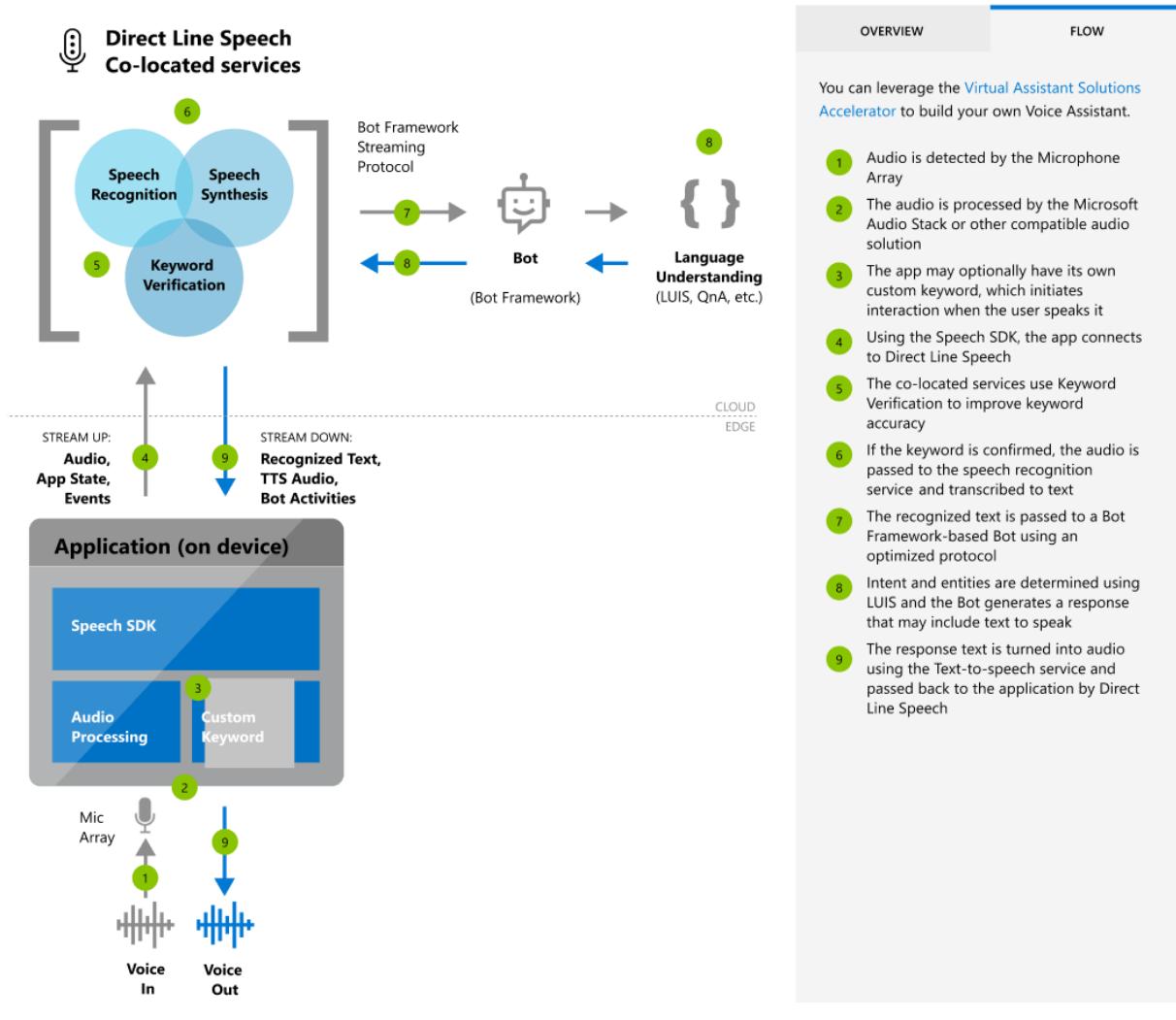
Los asistentes de voz escuchan a los usuarios y realizan una acción en respuesta, lo que a menudo implica una respuesta. Utilizan [conversión de voz en texto](#) para transcribir la voz del usuario y, a continuación, realizan una acción en función de su comprensión del lenguaje natural del texto. Esta acción suele incluir una respuesta hablada del asistente generada con la [conversión de texto en voz](#). Los dispositivos se conectan con los asistentes mediante el objeto `DialogServiceConnector` del SDK de Voz.

**Direct Line Speech** es una solución sólida y completa para crear un asistente de voz flexible y extensible, con la tecnología de Bot Framework y su canal de Direct Line Speech, que está optimizado para la interacción de conversaciones con bots.

Direct Line Speech ofrece los niveles más altos de personalización y sofisticación para los asistentes de voz. Es adecuado para escenarios de conversaciones no estructuradas, naturales o híbridas con la finalización de tareas o el uso de comandos y controles. Este alto grado de flexibilidad implica una mayor complejidad y los escenarios que se limitan a tareas bien definidas mediante la entrada de lenguaje natural quizás prefieran considerar los [comandos personalizados \(versión preliminar\)](#) para una experiencia de solución simplificada.

## Introducción a Direct Line Speech

Los primeros pasos para crear un asistente de voz con Direct Line Speech son [obtener una clave de suscripción de voz](#), crear un nuevo bot asociado a esa suscripción y conectar el bot al canal de Direct Line Speech.



Para obtener una guía paso a paso completa sobre la creación de un asistente de voz sencillo mediante Direct Line Speech, consulte [el tutorial para habilitar el bot con voz mediante el SDK de voz y el canal de Direct Line Speech](#).

También se ofrecen inicios rápidos diseñados para que ejecute el código en menos de 10 minutos. Esta tabla incluye una lista de inicios rápidos para asistente de voz ordenados por idioma.

GUÍA DE INICIO RÁPIDO	PLATAFORMA	REFERENCIA DE API
C#, UWP	Windows	<a href="#">Browse</a>
Java	Windows, macOS, Linux	<a href="#">Browse</a>
Java	Android	<a href="#">Browse</a>

## Código de ejemplo

El código de ejemplo para crear un asistente de voz está disponible en GitHub. Estos ejemplos abarcan la aplicación cliente para conectarse al asistente en varios lenguajes de programación conocidos.

- [Ejemplos del asistente de voz \(SDK\)](#)
- [Tutorial: Habilitación del asistente de voz mediante el SDK de Voz, C#](#)

## Personalización

Los asistentes de voz creados mediante el servicio de voz pueden usar la amplia variedad de opciones de

personalización disponibles para la [conversión de voz en texto](#), la [conversión de texto en voz](#) y la [selección de palabras claves personalizadas](#).

**NOTE**

Las opciones de personalización varían según el idioma o la configuración regional (consulte los [idiomas admitidos](#)).

Direct Line Speech y su funcionalidad asociada para los asistentes de voz constituyen un complemento perfecto para la [solución Virtual Assistant y la plantilla empresarial](#). Aunque Direct Line Speech puede funcionar con cualquier bot compatible, estos recursos ofrecen una línea de base reutilizable para experiencias conversacionales de alta calidad, así como habilidades y modelos complementarios comunes para comenzar rápidamente.

## Documentos de referencia

- [Speech SDK](#)
- [Azure Bot Service](#)

## Pasos siguientes

- [Obtenga una clave de suscripción gratuita a los servicios de Voz](#)
- [Obtención del SDK de voz](#)
- [Creación e implementación de un bot básico](#)
- [Obtención de la solución Virtual Assistant y la plantilla empresarial](#)

# Inicio rápido: Creación de un asistente de voz con el SDK de Voz, UWP

13/01/2020 • 18 minutes to read • [Edit Online](#)

También hay guías de inicio rápido para el [reconocimiento, la síntesis y la traducción de voz](#).

En este artículo, desarrollará una aplicación para Plataforma universal de Windows (UWP) de C# mediante el [SDK de Voz](#). El programa se conectará a un bot previamente creado y configurado para permitir una experiencia de asistente de voz desde la aplicación cliente. La aplicación se compila con el [paquete NuGet del SDK de voz](#) y Microsoft Visual Studio 2019 (cualquier edición).

## NOTE

La Plataforma universal de Windows permite desarrollar aplicaciones que se ejecutan en cualquier dispositivo que admite Windows 10, incluidos PC, Xbox, Surface Hub y otros dispositivos.

## Requisitos previos

Esta guía de inicio rápido requiere:

- [Visual Studio 2019](#).
- Una clave de suscripción de Azure para el servicio Voz. [Obtenga una gratis](#) o créela en [Azure Portal](#).
- Un bot creado previamente y configurado con el [canal Direct Line Speech](#).

## NOTE

Consulte [la lista de regiones admitidas para los asistentes de voz](#) y asegúrese de que sus recursos se implementan en una de esas regiones.

## Opcional: Empiece rápidamente

En este inicio rápido se describirá, paso a paso, cómo hacer que una aplicación cliente se conecte al bot habilitado para voz. Si prefiere sumergirse de lleno, el código fuente completo y listo para compilar utilizado en este inicio rápido está disponible en los [ejemplos del SDK de Voz](#) bajo la carpeta `quickstart`.

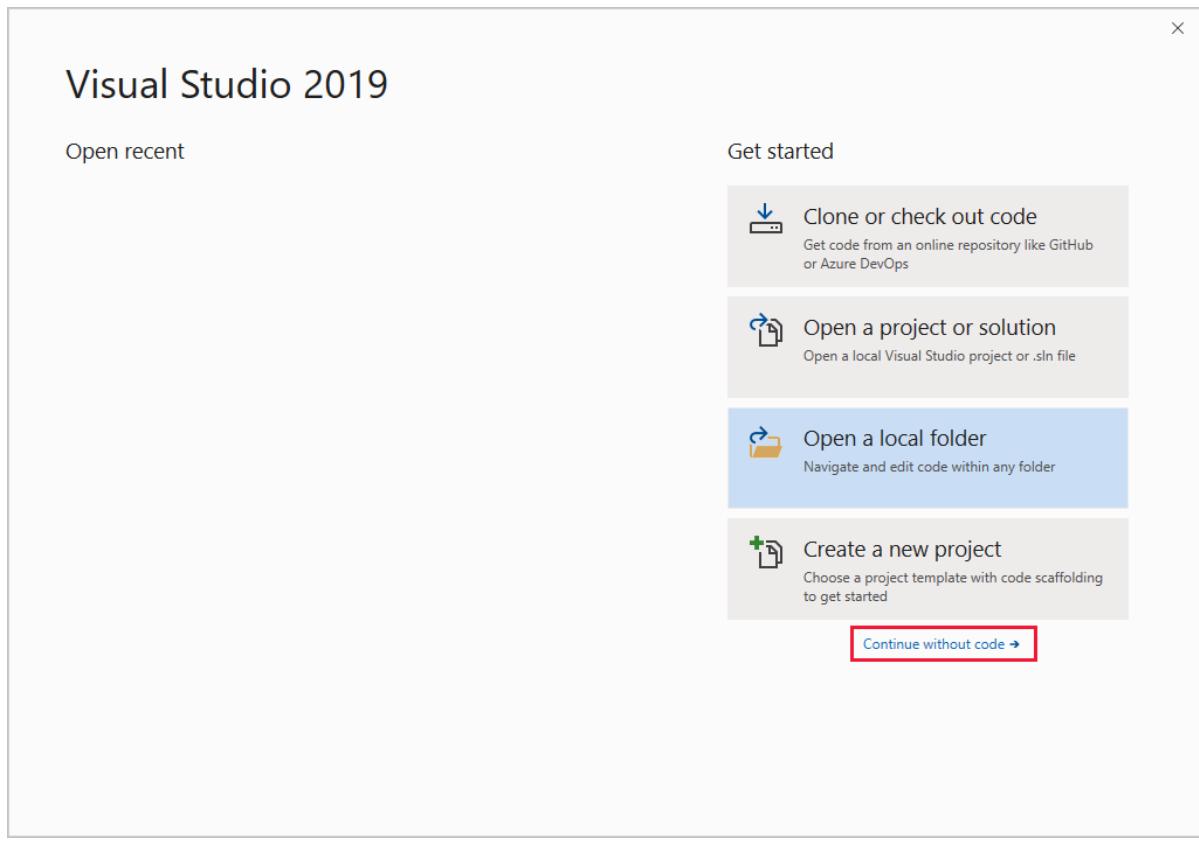
## Creación de un proyecto de Visual Studio

Para crear un proyecto de Visual Studio C++ para el desarrollo de la Plataforma universal de Windows (UPW), debe configurar las opciones de desarrollo de Visual Studio, crear el proyecto, seleccionar la arquitectura de destino, configurar la captura de audio e instalar el SDK de voz.

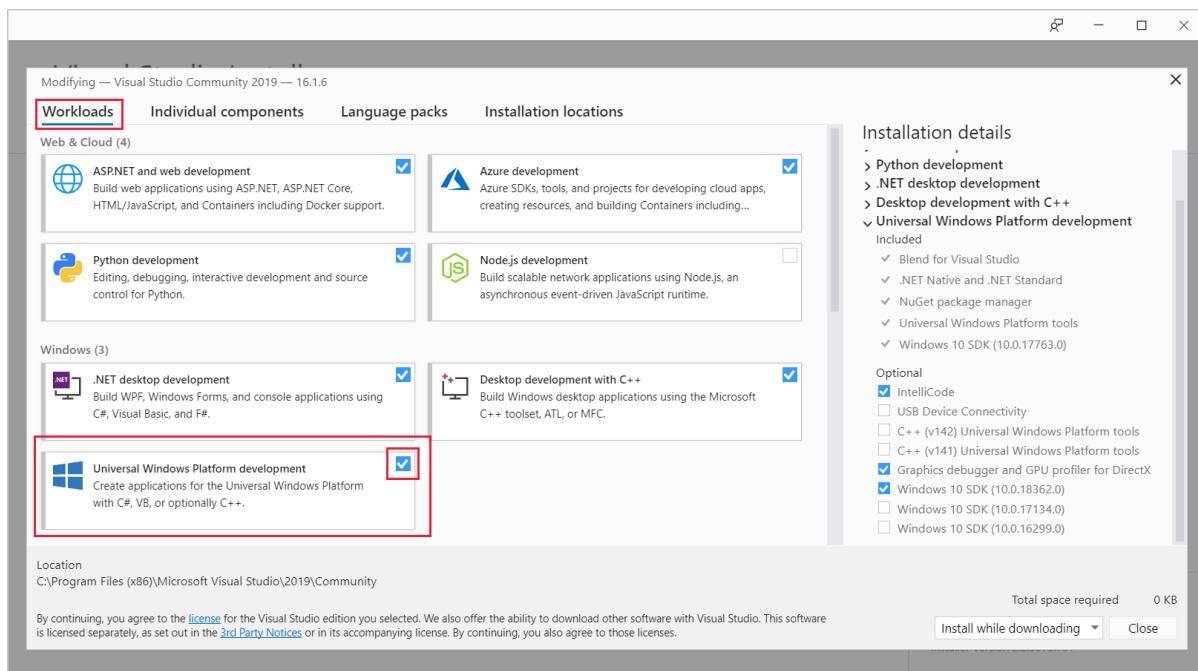
### Configuración de las opciones de desarrollo de Visual Studio

Para empezar, asegúrese de que Visual Studio está configurado correctamente para el desarrollo de UWP:

1. Abra Visual Studio 2019 para mostrar la ventana **Inicio**.



2. Seleccione **Continuar sin código** para ir al IDE de Visual Studio.
3. En la barra de menús de Visual Studio, seleccione **Herramientas > Get Tools and Features** (Obtener herramientas y características) para abrir el Instalador de Visual Studio y ver el cuadro de diálogo **Modificar**.

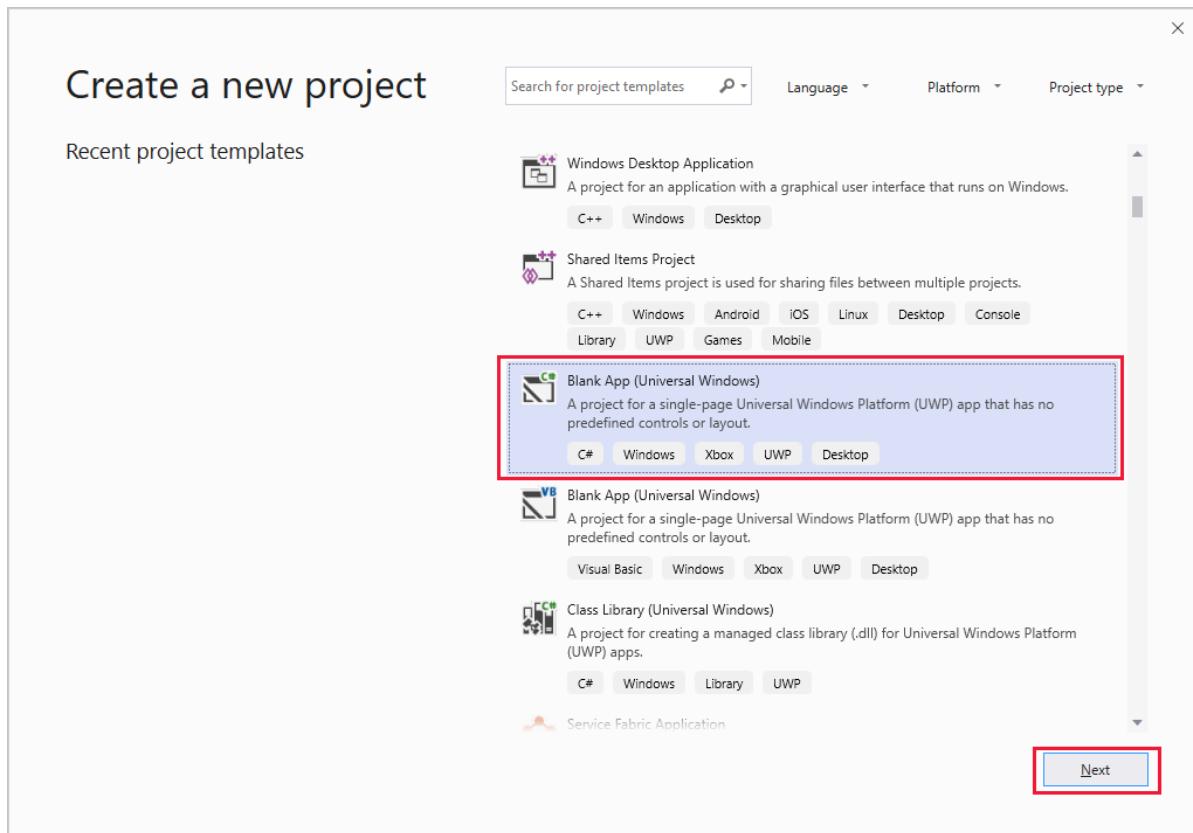


4. En la pestaña **Cargas de trabajo**, en **Windows**, busque la carga de trabajo **Desarrollo de la Plataforma universal de Windows**. Si la casilla situada junto a esa carga de trabajo ya está seleccionada, cierre el cuadro de diálogo **Modificar** y vaya al paso 6.
5. Active la casilla **Desarrollo de la Plataforma universal de Windows**, seleccione **Modificar** y, a continuación, en el cuadro de diálogo **Antes de comenzar**, seleccione **Continuar** para instalar la carga de trabajo de desarrollo de UWP. La instalación de la nueva característica puede tardar un rato.
6. Cierre el Instalador de Visual Studio.

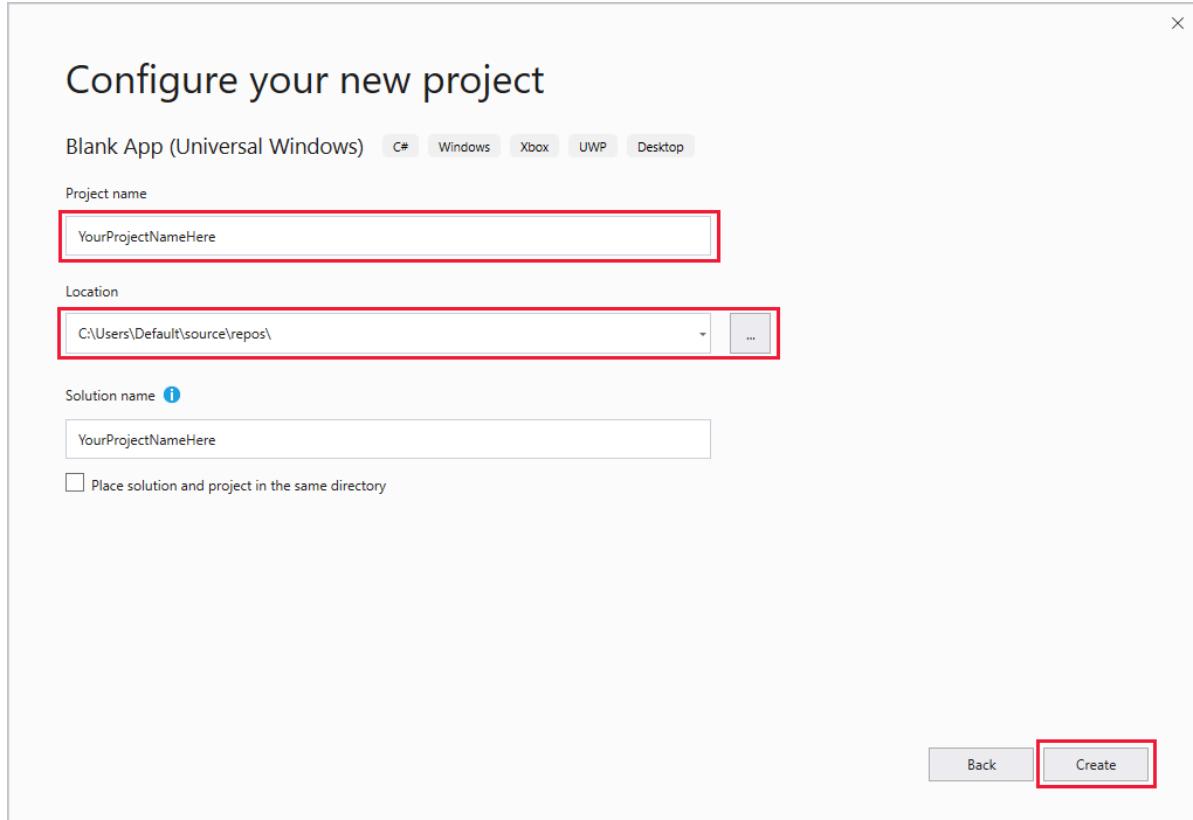
## Cree el proyecto y seleccione la arquitectura de destino.

Después, cree el proyecto.

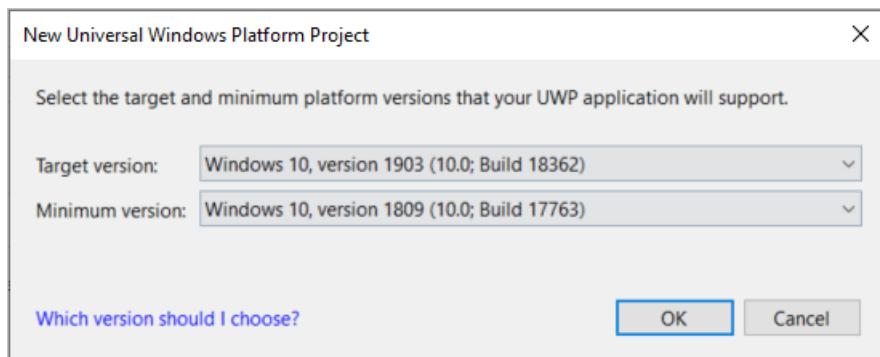
1. En la barra de menús de Visual Studio, elija **Archivo > Nuevo > Proyecto** para mostrar la ventana **Crear un nuevo proyecto**.



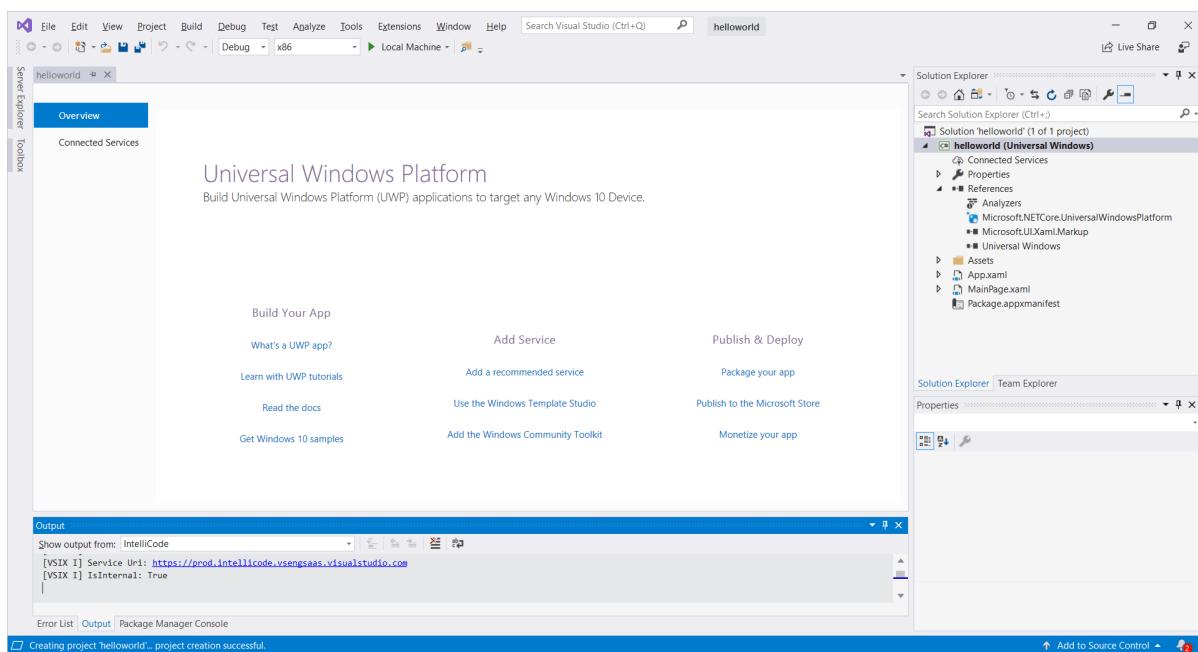
2. Busque y seleccione **Aplicación vacía (Windows universal)**. Asegúrese de seleccionar la versión de C# de este tipo de proyecto (en lugar de Visual Basic).
3. Seleccione **Siguiente** para abrir la pantalla **Configurar el nuevo proyecto**.



4. En **Nombre del proyecto**, escriba `helloworld`.
5. En **Ubicación**, vaya la carpeta en la que desea guardar el proyecto y selecciónela o créela.
6. Seleccione **Crear** para ir a la ventana **Nuevo proyecto de la Plataforma universal de Windows**.



7. En **Versión mínima** (el segundo cuadro desplegable), elija **Windows 10 Fall Creators Update (10.0; Compilación 16299)**, que es el requisito mínimo para el SDK de voz.
8. En **Versión de destino** (el primer cuadro desplegable), elija un valor idéntico o posterior al valor de **Versión mínima**.
9. Seleccione **Aceptar**. Se le devolverá al IDE de Visual Studio, con el nuevo proyecto creado y visible en el panel **Explorador de soluciones**.



Ahora seleccione la arquitectura de la plataforma de destino. En la barra de herramientas de Visual Studio, busque el cuadro desplegable **Plataformas de solución**. (Si no lo ve, elija **Ver > Barra de herramientas > Estándar** para mostrar la barra de herramientas que contiene **Plataformas de solución**). Si está ejecutando Windows de 64 bits, elija **x64** en el cuadro desplegable. Windows de 64 bits puede ejecutar también aplicaciones de 32 bits, por lo que puede elegir **x86**, si lo prefiere.

#### NOTE

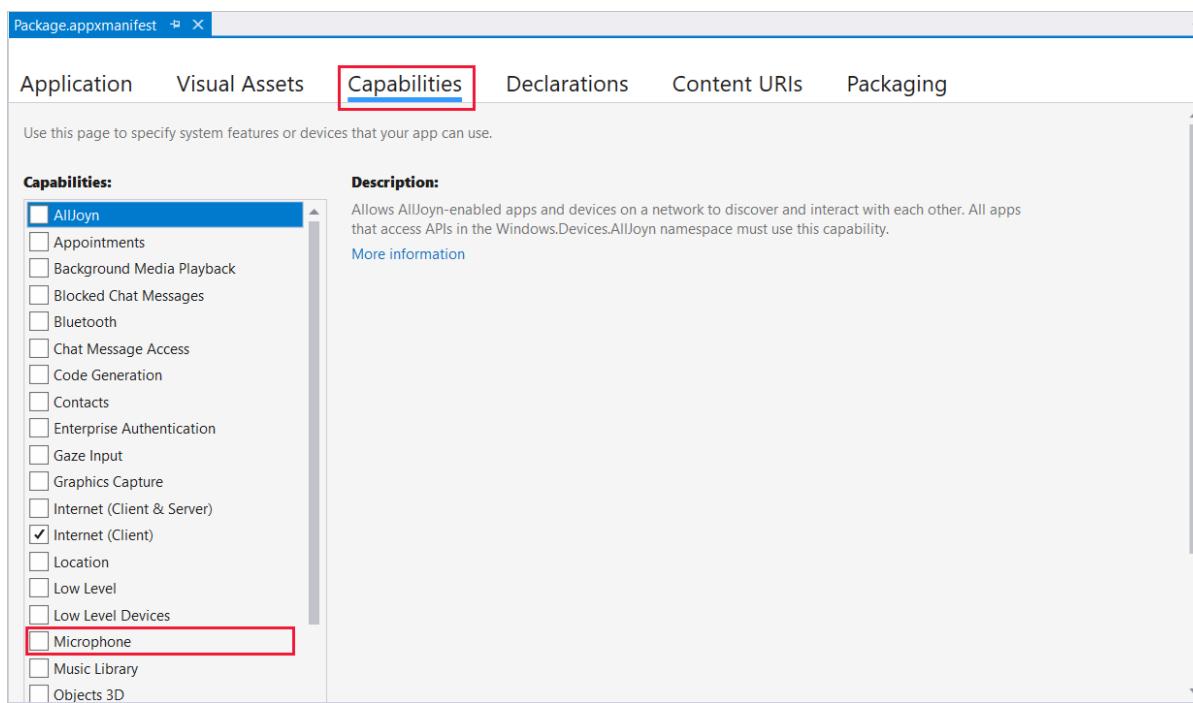
El SDK de Voz solo admite los procesadores compatibles con Intel. Actualmente no se admiten los procesadores ARM.

## Configuración de la captura de audio

A continuación, permita que el proyecto capture la entrada de audio:

1. En el **Explorador de soluciones**, haga doble clic en **Package.appxmanifest** para abrir el manifiesto de aplicación del paquete.

2. Seleccione la pestaña **Funcionalidades**.



3. Active la casilla de la funcionalidad **Micrófono**.

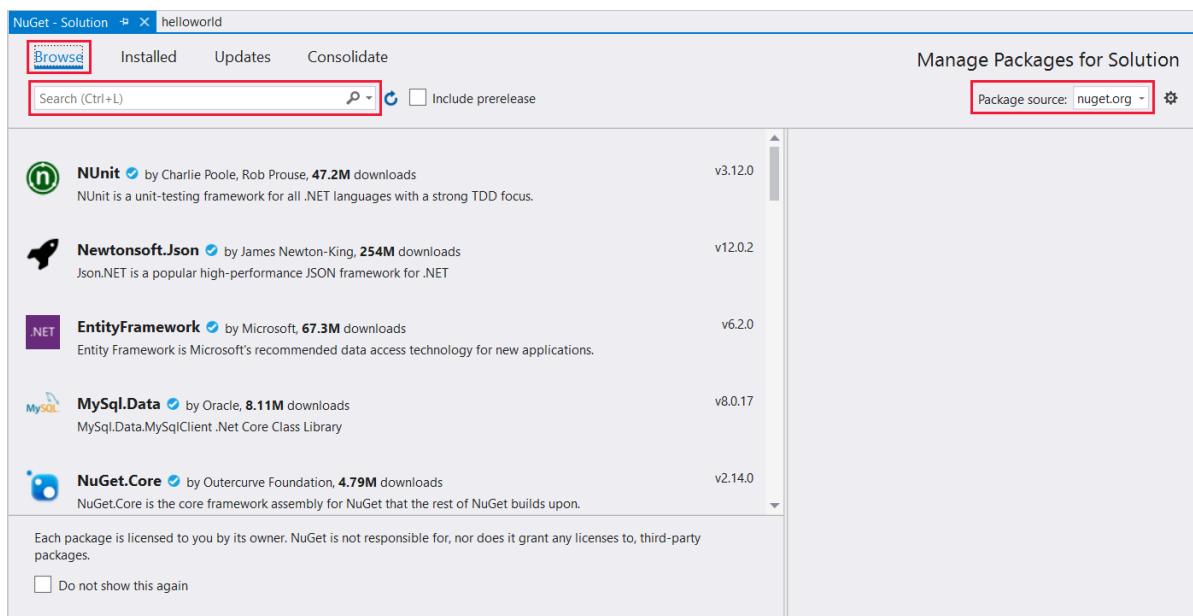
4. En la barra de menús, elija **Archivo > Guardar Package.appxmanifest** para guardar los cambios.

## Instalación de Speech SDK

Por último, instale el [paquete NuGet del SDK de voz](#) y haga referencia al SDK de voz en el proyecto:

1. En el **Explorador de soluciones**, haga clic con el botón derecho en la solución y elija **Manage NuGet Packages for Solution** (Administrar paquetes NuGet para la solución) para ir a la ventana **NuGet: solución**.

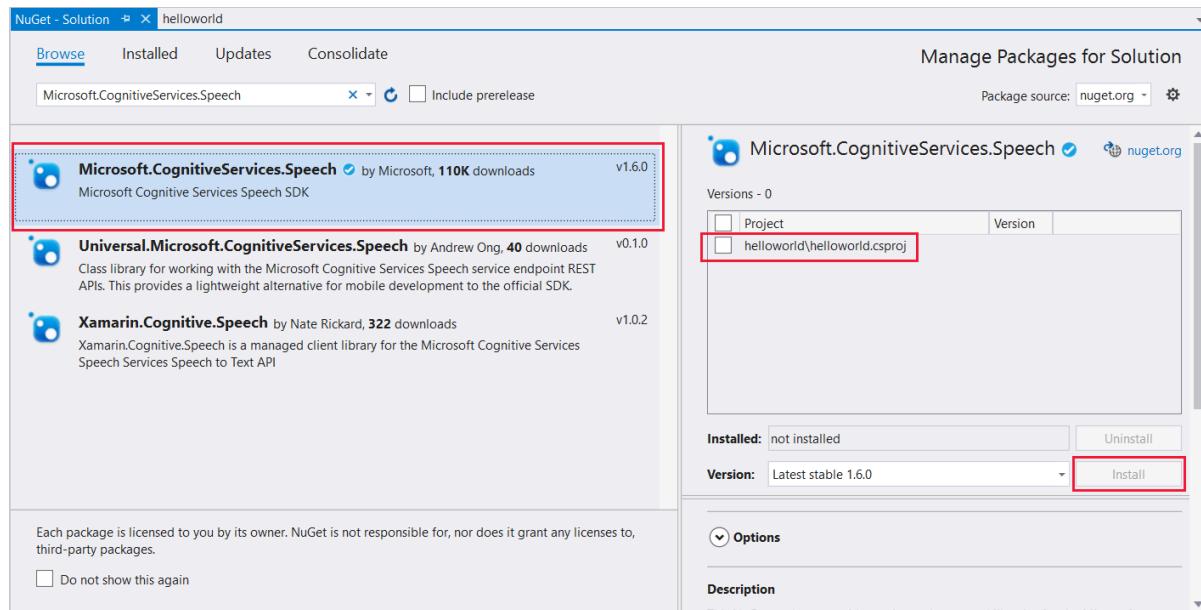
2. Seleccione **Examinar**.



3. En **Origen del paquete**, elija **nuget.org**.

4. En el cuadro de búsqueda **Buscar**, escriba `Microsoft.CognitiveServices.Speech` y, a continuación, elija el

paquete cuando aparezca en los resultados de la búsqueda.



5. En el panel de estado del paquete situado junto a los resultados de la búsqueda, seleccione el proyecto **HelloWorld**.
6. Seleccione **Instalar**.
7. En el cuadro de diálogo **Vista previa de los cambios**, seleccione **Aceptar**.
8. En el cuadro de diálogo **Aceptación de licencia**, vea la licencia y, a continuación, seleccione **Acepto**. La instalación del paquete comienza y, cuando se completa, el panel **Salida** muestra un mensaje similar al siguiente: `Successfully installed 'Microsoft.CognitiveServices.Speech 1.7.0' to helloworld`.

## Incorporación de código de ejemplo

Ahora, agregue el código XAML que define la interfaz de usuario de la aplicación y agregue la implementación de código C# subyacente.

### Código XAML

En primer lugar, creará la interfaz de usuario de la aplicación; para ello, agregará el código XAML:

1. En el **Explorador de soluciones**, abra `MainPage.xaml`.
2. En la vista XAML del diseñador, reemplace todo el contenido por el siguiente fragmento de código:

```

<Page
    x:Class="helloworld.MainPage"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:local="using:helloworld"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    mc:Ignorable="d"
    Background="{ThemeResource ApplicationPageBackgroundThemeBrush}">

    <Grid>
        <StackPanel Orientation="Vertical" HorizontalAlignment="Center"
            Margin="20,50,0,0" VerticalAlignment="Center" Width="800">
            <Button x:Name="EnableMicrophoneButton" Content="Enable Microphone"
                Margin="0,0,10,0" Click="EnableMicrophone_ButtonClicked"
                Height="35"/>
            <Button x:Name="ListenButton" Content="Talk to your bot"
                Margin="0,10,10,0" Click="ListenButton_ButtonClicked"
                Height="35"/>
            <StackPanel x:Name="StatusPanel" Orientation="Vertical"
                RelativePanel.AlignBottomWithPanel="True"
                RelativePanel.AlignRightWithPanel="True"
                RelativePanel.AlignLeftWithPanel="True">
                <TextBlock x:Name="StatusLabel" Margin="0,10,10,0"
                    TextWrapping="Wrap" Text="Status:" FontSize="20"/>
                <Border x:Name="StatusBorder" Margin="0,0,0,0">
                    <ScrollViewer VerticalScrollMode="Auto"
                        VerticalScrollBarVisibility="Auto" MaxHeight="200">
                        <!-- Use LiveSetting to enable screen readers to announce
                            the status update. -->
                    <TextBlock
                        x:Name="StatusBlock" FontWeight="Bold"
                        AutomationProperties.LiveSetting="Assertive"
                        MaxWidth="{Binding ElementName=Splitter, Path=ActualWidth}"
                        Margin="10,10,10,20" TextWrapping="Wrap" />
                    </ScrollViewer>
                </Border>
            </StackPanel>
        </StackPanel>
        <MediaElement x:Name="mediaElement"/>
    </Grid>
</Page>

```

La vista Diseño se actualiza para mostrar la interfaz de usuario de la aplicación.

## Código subyacente de C#

A continuación, agregue el código subyacente para que la aplicación funcione según lo previsto. El código fuente subyacente incluye:

- Instrucciones `using` para los espacios de nombres `Speech` y `Speech.Dialog`.
- Una implementación sencilla para garantizar el acceso del micrófono, conectado a un controlador de botón
- Asistentes básicos de la interfaz de usuario para presentar los errores y mensajes en la aplicación
- Un punto de aterrizaje para la ruta de acceso del código de inicialización que se rellenará más tarde
- Una asistente para la reproducción de texto a voz (sin compatibilidad con streaming)
- Un controlador de botón vacía para empezar a escuchar que se rellenará más tarde

Para agregar el código fuente subyacente, siga estos pasos:

1. En **Explorador de soluciones**, abra el archivo de código fuente subyacente `MainPage.xaml.cs`. (Se agrupa en `MainPage.xaml`).
2. Reemplace el contenido del archivo por el siguiente fragmento de código:

```
using Microsoft.CognitiveServices.Speech;
using Microsoft.CognitiveServices.Speech.Audio;
using Microsoft.CognitiveServices.Speech.Dialog;
using System;
using System.Diagnostics;
using System.IO;
using System.Text;
using Windows.Foundation;
using Windows.Storage.Streams;
using Windows.UI.Xaml;
using Windows.UI.Xaml.Controls;
using Windows.UI.Xaml.Media;

namespace helloworld
{
    public sealed partial class MainPage : Page
    {
        private DialogServiceConnector connector;

        private enum NotifyType
        {
            StatusMessage,
            ErrorMessage
        };

        public MainPage()
        {
            this.InitializeComponent();
        }

        private async void EnableMicrophone_ButtonClicked(
            object sender, RoutedEventArgs e)
        {
            bool isMicAvailable = true;
            try
            {
                var mediaCapture = new Windows.Media.Capture.MediaCapture();
                var settings =
                    new Windows.Media.Capture.MediaCaptureInitializationSettings();
                settings.StreamingCaptureMode =
                    Windows.Media.Capture.StreamingCaptureMode.Audio;
                await mediaCapture.InitializeAsync(settings);
            }
            catch (Exception)
            {
                isMicAvailable = false;
            }
            if (!isMicAvailable)
            {
                await Windows.System.Launcher.LaunchUriAsync(
                    new Uri("ms-settings:privacy-microphone"));
            }
            else
            {
                NotifyUser("Microphone was enabled", NotifyType.StatusMessage);
            }
        }

        private void NotifyUser(
            string strMessage, NotifyType type = NotifyType.StatusMessage)
        {
            // If called from the UI thread, then update immediately.
            // Otherwise, schedule a task on the UI thread to perform the update.
            if (Dispatcher.HasThreadAccess)
            {
                UpdateStatus(strMessage, type);
            }
            else

```

```

    {
        var task = Dispatcher.RunAsync(
            Windows.UI.Core.CoreDispatcherPriority.Normal,
            () => UpdateStatus(strMessage, type));
    }
}

private void UpdateStatus(string strMessage, NotifyType type)
{
    switch (type)
    {
        case NotifyType.StatusMessage:
            StatusBorder.Background = new SolidColorBrush(
                Windows.UI.Colors.Green);
            break;
        case NotifyType.ErrorMessage:
            StatusBorder.Background = new SolidColorBrush(
                Windows.UI.Colors.Red);
            break;
    }
    StatusBlock.Text += string.IsNullOrEmpty(StatusBlock.Text)
        ? strMessage : "\n" + strMessage;

    if (!string.IsNullOrEmpty(StatusBlock.Text))
    {
        StatusBorder.Visibility = Visibility.Visible;
        StatusPanel.Visibility = Visibility.Visible;
    }
    else
    {
        StatusBorder.Visibility = Visibility.Collapsed;
        StatusPanel.Visibility = Visibility.Collapsed;
    }
    // Raise an event if necessary to enable a screen reader
    // to announce the status update.
    var peer =
Windows.UI.Xaml.Automation.Peers.FrameworkElementAutomationPeer.FromElement(StatusBlock);
    if (peer != null)
    {
        peer.RaiseAutomationEvent(
            Windows.UI.Xaml.AutomationEvents.LiveRegionChanged);
    }
}

// Waits for and accumulates all audio associated with a given
// PullAudioOutputStream and then plays it to the MediaElement. Long spoken
// audio will create extra latency and a streaming playback solution
// (that plays audio while it continues to be received) should be used --
// see the samples for examples of this.
private void SynchronouslyPlayActivityAudio(
    PullAudioOutputStream activityAudio)
{
    var playbackStreamWithHeader = new MemoryStream();
    playbackStreamWithHeader.Write(Encoding.ASCII.GetBytes("RIFF"), 0, 4); // ChunkID
    playbackStreamWithHeader.Write(BitConverter.GetBytes(UInt32.MaxValue), 0, 4); // ChunkSize:
max
    playbackStreamWithHeader.Write(Encoding.ASCII.GetBytes("WAVE"), 0, 4); // Format
    playbackStreamWithHeader.Write(Encoding.ASCII.GetBytes("fmt "), 0, 4); // Subchunk1ID
    playbackStreamWithHeader.Write(BitConverter.GetBytes(16), 0, 4); // Subchunk1Size: PCM
    playbackStreamWithHeader.Write(BitConverter.GetBytes(1), 0, 2); // AudioFormat: PCM
    playbackStreamWithHeader.Write(BitConverter.GetBytes(1), 0, 2); // NumChannels: mono
    playbackStreamWithHeader.Write(BitConverter.GetBytes(16000), 0, 4); // SampleRate: 16kHz
    playbackStreamWithHeader.Write(BitConverter.GetBytes(32000), 0, 4); // ByteRate
    playbackStreamWithHeader.Write(BitConverter.GetBytes(2), 0, 2); // BlockAlign
    playbackStreamWithHeader.Write(BitConverter.GetBytes(16), 0, 2); // BitsPerSample: 16-bit
    playbackStreamWithHeader.Write(Encoding.ASCII.GetBytes("data"), 0, 4); // Subchunk2ID
    playbackStreamWithHeader.Write(BitConverter.GetBytes(UInt32.MaxValue), 0, 4); //
Subchunk2Size
}

```

```

byte[] pullBuffer = new byte[2056];

uint lastRead = 0;
do
{
    lastRead = activityAudio.Read(pullBuffer);
    playbackStreamWithHeader.Write(pullBuffer, 0, (int)lastRead);
}
while (lastRead == pullBuffer.Length);

var task = Dispatcher.RunAsync(
    Windows.UI.Core.CoreDispatcherPriority.Normal, () =>
{
    mediaElement.SetSource(
        playbackStreamWithHeader.AsRandomAccessStream(), "audio/wav");
    mediaElement.Play();
});
}

private void InitializeDialogServiceConnector()
{
    // New code will go here
}

private async void ListenButton_ButtonClicked(
    object sender, RoutedEventArgs e)
{
    // New code will go here
}
}
}

```

3. Agregue el siguiente fragmento de código al cuerpo del método de `InitializeDialogServiceConnector`. Este código crea el `DialogServiceConnector` con la información de la suscripción.

```

// Create a BotFrameworkConfig by providing a Speech service subscription key
// the RecoLanguage property is optional (default en-US)
const string speechSubscriptionKey = "YourSpeechSubscriptionKey"; // Your subscription key
const string region = "YourServiceRegion"; // Your subscription service region.

var botConfig = BotFrameworkConfig.FromSubscription(speechSubscriptionKey, region);
botConfig SetProperty(PropertyId.SpeechServiceConnection_RecoLanguage, "en-US");
connector = new DialogServiceConnector(botConfig);

```

#### **NOTE**

Consulte [la lista de regiones admitidas para los asistentes de voz](#) y asegúrese de que sus recursos se implementan en una de esas regiones.

#### **NOTE**

Para obtener información sobre la configuración del bot y la recuperación del secreto de un canal, consulte la documentación de Bot Framework para el [canal Direct Line Speech](#).

4. Agregue las cadenas `YourChannelSecret`, `YourSpeechSubscriptionKey` y `YourServiceRegion` con sus propios valores para el bot, la suscripción de voz y la [región](#).
5. Anexe el siguiente fragmento de código al final del cuerpo del método de `InitializeDialogServiceConnector`. Este código configura los controladores de los eventos en los que se basa `DialogServiceConnector`

comunicar sus actividades de bot, los resultados del reconocimiento de voz y otra información.

```
// ActivityReceived is the main way your bot will communicate with the client
// and uses bot framework activities
connector.ActivityReceived += (sender, activityReceivedEventArgs) =>
{
    NotifyUser(
        $"Activity received, hasAudio={activityReceivedEventArgs.HasAudio} activity=
{activityReceivedEventArgs.Activity}");

    if (activityReceivedEventArgs.HasAudio)
    {
        SynchronouslyPlayActivityAudio(activityReceivedEventArgs.Audio);
    }
};

// Canceled will be signaled when a turn is aborted or experiences an error condition
connector.Canceled += (sender, canceledEventArgs) =>
{
    NotifyUser($"Canceled, reason={canceledEventArgs.Reason}");
    if (canceledEventArgs.Reason == CancellationReason.Error)
    {
        NotifyUser(
            $"Error: code={canceledEventArgs.ErrorCode}, details={canceledEventArgs.ErrorDetails}");
    }
};

// Recognizing (not 'Recognized') will provide the intermediate recognized text
// while an audio stream is being processed
connector.Recognizing += (sender, recognitionEventArgs) =>
{
    NotifyUser($"Recognizing! in-progress text={recognitionEventArgs.Result.Text}");
};

// Recognized (not 'Recognizing') will provide the final recognized text
// once audio capture is completed
connector.Recognized += (sender, recognitionEventArgs) =>
{
    NotifyUser($"Final speech-to-text result: '{recognitionEventArgs.Result.Text}'");
};

// SessionStarted will notify when audio begins flowing to the service for a turn
connector.SessionStarted += (sender, sessionEventArgs) =>
{
    NotifyUser($"Now Listening! Session started, id={sessionEventArgs.SessionId}");
};

// SessionStopped will notify when a turn is complete and
// it's safe to begin listening again
connector.SessionStopped += (sender, sessionEventArgs) =>
{
    NotifyUser($"Listening complete. Session ended, id={sessionEventArgs.SessionId}");
};
```

6. Agregue el siguiente fragmento de código al cuerpo del método `ListenButton_ButtonClicked` de la clase `MainPage`. Este código configura `DialogServiceConnector` para escuchar, ya que ya ha establecido la configuración y ha registrado los controladores de eventos.

```
if (connector == null)
{
    InitializeDialogServiceConnector();
    // Optional step to speed up first interaction: if not called,
    // connection happens automatically on first use
    var connectTask = connector.ConnectAsync();
}

try
{
    // Start sending audio to your speech-enabled bot
    var listenTask = connector.ListenOnceAsync();

    // You can also send activities to your bot as JSON strings --
    // Microsoft.Bot.Schema can simplify this
    string speakActivity =
        @"{""type"":""message"" , ""text"":""Greeting Message"" , ""speak"":""Hello there!""}";
    await connector.SendActivityAsync(speakActivity);

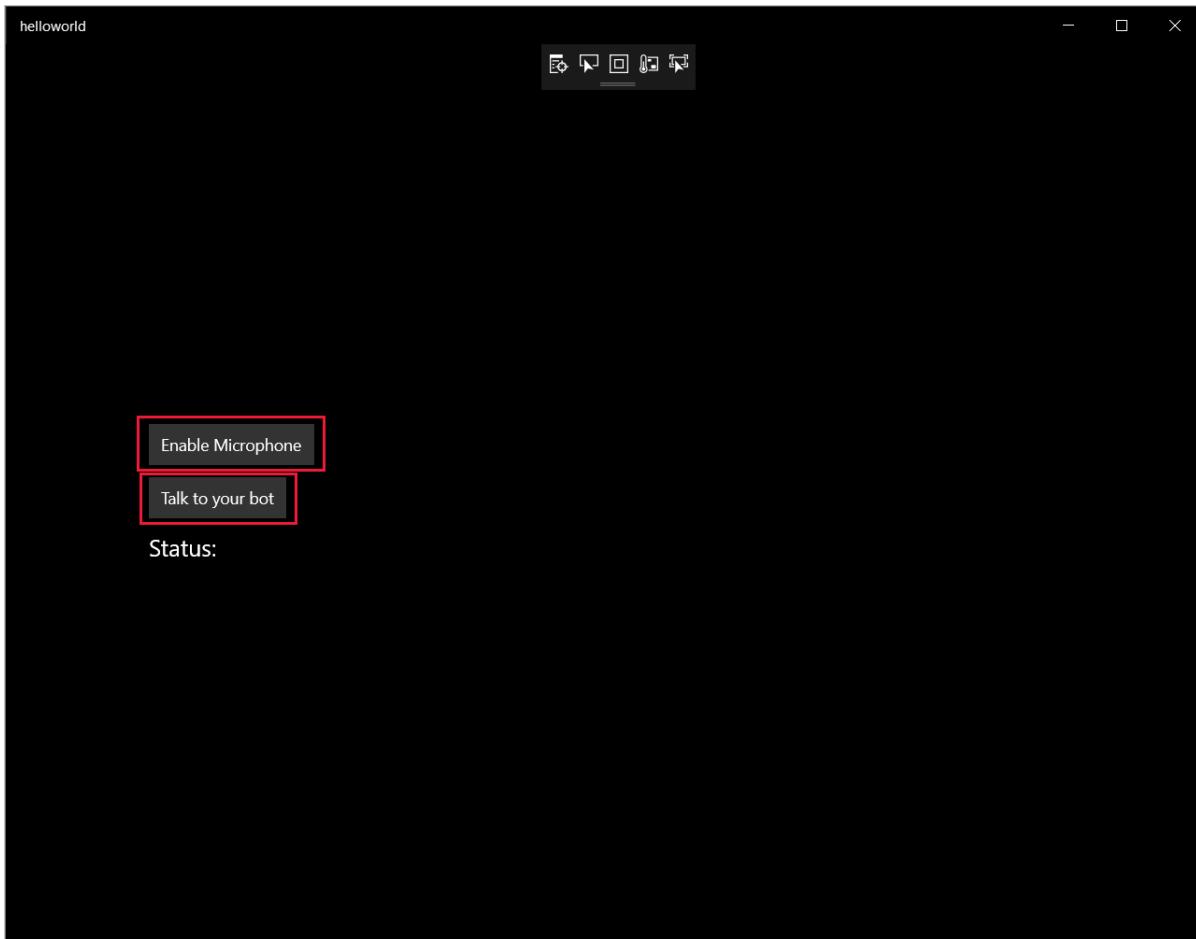
}
catch (Exception ex)
{
    NotifyUser($"Exception: {ex.ToString()}", NotifyType.ErrorMessage);
}
```

7. En la barra de menús, elija **Archivo** > **Guardar todo** para guardar los cambios.

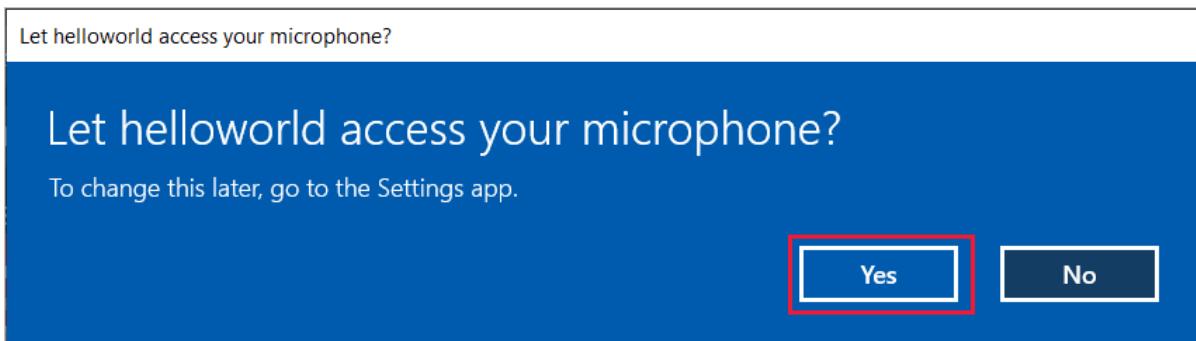
## Compilación y ejecución de la aplicación

Ahora está listo para compilar y probar la aplicación.

1. En la barra de menús, elija **Compilar** > **Compilar solución** para compilar la aplicación. El código se debería compilar sin errores ahora.
2. Elija **Depurar** > **Iniciar depuración** o presione **F5** para iniciar la aplicación. Aparece la ventana **HelloWorld**.



3. Seleccione **Habilitar micrófono** y, cuando aparezca la solicitud de permiso de acceso, seleccione **Sí**.



4. Seleccione **Talk to your bot** (Hablar con el bot) y diga una expresión o frase en inglés en el micrófono del dispositivo. Lo que diga se transmitirá al canal Direct Line Speech y se transcribirá en texto, que aparece en la misma ventana.

## Pasos siguientes

[Creación e implementación de un bot básico](#)

## Otras referencias

- [Acera de los asistentes de voz](#)
- [Obtenga una clave de suscripción gratuita a los servicios de Voz](#)
- [Palabras clave personalizadas](#)
- [Conexión de Direct Line de Voz al bot](#)
- [Exploración de ejemplos de C# en GitHub](#)

# Inicio rápido: Creación de un asistente de voz con el SDK de Voz, Java (versión preliminar)

13/01/2020 • 17 minutes to read • [Edit Online](#)

También hay disponibles inicios rápidos para [conversión de voz en texto](#), [conversión de texto en voz](#) y [traducción de voz](#).

En este artículo creará una aplicación de consola Java mediante el [SDK de Voz de Azure Cognitive Services](#). La aplicación se conectará a un bot previamente creado y configurado para utilizar el canal Direct Line Speech, enviar una solicitud de voz y devolver una actividad de respuesta de voz (si se ha configurado). La aplicación se crea con el paquete de Maven del SDK de Voz y el IDE de Java de Eclipse en Windows, Ubuntu Linux y en macOS. Se ejecuta en un entorno de tiempo de ejecución de Java 8 (JRE) de 64 bits.

## Requisitos previos

Esta guía de inicio rápido requiere:

- Sistema operativo: Windows (64 bits), Ubuntu Linux 16.04/18.04 (64 bits) o macOS 10.13 o superior.
- [IDE de Java de Eclipse](#).
- [Java 8 o JDK 8](#).
- Una clave de suscripción de Azure para el servicio Voz. [Obtenga una gratis](#) o créela en [Azure Portal](#).
- Un bot configurado previamente y creado mediante la versión 4.2 o superior de Bot Framework. El bot tendría que suscribirse al nuevo canal Direct Line Speech para recibir entradas de voz.

### NOTE

Consulte la [lista de regiones admitidas para los asistentes de voz](#) y asegúrese de que sus recursos se implementan en una de esas regiones.

Si ejecuta Ubuntu 16.04 o 18.04, asegúrese de que estén instaladas estas dependencias antes de iniciar Eclipse:

```
sudo apt-get update  
sudo apt-get install build-essential libssl1.0.0 libasound2 wget
```

Si está ejecutando Windows (64 bits), asegúrese de que ha instalado Microsoft Visual C++ Redistributable para su plataforma:

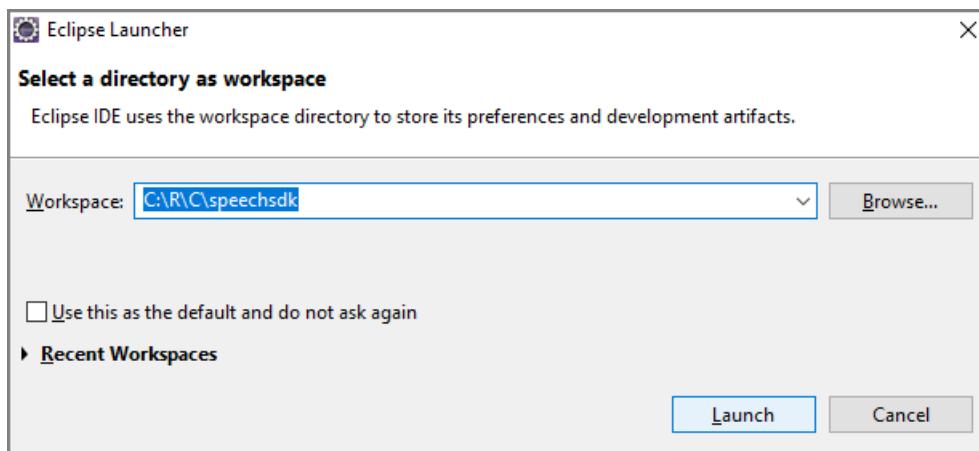
- [Descarga de Microsoft Visual C++ Redistributable para Visual Studio 2017](#)

## Opcional: Empiece rápidamente

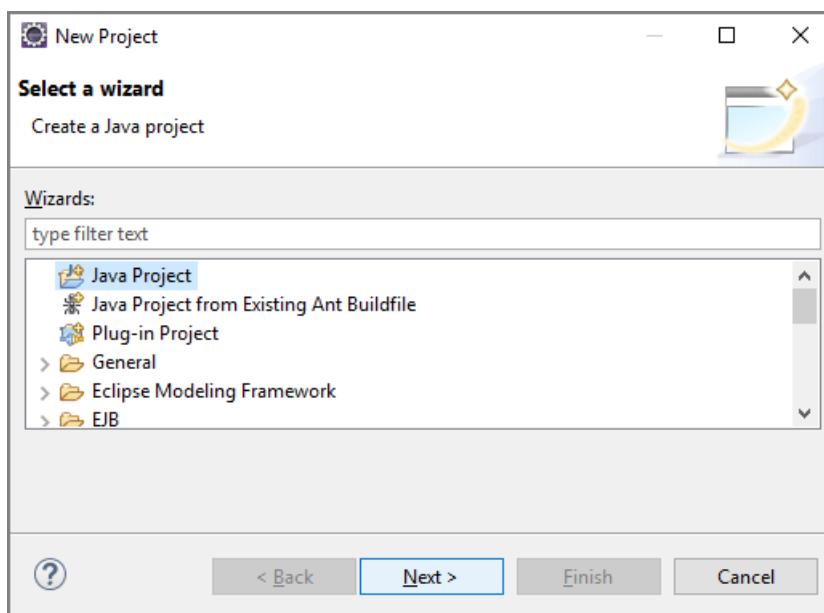
En este inicio rápido se describe, paso a paso, cómo hacer que una aplicación cliente simple se conecte al bot habilitado para voz. Si desea sumergirse de lleno, el código fuente completo y listo para compilar utilizado en este inicio rápido está disponible en los [ejemplos del SDK de Voz](#) bajo la carpeta `quickstart`.

## Creación y configuración de un proyecto

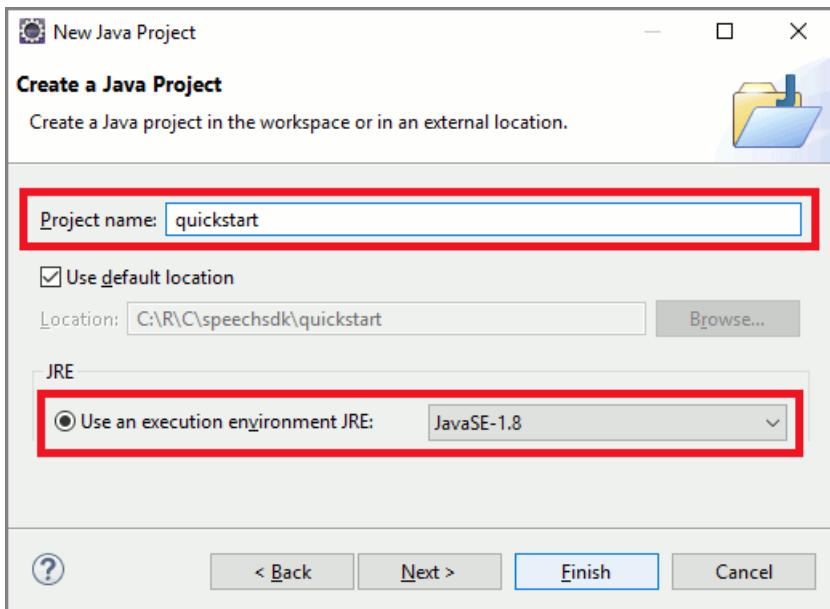
1. Inicie Eclipse.
2. En el selector de Eclipse, en el campo **Workspace** (Área de trabajo), escriba el nombre de un nuevo directorio de área de trabajo. Luego seleccione **Launch** (Iniciar).



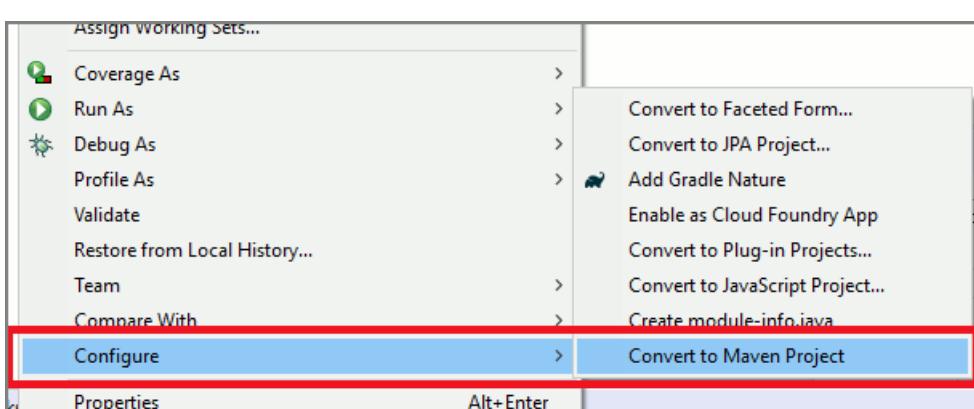
3. Al cabo de unos segundos, aparece la ventana principal del IDE de Eclipse. Cierre la pantalla de **bienvenida** si hay alguna.
4. En la barra de menús de Eclipse, cree un nuevo proyecto eligiendo **File (Archivo) > New (Nuevo) > Project (Proyecto)**.
5. Aparecerá el cuadro de diálogo **Nuevo proyecto**. Seleccione **Java Project** (Proyecto de Java) y seleccione **Next (Siguiente)**.



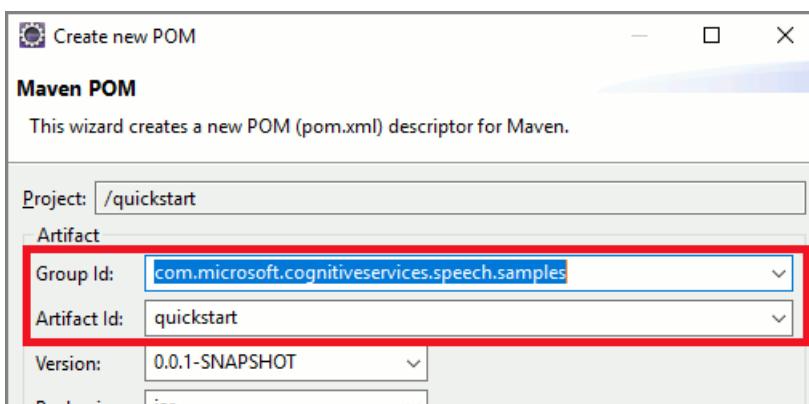
6. Se inicia el asistente para **nuevo proyecto de Java**. En el campo **Project name** (Nombre del proyecto), escriba **quickstart** y elija **JavaSE-1.8** como el entorno de ejecución. Seleccione **Finalizar**.



7. Si aparece una ventana titulada **Open Associated Perspective?** (¿Abrir perspectiva asociada?), seleccione **Open Perspective** (Abrir perspectiva).
8. En el **Explorador de paquetes**, haga clic en el proyecto **quickstart**. Elija **Configure (Configurar) > Convert to Maven Project (Convertir en proyecto de Maven)** en el menú contextual.



9. Aparece la ventana **Create new POM** (Crear nuevo POM). En el campo **Group Id** (Identificador de grupo), escriba `com.microsoft.cognitiveservices.speech.samples`, y en el campo **Artifact Id** (Identificador de artefacto), escriba `quickstart`. A continuación, seleccione **Finish** (Finalizar).



10. Abra el archivo `pom.xml` y edítelo.

- Al final del archivo, antes cerrar la etiqueta de cierre `</project>`, cree un elemento `repositories` con una referencia al repositorio de Maven para el SDK de Voz, tal y como se muestra aquí:

```
<repositories>
<repository>
<id>maven-cognitiveservices-speech</id>
<name>Microsoft Cognitive Services Speech Maven Repository</name>
<url>https://csspeechstorage.blob.core.windows.net/maven/</url>
</repository>
</repositories>
```

- Agregue también un elemento `dependencies`, con la versión 1.7.0 del SDK de Voz como dependencia:

```
<dependencies>
<dependency>
<groupId>com.microsoft.cognitiveservices.speech</groupId>
<artifactId>client-sdk</artifactId>
<version>1.7.0</version>
</dependency>
</dependencies>
```

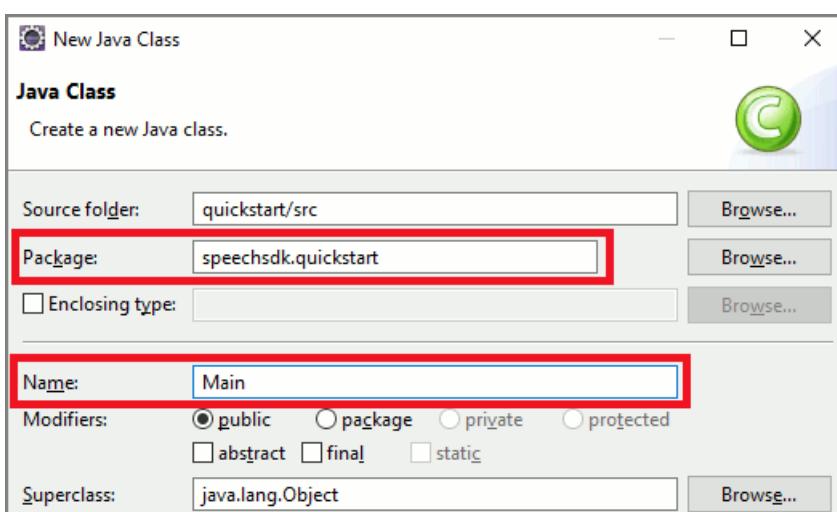
- Guarde los cambios.

Además, para habilitar el registro, actualice el archivo `pom.xml` para incluir la dependencia siguiente:

```
<dependency>
<groupId>org.slf4j</groupId>
<artifactId>slf4j-simple</artifactId>
<version>1.7.5</version>
</dependency>
```

## Incorporación de código de ejemplo

1. Para agregar una nueva clase vacía al proyecto de Java, seleccione **File (Archivo) > New (Nuevo) > Class (Clase)**.
2. En la ventana **New Java Class** (Nueva clase de Java) escriba `speechsdk.quickstart` en el campo **Package** (Paquete) y `Main` en el campo **Name** (Nombre).



3. Abra la clase recién creada `Main` y reemplace el contenido del archivo `Main.java` por el código de inicio siguiente:

```

package speechsdk.quickstart;

import com.microsoft.cognitiveservices.speech.audio.AudioConfig;
import com.microsoft.cognitiveservices.speech.audio.PullAudioOutputStream;
import com.microsoft.cognitiveservices.speech.dialog.DialogServiceConfig;
import com.microsoft.cognitiveservices.speech.dialog.DialogServiceConnector;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

import javax.sound.sampled.AudioFormat;
import javax.sound.sampled.AudioSystem;
import javax.sound.sampled.DataLine;
import javax.sound.sampled.SourceDataLine;
import java.io.InputStream;

public class Main {
    final Logger log = LoggerFactory.getLogger(Main.class);

    public static void main(String[] args) {
        // New code will go here
    }

    private void playAudioStream(PullAudioOutputStream audio) {
        ActivityAudioStream stream = new ActivityAudioStream(audio);
        final ActivityAudioStream.ActivityAudioFormat audioFormat = stream.getActivityAudioFormat();
        final AudioFormat format = new AudioFormat(
            AudioFormat.Encoding.PCM_SIGNED,
            audioFormat.getSamplesPerSecond(),
            audioFormat.getBitsPerSample(),
            audioFormat.getChannels(),
            audioFormat.getFrameSize(),
            audioFormat.getSamplesPerSecond(),
            false);
        try {
            int bufferSize = format.getFrameSize();
            final byte[] data = new byte[bufferSize];

            SourceDataLine.Info info = new DataLine.Info(SourceDataLine.class, format);
            SourceDataLine line = (SourceDataLine) AudioSystem.getLine(info);
            line.open(format);

            if (line != null) {
                line.start();
                int nBytesRead = 0;
                while (nBytesRead != -1) {
                    nBytesRead = stream.read(data);
                    if (nBytesRead != -1) {
                        line.write(data, 0, nBytesRead);
                    }
                }
                line.drain();
                line.stop();
                line.close();
            }
            stream.close();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}

```

4. En el método `main`, primero configurará el `DialogServiceConfig` y lo utilizará para crear una instancia de `DialogServiceConnector`. Esta instancia se conecta al canal de Direct Line Speech para interactuar con el bot. También se utiliza una instancia de `AudioConfig` para especificar el origen de la entrada de audio. En este

ejemplo, se usa el micrófono predeterminado con `AudioConfig.fromDefaultMicrophoneInput()`.

- Reemplace la cadena `YourSubscriptionKey` por la clave de suscripción, que puede obtener en [este sitio web](#).
- Reemplace la cadena `YourServiceRegion` por la [región](#) asociada a su suscripción.

**NOTE**

Consulte [la lista de regiones admitidas para los asistentes de voz](#) y asegúrese de que sus recursos se implementan en una de esas regiones.

```
final String subscriptionKey = "YourSubscriptionKey"; // Your subscription key
final String region = "YourServiceRegion"; // Your speech subscription service region
final BotFrameworkConfig botConfig = BotFrameworkConfig.fromSubscription(subscriptionKey, region);

// Configure audio input from a microphone.
final AudioConfig audioConfig = AudioConfig.fromDefaultMicrophoneInput();

// Create a DialogServiceConnector instance.
final DialogServiceConnector connector = new DialogServiceConnector(botConfig, audioConfig);
```

5. El conector `DialogServiceConnector` se basa en varios eventos para comunicar sus actividades de bot, los resultados del reconocimiento de voz y otra información. Agregue estos clientes de escucha de eventos a continuación.

```

// Recognizing will provide the intermediate recognized text while an audio stream is being processed.
connector.recognizing.addEventListener((o, speechRecognitionEventArgs) -> {
    log.info("Recognizing speech event text: {}", speechRecognitionEventArgs.getResult().getText());
});

// Recognized will provide the final recognized text once audio capture is completed.
connector.recognized.addEventListener((o, speechRecognitionEventArgs) -> {
    log.info("Recognized speech event reason text: {}", speechRecognitionEventArgs.getResult().getText());
});

// SessionStarted will notify when audio begins flowing to the service for a turn.
connector.sessionStarted.addEventListener((o, sessionEventArgs) -> {
    log.info("Session Started event id: {} ", sessionEventArgs.getSessionId());
});

// SessionStopped will notify when a turn is complete and it's safe to begin listening again.
connector.sessionStopped.addEventListener((o, sessionEventArgs) -> {
    log.info("Session stopped event id: {}", sessionEventArgs.getSessionId());
});

// Canceled will be signaled when a turn is aborted or experiences an error condition.
connector.canceled.addEventListener((o, canceledEventArgs) -> {
    log.info("Canceled event details: {}", canceledEventArgs.getErrorDetails());
    connector.disconnectAsync();
});

// ActivityReceived is the main way your bot will communicate with the client and uses Bot Framework
activities.
connector.activityReceived.addEventListener((o, activityEventArgs) -> {
    final String act = activityEventArgs.getActivity().serialize();
    log.info("Received activity {} audio", activityEventArgs.hasAudio() ? "with" : "without");
    if (activityEventArgs.hasAudio()) {
        playAudioStream(activityEventArgs.getAudio());
    }
});

```

6. Conecte el `DialogServiceConnector` a Direct Line Speech mediante la llamada al método `connectAsync()`. Para probar el bot, puede invocar el método `listenOnceAsync` para enviar una entrada de audio desde el micrófono. Además, también puede utilizar el método `sendActivityAsync` para enviar una actividad personalizada como una cadena serializada. Estas actividades personalizadas pueden proporcionar datos adicionales que el bot utilizará en la conversación.

```

connector.connectAsync();
// Start listening.
System.out.println("Say something ...");
connector.listenOnceAsync();

// connector.sendActivityAsync(...)

```

7. Guarde los cambios en el archivo `Main`.
8. Para admitir la reproducción de respuestas, puede agregar una clase adicional que se transforma el objeto `PullAudioOutputStream` devuelto desde la API de `getAudio()` a un elemento `InputStream` de Java para facilitar el control. Esta `ActivityAudioStream` es una clase especializada que controla la respuesta de audio desde el canal de Direct Line Speech. Proporcionará descriptores de acceso para capturar la información del formato de audio que se requiere para controlar la reproducción. Para ello, seleccione **Archivo > Nuevo > Clase**.
9. En la ventana **New Java Class** (Nueva clase Java), escriba `speechsdk.quickstart` en el campo **Package**

(Paquete) y *ActivityAudioStream* en el campo **Name** (Nombre).

10. Abra la clase `ActivityAudioStream` que se creó recientemente y reemplace el contenido por el código siguiente:

```
package com.speechsdk.quickstart;

import com.microsoft.cognitiveservices.speech.audio.PullAudioOutputStream;

import java.io.IOException;
import java.io.InputStream;

public final class ActivityAudioStream extends InputStream {
    /**
     * The number of samples played per second (16 kHz).
     */
    public static final long SAMPLE_RATE = 16000;
    /**
     * The number of bits in each sample of a sound that has this format (16 bits).
     */
    public static final int BITS_PER_SECOND = 16;
    /**
     * The number of audio channels in this format (1 for mono).
     */
    public static final int CHANNELS = 1;
    /**
     * The number of bytes in each frame of a sound that has this format (2).
     */
    public static final int FRAME_SIZE = 2;

    /**
     * Reads up to a specified maximum number of bytes of data from the audio
     * stream, putting them into the given byte array.
     *
     * @param b   the buffer into which the data is read
     * @param off the offset, from the beginning of array <code>b</code>, at which
     *           the data will be written
     * @param len the maximum number of bytes to read
     * @return the total number of bytes read into the buffer, or -1 if there
     * is no more data because the end of the stream has been reached
     */
    @Override
    public int read(byte[] b, int off, int len) {
        byte[] tempBuffer = new byte[len];
        int n = (int) this.pullStreamImpl.read(tempBuffer);
        for (int i = 0; i < n; i++) {
            if (off + i > b.length) {
                throw new ArrayIndexOutOfBoundsException(b.length);
            }
            b[off + i] = tempBuffer[i];
        }
        if (n == 0) {
            return -1;
        }
        return n;
    }

    /**
     * Reads the next byte of data from the activity audio stream if available.
     *
     * @return the next byte of data, or -1 if the end of the stream is reached
     * @see #read(byte[], int, int)
     * @see #read(byte[])
     * @see #available
     * <p>
     */
    @Override
```

```

public int read() {
    byte[] data = new byte[1];
    int temp = read(data);
    if (temp <= 0) {
        // we have a weird situation if read(byte[]) returns 0!
        return -1;
    }
    return data[0] & 0xFF;
}

/**
 * Reads up to a specified maximum number of bytes of data from the activity audio stream,
 * putting them into the given byte array.
 *
 * @param b the buffer into which the data is read
 * @return the total number of bytes read into the buffer, or -1 if there
 * is no more data because the end of the stream has been reached
 */
@Override
public int read(byte[] b) {
    int n = (int) pullStreamImpl.read(b);
    if (n == 0) {
        return -1;
    }
    return n;
}

/**
 * Skips over and discards a specified number of bytes from this
 * audio input stream.
 *
 * @param n the requested number of bytes to be skipped
 * @return the actual number of bytes skipped
 * @throws IOException if an input or output error occurs
 * @see #read
 * @see #available
 */
@Override
public long skip(long n) {
    if (n <= 0) {
        return 0;
    }
    if (n <= Integer.MAX_VALUE) {
        byte[] tempBuffer = new byte[(int) n];
        return read(tempBuffer);
    }
    long count = 0;
    for (long i = n; i > 0; i -= Integer.MAX_VALUE) {
        int size = (int) Math.min(Integer.MAX_VALUE, i);
        byte[] tempBuffer = new byte[size];
        count += read(tempBuffer);
    }
    return count;
}

/**
 * Closes this audio input stream and releases any system resources associated
 * with the stream.
 */
@Override
public void close() {
    this.pullStreamImpl.close();
}

/**
 * Fetch the audio format for the ActivityAudioStream. The ActivityAudioFormat defines the sample
 * rate, bits per sample, and the # channels.
 *
 * @return instance of the ActivityAudioFormat associated with the stream

```

```

/*
public ActivityAudioStream.ActivityAudioFormat getActivityAudioFormat() {
    return activityAudioFormat;
}

/**
 * Returns the maximum number of bytes that can be read (or skipped over) from this
 * audio input stream without blocking.
 *
 * @return the number of bytes that can be read from this audio input stream without blocking.
 * As this implementation does not buffer, this will be defaulted to 0
 */
@Override
public int available() {
    return 0;
}

public ActivityAudioStream(final PullAudioOutputStream stream) {
    pullStreamImpl = stream;
    this.activityAudioFormat = new ActivityAudioStream.ActivityAudioFormat(SAMPLE_RATE,
BITS_PER_SECOND, CHANNELS, FRAME_SIZE, AudioEncoding.PCM_SIGNED);
}

private PullAudioOutputStream pullStreamImpl;

private ActivityAudioFormat activityAudioFormat;

/**
 * ActivityAudioFormat is an internal format which contains metadata regarding the type of
arrangement of
 * audio bits in this activity audio stream.
 */
static class ActivityAudioFormat {

    private long samplesPerSecond;
    private int bitsPerSample;
    private int channels;
    private int frameSize;
    private AudioEncoding encoding;

    public ActivityAudioFormat(long samplesPerSecond, int bitsPerSample, int channels, int
frameSize, AudioEncoding encoding) {
        this.samplesPerSecond = samplesPerSecond;
        this.bitsPerSample = bitsPerSample;
        this.channels = channels;
        this.encoding = encoding;
        this.frameSize = frameSize;
    }

    /**
     * Fetch the number of samples played per second for the associated audio stream format.
     *
     * @return the number of samples played per second
     */
    public long getSamplesPerSecond() {
        return samplesPerSecond;
    }

    /**
     * Fetch the number of bits in each sample of a sound that has this audio stream format.
     *
     * @return the number of bits per sample
     */
    public int getBitsPerSample() {
        return bitsPerSample;
    }

    /**
     * Fetch the number of audio channels used by this audio stream format.
     */
}

```

```

        *
        * @return the number of channels
        */
    public int getChannels() {
        return channels;
    }

    /**
     * Fetch the default number of bytes in a frame required by this audio stream format.
     *
     * @return the number of bytes
     */
    public int getFrameSize() {
        return frameSize;
    }

    /**
     * Fetch the audio encoding type associated with this audio stream format.
     *
     * @return the encoding associated
     */
    public AudioEncoding getEncoding() {
        return encoding;
    }

}

/**
 * Enum defining the types of audio encoding supported by this stream.
 */
public enum AudioEncoding {
    PCM_SIGNED("PCM_SIGNED");

    String value;

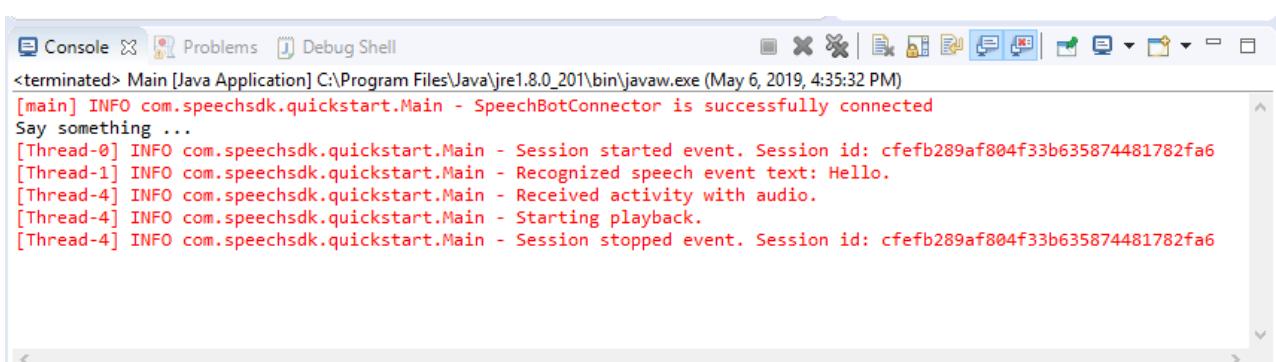
    AudioEncoding(String value) {
        this.value = value;
    }
}
}

```

11. Guarde los cambios en el archivo `ActivityAudioStream`.

## Compilación y ejecución de la aplicación

Seleccione F11, o seleccione **Run (Ejecutar) > Debug (Depurar)**. La consola muestra el mensaje: "Say something" (Diga algo). En ese momento, diga una frase en inglés que el bot pueda entender. La voz se transmite al bot a través del canal de Direct Line Speech y, una vez allí, el bot la reconoce y la procesa. La respuesta se devuelve como una actividad. Si el bot devuelve la voz como respuesta, el audio se reproducirá mediante la clase `AudioPlayer`.



```

Console X Problems Debug Shell
<terminated> Main [Java Application] C:\Program Files\Java\jre1.8.0_201\bin\javaw.exe (May 6, 2019, 4:35:32 PM)
[main] INFO com.speechsdk.quickstart.Main - SpeechBotConnector is successfully connected
Say something ...
[Thread-0] INFO com.speechsdk.quickstart.Main - Session started event. Session id: cfefb289af804f33b635874481782fa6
[Thread-1] INFO com.speechsdk.quickstart.Main - Recognized speech event text: Hello.
[Thread-4] INFO com.speechsdk.quickstart.Main - Received activity with audio.
[Thread-4] INFO com.speechsdk.quickstart.Main - Starting playback.
[Thread-4] INFO com.speechsdk.quickstart.Main - Session stopped event. Session id: cfefb289af804f33b635874481782fa6

```

## Pasos siguientes

Se pueden encontrar ejemplos adicionales, por ejemplo, cómo leer voz de un archivo de audio, en GitHub.

[Creación e implementación de un bot básico](#)

## Otras referencias

- [Acerca de los asistentes de voz](#)
- [Obtenga una clave de suscripción gratuita a los servicios de Voz](#)
- [Palabras clave personalizadas](#)
- [Conexión de Direct Line de Voz al bot](#)
- [Exploración de ejemplos de Java en GitHub](#)

# Inicio rápido: Creación de un asistente de voz de Java en Android con el SDK de Voz

13/01/2020 • 12 minutes to read • [Edit Online](#)

También hay un inicio rápido disponible para la [conversión de voz en texto y de texto en voz](#).

En este artículo, va a crear un asistente de voz con Java para Android mediante el [SDK de Voz](#). Esta aplicación se conectará a un bot que ya se haya creado y configurado con el [canal Direct Line Speech](#). A continuación, enviará una solicitud de voz al bot y presentará una actividad de respuesta habilitada para voz.

Esta aplicación se basa en el paquete de Maven de Speech SDK y en Android Studio 3.3. El SDK de Voz es compatible actualmente con dispositivos Android que tienen procesadores ARM de 32 o 64 bits o Intel x86 o x64.

## NOTE

Para el SDK de dispositivos de Voz y el dispositivo Roobo, consulte [SDK de dispositivos de voz](#).

## Requisitos previos

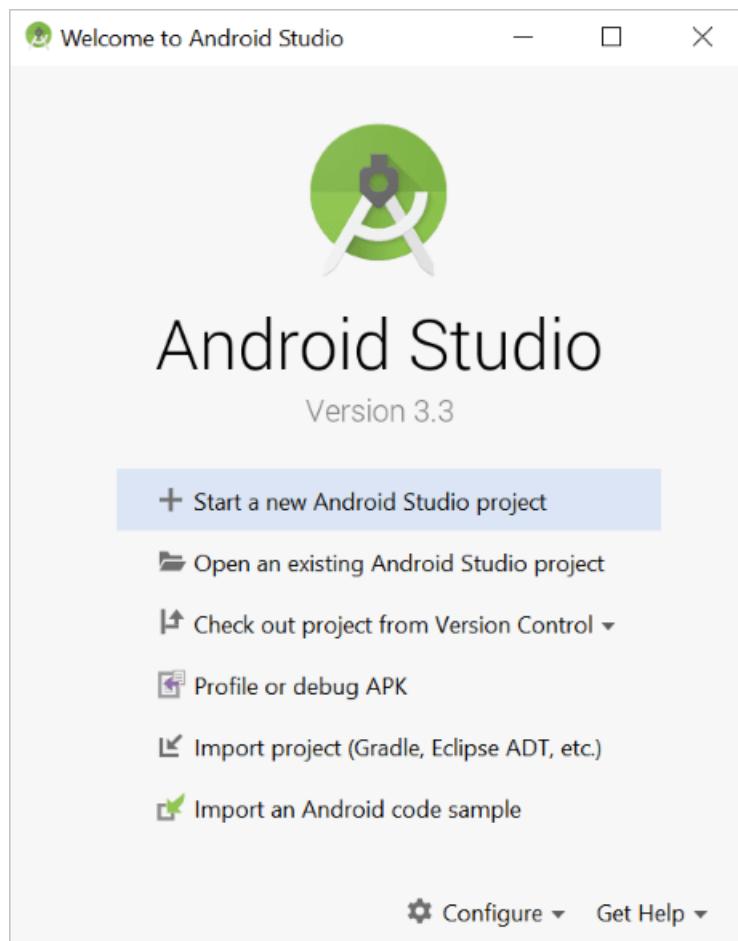
- Una clave de suscripción de Azure para el servicio Voz. [Obtenga una gratis](#) o créela en [Azure Portal](#).
- Un bot creado previamente y configurado con el [canal Direct Line Speech](#)
- [Android Studio v3.3](#) o versiones posteriores

## NOTE

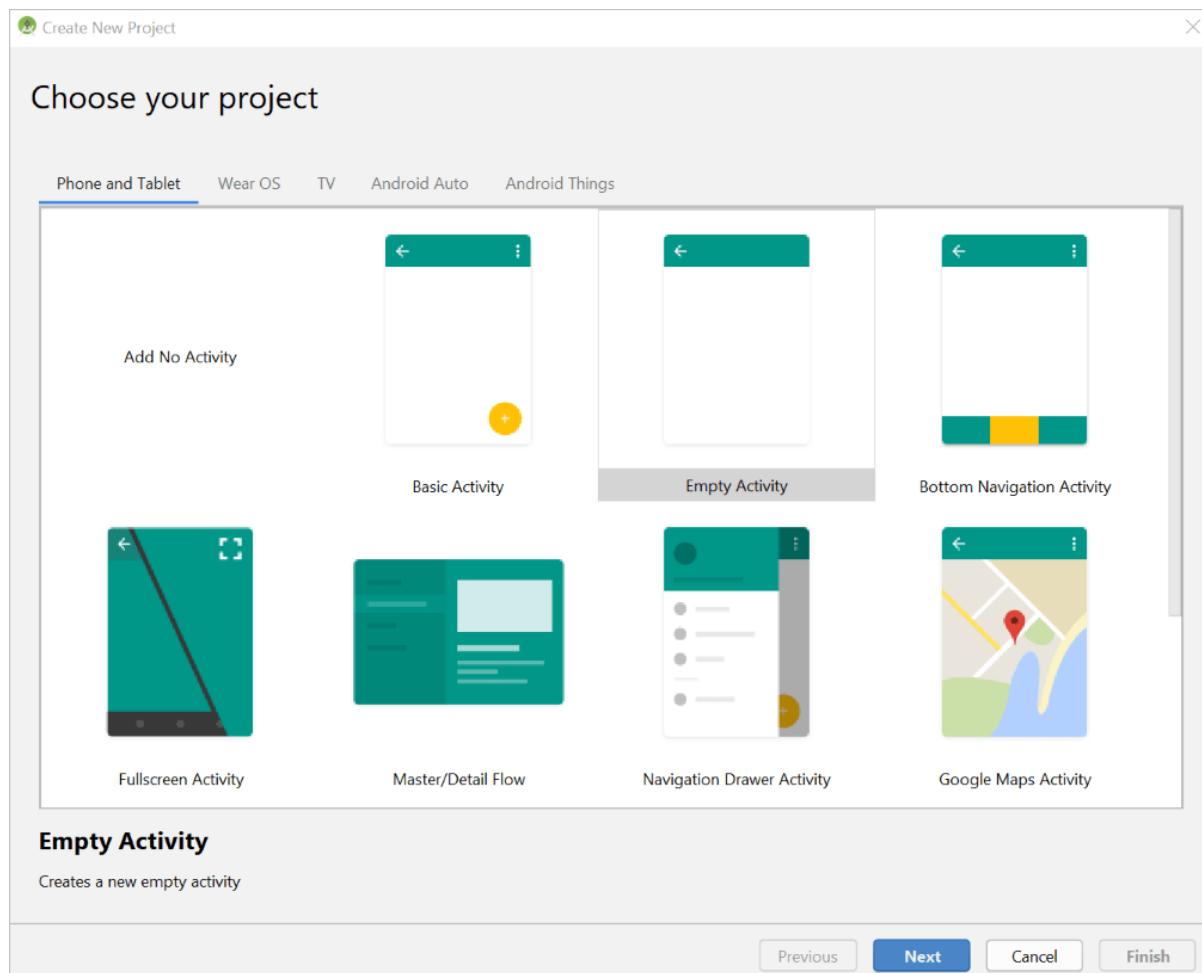
Consulte [la lista de regiones admitidas para los asistentes de voz](#) y asegúrese de que sus recursos se implementan en una de esas regiones.

## Creación y configuración de un proyecto

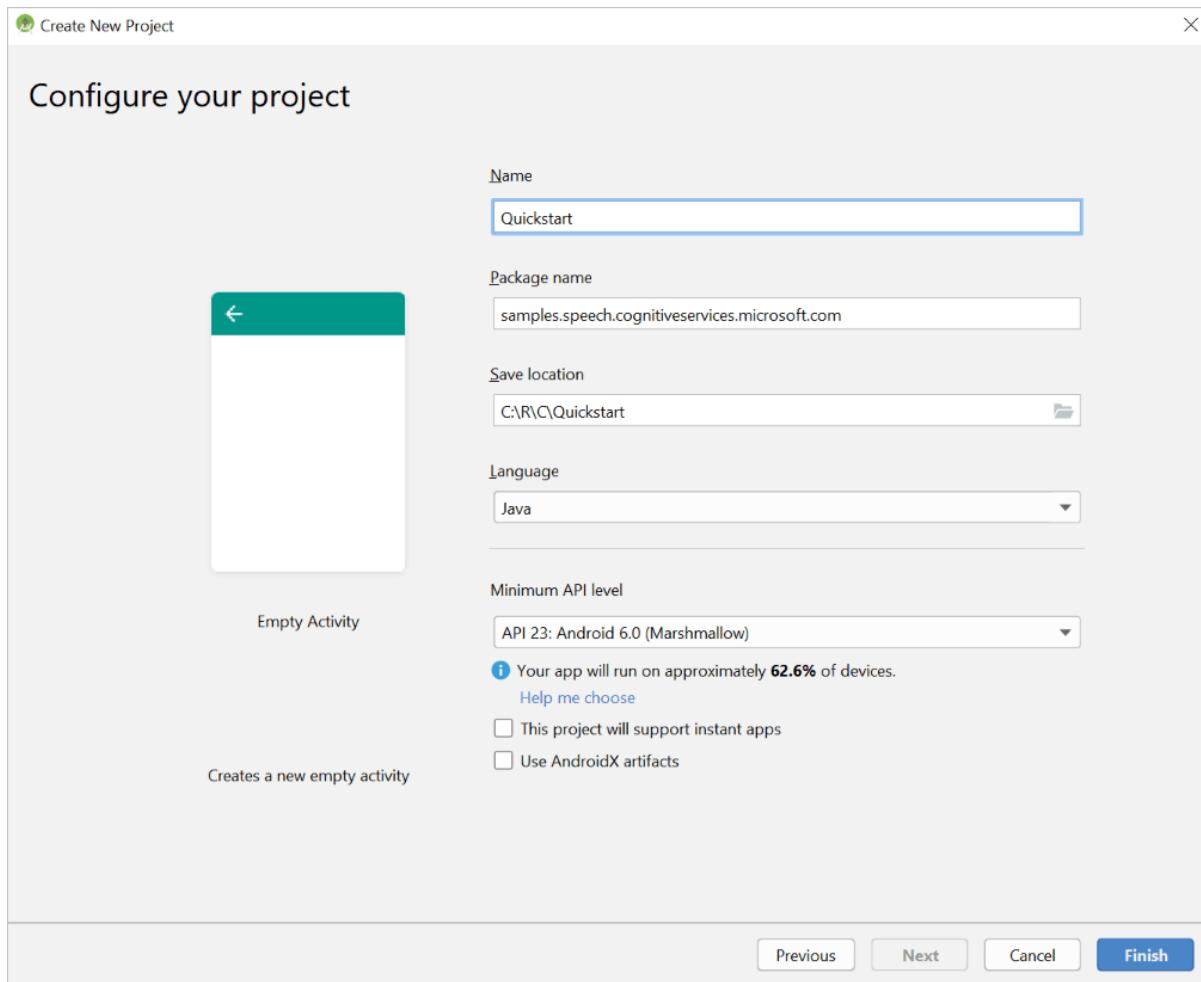
1. Inicie Android Studio y seleccione **Start a new Android Studio project** (Iniciar un nuevo proyecto de Android Studio) en la ventana **Welcome** (Bienvenida).



2. Aparece el asistente para la **elección del proyecto**. Seleccione **Phone and Tablet** (Teléfono y tableta) y **Empty Activity** (Actividad vacía) en el cuadro de selección de actividad. Seleccione **Next** (Siguiente).



3. En la pantalla **Configure your project** (Configure su proyecto), escriba *Inicio rápido* como **nombre** y escriba *samples.speech.cognitiveservices.microsoft.com* como **nombre del paquete**. Despues, seleccione un directorio de proyecto. En **Minimum API level** (Nivel de API mínimo), seleccione **API 23: Android Marshmallow 6.0 o posterior**. Deje todas las demás casillas desactivadas y seleccione **Finalizar**.



Android Studio tardará unos minutos en preparar el nuevo proyecto de Android. A continuación, configure el proyecto para obtener información del SDK de Voz de Azure Cognitive Services y para usar Java 8.

#### IMPORTANT

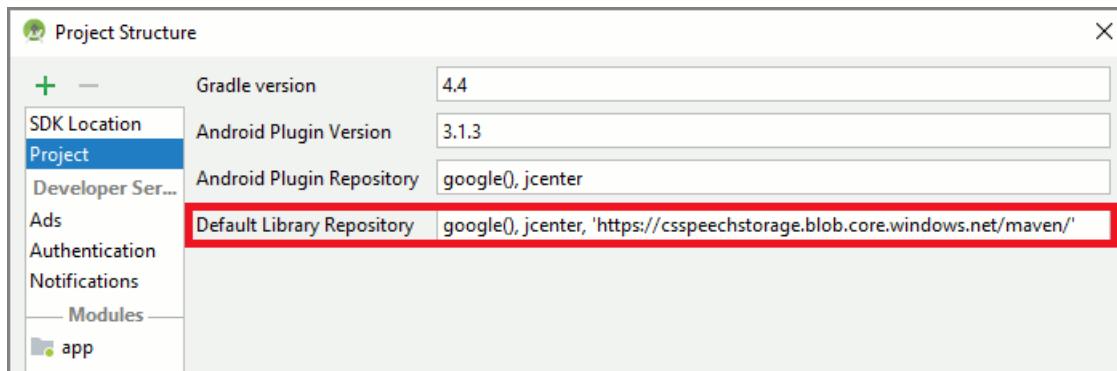
Al descargar cualquiera de los componentes del SDK de Voz de Azure Cognitive Services de esta página, acepta su licencia. Consulte los [términos de licencia del software de Microsoft para el SDK de Voz](#).

La versión actual del SDK de Voz de Cognitive Services es 1.7.0.

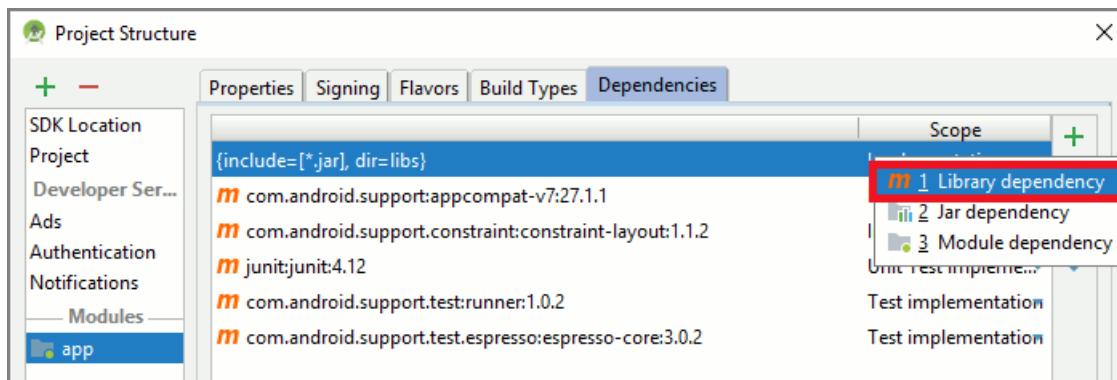
El SDK de Voz para Android está empaquetado como una **biblioteca de Android (AAR)**, que incluye las bibliotecas necesarias, así como los permisos necesarios de Android. Se hospeda en un repositorio de Maven en <https://csspeechstorage.blob.core.windows.net/maven/>.

Configure el proyecto para usar el SDK de Voz. Abra la ventana de la **estructura del proyecto** seleccionando **File** (Archivo) > **Project Structure** (Estructura del proyecto) desde la barra de menús de Android Studio. En la ventana **Project Structure** (Estructura del proyecto), realice los siguientes cambios:

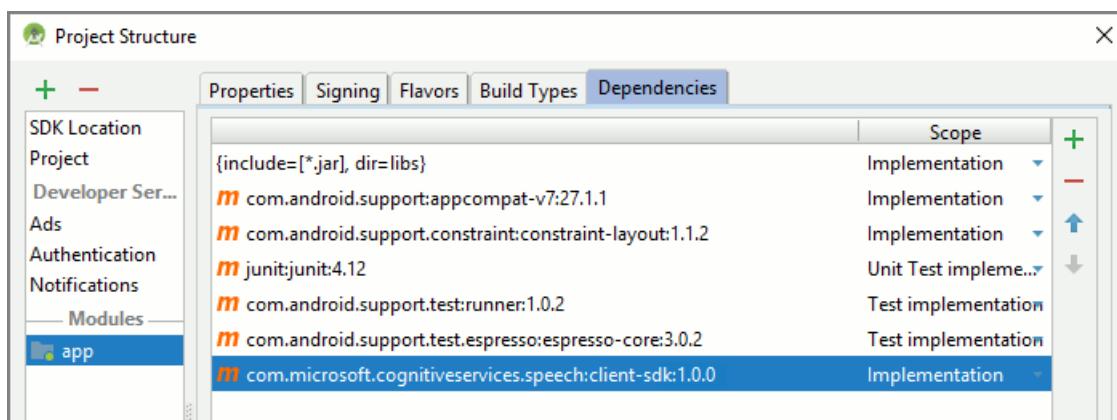
1. En la lista, en el lado izquierdo de la ventana, seleccione **Project** (Proyecto). Edite la opción **Default Library Repository** (Repositorio de bibliotecas predeterminado). Para ello, debe anexar una coma y poner entre comillas simples la dirección URL del repositorio de Maven:  
`'https://csspeechstorage.blob.core.windows.net/maven/'`.



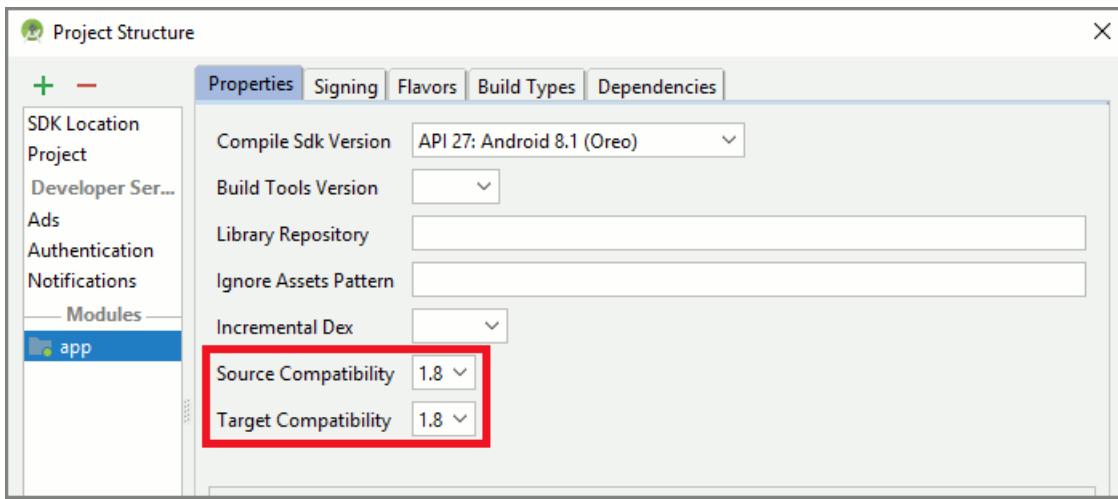
2. En la misma pantalla, en el lado izquierdo, seleccione **App** (Aplicación). A continuación, seleccione la pestaña **Dependencies** (Dependencias) en la parte superior de la ventana. Seleccione el signo más verde (+) y seleccione **Library dependency** (Dependencia de biblioteca) en el menú desplegable.



3. En la ventana que aparece, escriba el nombre y la versión del SDK de Voz para Android, *com.microsoft.cognitiveservices.speech:client-sdk:1.7.0*. Después seleccione **Aceptar**. El SDK de Voz se agregará a la lista de dependencias, como se muestra a continuación:



4. Haga clic en la pestaña **Properties** (Propiedades). En **Source Compatibility** (Compatibilidad de origen) y **Target Compatibility** (Compatibilidad de destino), seleccione **1.8**.



5. Seleccione **OK** (Aceptar) para cerrar la ventana de la **estructura del proyecto** y aplicar los cambios al proyecto.

## Interfaz de Create user (Crear usuario)

En esta sección, se va a crear una interfaz de usuario básica para la aplicación. Comencemos por abrir la actividad principal: `activity_main.xml`. La plantilla básica incluye una barra de título con el nombre de la aplicación y un `TextView` con el mensaje "Hello world!".

A continuación, reemplace el contenido de `activity_main.xml` por el código siguiente:

```

<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context=".MainActivity">

    <Button
        android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:onClick="onBotButtonClicked"
        android:text="Talk to your bot" />

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Recognition Data"
        android:textSize="18dp"
        android:textStyle="bold" />

    <TextView
        android:id="@+id/recoText"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="\n(Recognition goes here)\n" />

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Activity Data"
        android:textSize="18dp"
        android:textStyle="bold" />

    <TextView
        android:id="@+id/activityText"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:scrollbars="vertical"
        android:text="\n(Activities go here)\n" />

</LinearLayout>

```

Este XML define una interfaz de usuario sencilla para interactuar con el bot.

- El elemento `button` inicia una interacción e invoca al método `onBotButtonClicked` cuando se hace clic.
- El elemento `recoText` mostrará los resultados de la conversión de voz a texto cuando hable al bot.
- El elemento `activityText` mostrará la carga de JSON para la actividad de Bot Framework más reciente desde el bot.

El texto y la representación gráfica de la interfaz de usuario ahora deben tener este aspecto:

TALK TO YOUR BOT

## Recognition Data

(Recognition goes here)

## Activity Data

(Activities go here)

## Incorporación de código de ejemplo

1. Abra `MainActivity.java` y reemplace el contenido por el código siguiente:

```
package samples.speech.cognitiveservices.microsoft.com;

import android.media.AudioFormat;
import android.media.AudioManager;
import android.media.AudioTrack;
import android.support.v4.app.ActivityCompat;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.text.method.ScrollingMovementMethod;
import android.view.View;
import android.widget.TextView;

import com.microsoft.cognitiveservices.speech.audio.AudioConfig;
import com.microsoft.cognitiveservices.speech.audio.PullAudioOutputStream;
import com.microsoft.cognitiveservices.speech.dialog.DialogServiceConfig;
import com.microsoft.cognitiveservices.speech.dialog.DialogServiceConnector;

import org.json.JSONException;
import org.json.JSONObject;

import static android.Manifest.permission.*;

public class MainActivity extends AppCompatActivity {
    // Replace below with your own speech subscription key
    private static String speechSubscriptionKey = "YourSpeechSubscriptionKey";
    // Replace below with your own speech service region
    private static String serviceRegion = "YourSpeechServiceRegion";

    private DialogServiceConnector connector;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        TextView recoText = (TextView) this.findViewById(R.id.recoText);
        TextView activityText = (TextView) this.findViewById(R.id.activityText);
        recoText.setMovementMethod(new ScrollingMovementMethod());
        activityText.setMovementMethod(new ScrollingMovementMethod());

        // Note: we need to request permissions for audio input and network access
        int requestCode = 5; // unique code for the permission request
    }
}
```

```

        ActivityCompat.requestPermissions(MainActivity.this, new String[]{RECORD_AUDIO, INTERNET},
requestCode);
    }

    public void onBotButtonClicked(View v) {
        // Recreate the DialogServiceConnector on each button press, ensuring that the existing one is
closed
        if (connector != null) {
            connector.close();
            connector = null;
        }

        // Create the DialogServiceConnector from speech subscription information
        BotFrameworkConfig config = BotFrameworkConfig.fromSubscription(speechSubscriptionKey,
serviceRegion);
        connector = new DialogServiceConnector(config, AudioConfig.fromDefaultMicrophoneInput());

        // Optional step: preemptively connect to reduce first interaction latency
        connector.connectAsync();

        // Register the DialogServiceConnector's event listeners
        registerEventListeners();

        // Begin sending audio to your bot
        connector.listenOnceAsync();
    }

    private void registerEventListeners() {
        TextView recoText = (TextView) this.findViewById(R.id.recoText); // 'recoText' is the ID of
your text view
        TextView activityText = (TextView) this.findViewById(R.id.activityText); // 'activityText' is
the ID of your text view

        // Recognizing will provide the intermediate recognized text while an audio stream is being
processed
        connector.recognizing.addEventListener((o, recoArgs) -> {
            recoText.setText(" Recognizing: " + recoArgs.getResult().getText());
        });

        // Recognized will provide the final recognized text once audio capture is completed
        connector.recognized.addEventListener((o, recoArgs) -> {
            recoText.setText(" Recognized: " + recoArgs.getResult().getText());
        });

        // SessionStarted will notify when audio begins flowing to the service for a turn
        connector.sessionStarted.addEventListener((o, sessionArgs) -> {
            recoText.setText("Listening...");
        });

        // SessionStopped will notify when a turn is complete and it's safe to begin listening again
        connector.sessionStopped.addEventListener((o, sessionArgs) -> {
        });

        // Canceled will be signaled when a turn is aborted or experiences an error condition
        connector.canceled.addEventListener((o, canceledArgs) -> {
            recoText.setText("Canceled (" + canceledArgs.getReason().toString() + ") error details: {" +
+ canceledArgs.getErrorDetails());
            connector.disconnectAsync();
        });

        // ActivityReceived is the main way your bot will communicate with the client and uses bot
framework activities.
        connector.activityReceived.addEventListener((o, activityArgs) -> {
            try {
                // Here we use JSONObject only to "pretty print" the condensed Activity JSON
                String rawActivity = activityArgs.getActivity().serialize();
                String formattedActivity = new JSONObject(rawActivity).toString(2);
                activityText.setText(formattedActivity);
            } catch (JSONException e) {

```

```
        activityText.setText("Couldn't format activity text: " + e.getMessage());
    }

    if (activityArgs.hasAudio()) {
        // Text-to-speech audio associated with the activity is 16 kHz 16-bit mono PCM data
        final int sampleRate = 16000;
        int bufferSize = AudioTrack.getMinBufferSize(sampleRate, AudioFormat.CHANNEL_OUT_MONO,
AudioFormat.ENCODING_PCM_16BIT);

        AudioTrack track = new AudioTrack(
            AudioManager.STREAM_MUSIC,
            sampleRate,
            AudioFormat.CHANNEL_OUT_MONO,
            AudioFormat.ENCODING_PCM_16BIT,
            bufferSize,
            AudioTrack.MODE_STREAM);

        track.play();

        PullAudioOutputStream stream = activityArgs.getAudio();

        // Audio is streamed as it becomes available. Play it as it arrives.
        byte[] buffer = new byte[bufferSize];
        long bytesRead = 0;

        do {
            bytesRead = stream.read(buffer);
            track.write(buffer, 0, (int) bytesRead);
        } while (bytesRead == bufferSize);

        track.release();
    }
});
```

- El método `onCreate` incluye código que solicita permisos para micrófono e Internet.
  - El método `onBotButtonClicked` es, como se indicó anteriormente, el controlador de clic de botón. Al presionar un botón, se desencadena una única interacción ("turno") con el bot.
  - El método `registerEventListeners` muestra los eventos utilizados por `DialogServiceConnector`, así como el control básico de las actividades entrantes.

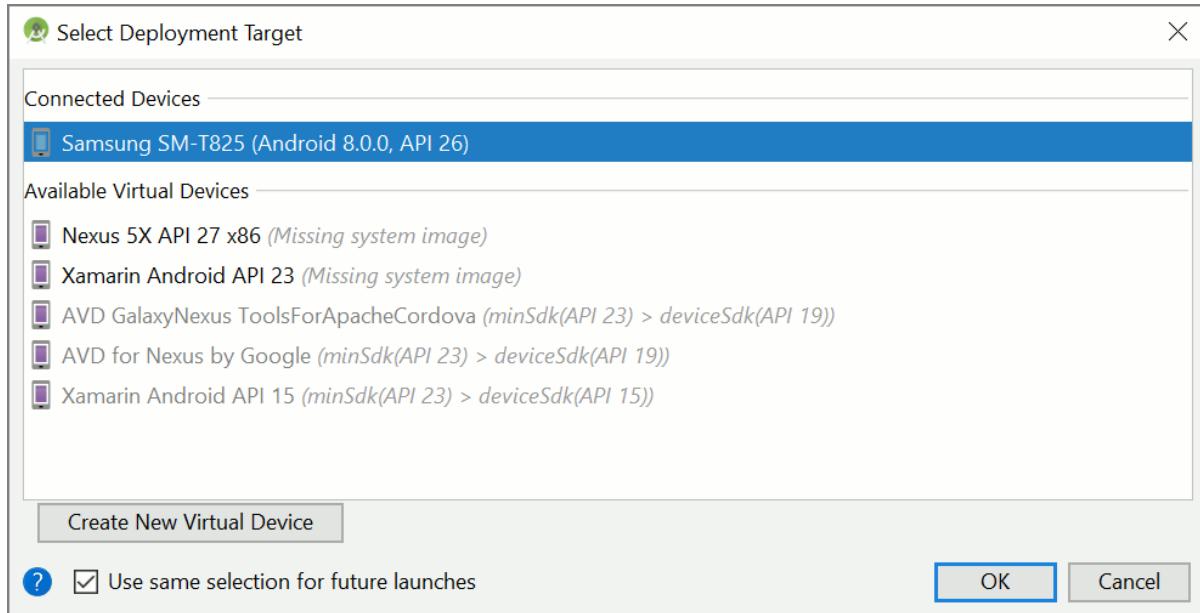
2. En el mismo archivo, reemplace las cadenas de configuración para que coincidan con los recursos:

- Reemplace `YourChannelSecret` por el secreto de canal Direct Line Speech para el bot.
  - Reemplace `YourSpeechSubscriptionKey` por la clave de suscripción.
  - Reemplace `YourServiceRegion` por la [región](#) asociada con la suscripción. Actualmente solo se admite un subconjunto de regiones del servicio de voz con Direct Line Speech. Para más información, consulte [Regiones](#).

# Compilación y ejecución de la aplicación

1. Conecte el dispositivo Android al equipo de desarrollo. Asegúrese de tener **habilitado el modo de desarrollo y la depuración USB** en el dispositivo.
  2. Para compilar la aplicación, presione Ctrl+F9 o elija **Build (Compilar) > Make Project (Crear proyecto)** desde la barra de menús.
  3. Para iniciar la aplicación, presione Mayús+F10 o elija **Run (Ejecutar) > Run 'app' (Ejecutar "aplicación")**.

4. En la ventanas de destino de implementación que aparece, elija el dispositivo Android.



Cuando se hayan iniciado la aplicación y su actividad, haga clic en el botón para comenzar a hablar con el bot. El texto transscrito aparecerá mientras habla y la última actividad que haya recibido del bot aparecerá cuando se reciba. Si el bot está configurado para proporcionar respuestas de voz, la conversión de voz a texto se reproducirá automáticamente.

## TALK TO YOUR BOT

### Recognition Data

Recognized: Hello there bot.

### Activity Data

```
{  
  "channelData": {  
    "conversationalAiData": {  
      "requestInfo": {  
        "interactionId": "d950b663-43e4-4ad7-a408-  
4dc5d218b76f",  
        "version": "0.2"  
      }  
    }  
  },  
  "channelId": "directlinespeech",  
  "conversation": {  
    "id": "dbae3b84-0717-4c2b-a29f-2d62b138209a",  
    "isGroup": false  
  },  
  "from": {  
    "id": "abigaildemo"  
  },  
  "id": "2814938a-fead-4aba-8fd7-4e025ebcfe7e",  
  "inputHint": "acceptingInput",  
  "recipient": {  
    "id": "dbae3b84-0717-4c2b-a29f-2d62b138209a|0000"  
  },  
  "replyTold": "f95e8857-f7cc-4370-9fba-784c0d5b721e",  
  "serviceUrl":  
  "urn:botframework:websocket:directlinespeech",  
  "speak": "<speak version=\"1.0\" xml:lang=\"en-us\"  
xmlns=\"https://www.w3.org/2001/10/  
synthesis\"><voice name=\"en-US-JessaNeural\"  
xmlns=\"\">Hi!</voice></speak>",  
  "text": "Hi!",  
  ...  
}
```



## Pasos siguientes

[Creación e implementación de un bot básico](#)

## Consulte también

- [Acerca de los asistentes de voz](#)
- [Obtenga una clave de suscripción gratuita a los servicios de Voz](#)
- [Palabras clave personalizadas](#)
- [Conexión de Direct Line de Voz al bot](#)
- [Exploración de ejemplos de Java en GitHub](#)

# Inicio rápido: Creación de un comando personalizado (versión preliminar)

13/01/2020 • 5 minutes to read • [Edit Online](#)

En este artículo, obtendrá información sobre cómo crear y probar una aplicación de comandos personalizados hospedada. La aplicación reconocerá una expresión como "enciende el televisor" y responderá con un mensaje sencillo como "de acuerdo, encenderé el televisor".

## Requisitos previos

- Una suscripción a Voz. [Prueba gratuita del servicio Voz](#).

### NOTE

Durante la versión preliminar, solo se admite la región westus2 para las claves de suscripción.

## Navegación a Speech Studio para los comandos personalizados

1. Abra el explorador web y navegue a [Speech Studio](#).
2. Introduzca sus credenciales para iniciar sesión en el portal.
  - La vista predeterminada es la lista de suscripciones de Voz.

### NOTE

Si no ve la página para seleccionar una suscripción, puede navegar a ella al seleccionar "Recursos de voz" en el menú de configuración de la barra superior.

3. Seleccione su suscripción a voz y, a continuación, seleccione **Ir a Studio**.
4. Seleccione **Comandos personalizados (versión preliminar)**.

La vista predeterminada es una lista de las aplicaciones de comandos personalizados que ha creado.

## Creación de un proyecto de comandos personalizados

1. Seleccione **Nuevo proyecto** para crear un nuevo proyecto.

## New project

X

Name \*

Name your project

Description

Describe your project

Language \*

Select a language

Authoring Resource \*

Select a resource

[Create new resource](#)

[Cancel](#)

[Create](#)

2. Escriba el nombre del proyecto y el lenguaje.

3. Seleccione un recurso de creación. Si no hay ninguno válido, créelo. Para ello, debe seleccionar **Crear nuevo recurso**.

## New LUIS Authoring Resource

X

Resource Name \*

Resource Group \*

 Search for or create a new resource group

Location \*

 Select a Location

Pricing Tier \*

 Select a pricing tier

[Cancel](#)

[Create](#)

a. Escriba el nombre del recurso, el grupo, la ubicación y el plan de tarifa.

#### NOTE

Para crear grupos de recursos, escriba el nombre del grupo de recursos deseado en el campo "Grupo de recursos". El grupo de recursos se creará cuando se seleccione **Crear**.

4. Haga clic en **Crear** para crear el proyecto.

5. Una vez que cree el proyecto, selecciónelo.

La vista ahora debe mostrar una visión general de la aplicación de comandos personalizados.

## Creación de un nuevo comando

Ahora puede crear un comando. Vamos a usar un ejemplo que tomará una sola expresión (`turn on the tv`) y responderá con el mensaje `ok, turning on the TV`.

1. Cree un nuevo comando. Para ello, seleccione el icono de **+** junto a comandos y asígnele el nombre `TurnOn`.
2. Seleccione **Guardar**.

Cognitive Services | Speech Studio

Speech Studio > Custom Commands

## Device Control Quickstart

English (United States)

Commands	+	Sample sentences
FallbackCommand	+	Parameters
<b>TurnOn</b>	+	Http Endpoints
Settings	+	Completion Rules
	+	Advanced Rules

Un comando es un conjunto de:

GRUPO	DESCRIPCIÓN
Oraciones de ejemplo	Expresiones de ejemplo que el usuario puede decir para desencadenar este comando.
Parámetros	Información necesaria para completar el comando.
Reglas de finalización	Acciones que se van a realizar para finalizar el comando. Por ejemplo, para responder al usuario o comunicarse con otro servicio web.
Reglas avanzadas	Reglas adicionales para controlar situaciones más específicas o complejas.

### Adición de una oración de ejemplo

Se comienza con las oraciones de ejemplo y se proporciona un ejemplo de lo que el usuario puede decir:

turn on the tv

Por ahora, no hay ningún parámetro, así que es posible continuar con las reglas de finalización.

### Adición de una regla de finalización

Ahora, agregue una regla de finalización para responder al usuario que una acción se está llevando a cabo.

1. Cree una regla de finalización, para lo que debe seleccionar el icono que hay junto a las reglas de finalización.
2. Escriba el nombre de la regla.
3. Agregar una acción
  - a. Cree una nueva acción de respuesta de voz, para lo que debe seleccionar el icono que se encuentra junto a las acciones, y, después, seleccionar
  - b. Escriba la respuesta.

#### NOTE

El texto normal debe empezar por un guión. Para más información, consulte [aquí](#)

## Edit Action

X

Type \*

SpeechResponse

▼

### Response template



1 - Ok, turning on the TV

Save

Cancel

4. Haga clic en **Guardar** para guardar la regla

## Completion Rules

Completion rules are executed once we're ready to fulfill, or complete, the command. They occur when all the required parameters are gathered.

### Rule Name ⓘ

Name \*

ConfirmationResponse

### Conditions +

### Actions +

SpeechResponse

- Ok, turning on the TV

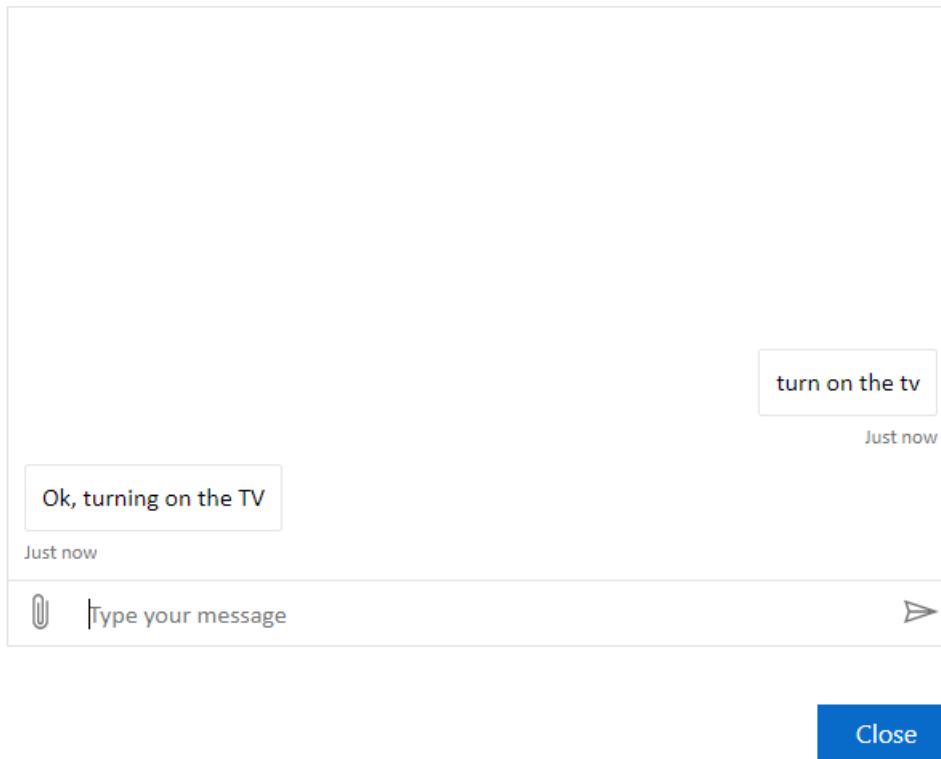
Save

CONFIGURACIÓN	VALOR SUGERIDO	DESCRIPCIÓN
Nombre de la regla	"ConfirmationResponse"	Nombre que describe el propósito de la regla.
Condiciones	None	Condiciones que determinan cuándo se puede ejecutar la regla.
Acciones	SpeechResponse "- Ok, turning on the TV" (De acuerdo, encenderé el televisor)	Acción que se realizará cuando la condición de la regla sea true.

## Prueba

Pruebe el comportamiento mediante el panel de conversación de prueba.

Test your application



- Si escribe: "turn on the tv" (enciende el televisor)
- La respuesta esperada: "Ok, turning on the TV" (de acuerdo, encenderé el televisor)

## Pasos siguientes

[Inicio rápido: Creación de un comando personalizado con parámetros \(versión preliminar\)](#)

# Inicio rápido: Creación de un comando personalizado con parámetros (versión preliminar)

15/01/2020 • 6 minutes to read • [Edit Online](#)

En el [artículo anterior](#), creamos un nuevo proyecto de comandos personalizados para responder a comandos sin parámetros.

En este artículo, extenderemos esta aplicación con parámetros para que pueda controlar el encendido y apagado de varios dispositivos.

## Creación de parámetros

1. Abra el proyecto [creado anteriormente](#).
2. Dado que el comando controlará ahora el encendido y apagado, cambie el nombre del comando a "TurnOnOff".
  - Mantenga el mouse sobre el nombre del comando y seleccione el ícono de edición para cambiar el nombre.
3. Cree un nuevo parámetro para indicar si el usuario quiere encender o apagar el dispositivo.
  - Seleccione el ícono de  junto a la sección de parámetros.

# Parameters

A parameter is a variable that's filled during dialogue with a Custom Commands application. [Learn more](#)

## Parameter details

Name \*

OnOff

Is Global

Required

Response template \*

1 - On or off?

Type \*

String

Default Value

Configuration \*

String List

## String list values



on

off

**Save**

CONFIGURACIÓN	VALOR SUGERIDO	DESCRIPCIÓN
Nombre	OnOff	Nombre descriptivo para el parámetro.

CONFIGURACIÓN	VALOR SUGERIDO	DESCRIPCIÓN
Es global	Desactivado	Casilla que indica si el valor de este parámetro se aplica globalmente a todos los comandos del proyecto.
Obligatorio	Activado	Casilla que indica si es obligatorio especificar un valor para este parámetro antes de completar el comando.
Plantilla de respuesta	On (Encendido) u Off (Apagado)	Pregunta para solicitar el valor de este parámetro cuando no se conoce.
Tipo	String	Tipo de parámetro, como número, cadena o fecha y hora.
Configuración	Lista de cadenas	En el caso de las cadenas, una lista de cadenas limita las entradas a un conjunto de valores posibles.
Valores de la lista de cadenas.	on, off	Para un parámetro de lista de cadenas, el conjunto de valores posibles y sus sinónimos.

- A continuación, vuelva a seleccionar el icono de para agregar un segundo parámetro que represente el nombre de los dispositivos. En este ejemplo, un televisor y un ventilador.

CONFIGURACIÓN	VALOR SUGERIDO	DESCRIPCIÓN
Nombre	SubjectDevice	Nombre descriptivo para el parámetro.
Es global	Desactivado	Casilla que indica si el valor de este parámetro se aplica globalmente a todos los comandos del proyecto.
Obligatorio	Activado	Casilla que indica si es obligatorio especificar un valor para este parámetro antes de completar el comando.
Plantilla de respuesta	¿Qué dispositivo?	Pregunta para solicitar el valor de este parámetro cuando no se conoce.
Tipo	String	Tipo de parámetro, como número, cadena o fecha y hora.
Configuración	Lista de cadenas	En el caso de las cadenas, una lista de cadenas limita las entradas a un conjunto de valores posibles.
Valores de la lista de cadenas.	TV, ventilador	Para un parámetro de lista de cadenas, el conjunto de valores posibles y sus sinónimos.

CONFIGURACIÓN	VALOR SUGERIDO	DESCRIPCIÓN
Sinónimos (TV)	televisión, tele	Sinónimos opcionales para cada valor posible de un parámetro de lista de cadenas.

## Adición de oraciones de ejemplo

Con los parámetros, resulta útil agregar oraciones de ejemplo que cubran todas las combinaciones posibles. Por ejemplo:

1. Información del parámetro completa: "turn {OnOff} the {SubjectDevice}" .
2. Información del parámetro parcial: "turn it {OnOff}" .
3. Información del parámetro faltante: "turn something" .

Las oraciones de ejemplo con diferentes cantidades de información permiten que la aplicación de comandos personalizados resuelva las resoluciones de una sola captura y las resoluciones multigiro con información parcial.

Teniendo esto en cuenta, edite las oraciones de ejemplo para usar los parámetros como se sugiere a continuación.

### TIP

En el editor de oraciones de ejemplo, use llaves para hacer referencia a los parámetros. -

turn {OnOff} the {SubjectDevice} Use la finalización mediante tabulación para hacer referencia a los parámetros creados previamente.

```

1  turn {OnOff} the {SubjectDevice}
2  {SubjectDevice} {OnOff}
3  turn it {OnOff}
4  turn something {OnOff}
5  turn something

```

```

turn {OnOff} the {SubjectDevice}
{SubjectDevice} {OnOff}
turn it {OnOff}
turn something {OnOff}
turn something

```

## Adición de parámetros a la regla de finalización

Modifique la regla de finalización que creó en [la guía de inicio rápido anterior](#):

1. Agregue una nueva condición y seleccione el parámetro obligatorio. Seleccione `onoff` y `SubjectDevice` .
2. Edite la acción de respuesta de voz para que use `onoff` y `SubjectDevice` :

```
Ok, turning {OnOff} the {SubjectDevice}
```

## Prueba

Abra el panel de chat de prueba y pruebe algunas interacciones.

- Entrada: Turn off the tv (apagar el televisor)

- Salida: Ok, turning on the TV (de acuerdo, encenderé el televisor)
- Entrada: turn off the television (apaga el televisor)
- Salida: Ok, turning on the TV (de acuerdo, encenderé el televisor)
- Entrada: turn it off (apágalo)
- Salida: ¿Qué dispositivo?
- Entrada: the tv (el televisor)
- Salida: Ok, turning on the TV (de acuerdo, encenderé el televisor)

## Pasos siguientes

[Inicio rápido: conexión a una aplicación de comandos personalizados con el SDK de Voz \(versión preliminar\)](#)

# Inicio rápido: conexión a una aplicación de comandos personalizados con el SDK de Voz (versión preliminar)

13/01/2020 • 16 minutes to read • [Edit Online](#)

Después de crear una aplicación de comandos personalizados hospedada, puede empezar a hablar con ella desde un dispositivo cliente.

En este artículo, hará lo siguiente:

- Publicación de una aplicación de comandos personalizados y obtención de un identificador de aplicación (Id. de aplicación)
- Creación de una aplicación cliente con el SDK de Voz para que pueda comunicarse con su aplicación de comandos personalizados

## Requisitos previos

Se requiere una aplicación de comandos personalizados para completar este artículo. Si aún no ha creado una aplicación de comandos personalizados, puede hacerlo en estos inicios rápidos anteriores:

- [Inicio rápido: Creación de un comando personalizado \(versión preliminar\)](#)
- [Inicio rápido: Creación de un comando personalizado con parámetros \(versión preliminar\)](#)

También necesitará:

- [Visual Studio 2019](#)
- Una clave de suscripción de Azure para el servicio Voz. [Obtenga una gratis](#) o créela en [Azure Portal](#).

## Opcional: Empiece rápidamente

En este inicio rápido se describe, paso a paso, cómo hacer que una aplicación cliente se conecte a la aplicación de comandos personalizados. Si prefiere sumergirse de lleno, el código fuente completo y listo para compilar utilizado en este inicio rápido está disponible en los [ejemplos del SDK de Voz](#) bajo la carpeta `quickstart`.

## Paso 1: Publicación de la aplicación de comandos personalizados

1. Abra la [aplicación de comandos personalizados creada anteriormente](#) y seleccione **Publicar**.

[Publish your application](#)

**Device Control Quickstart**'s has been successfully trained and published.

To use your application from the Speech SDK.

- Enter the speech command app id [REDACTED] in your client application.
- Enter one of your subscription keys in your client application.
- [Learn more](#).

2. Copie el Id. de la aplicación de la notificación de publicación para usarlo posteriormente.

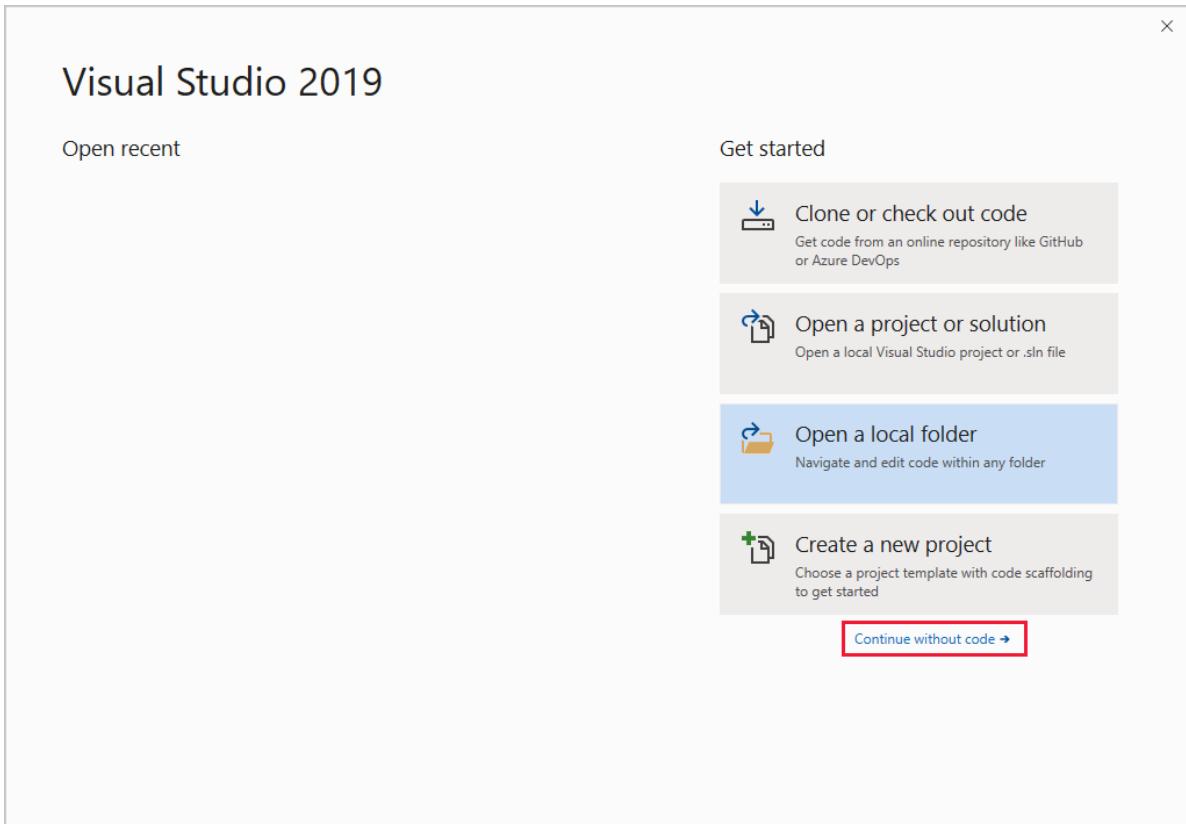
## Paso 2: Creación de un proyecto de Visual Studio

Para crear un proyecto de Visual Studio C++ para el desarrollo de la Plataforma universal de Windows (UPW), debe configurar las opciones de desarrollo de Visual Studio, crear el proyecto, seleccionar la arquitectura de destino, configurar la captura de audio e instalar el SDK de voz.

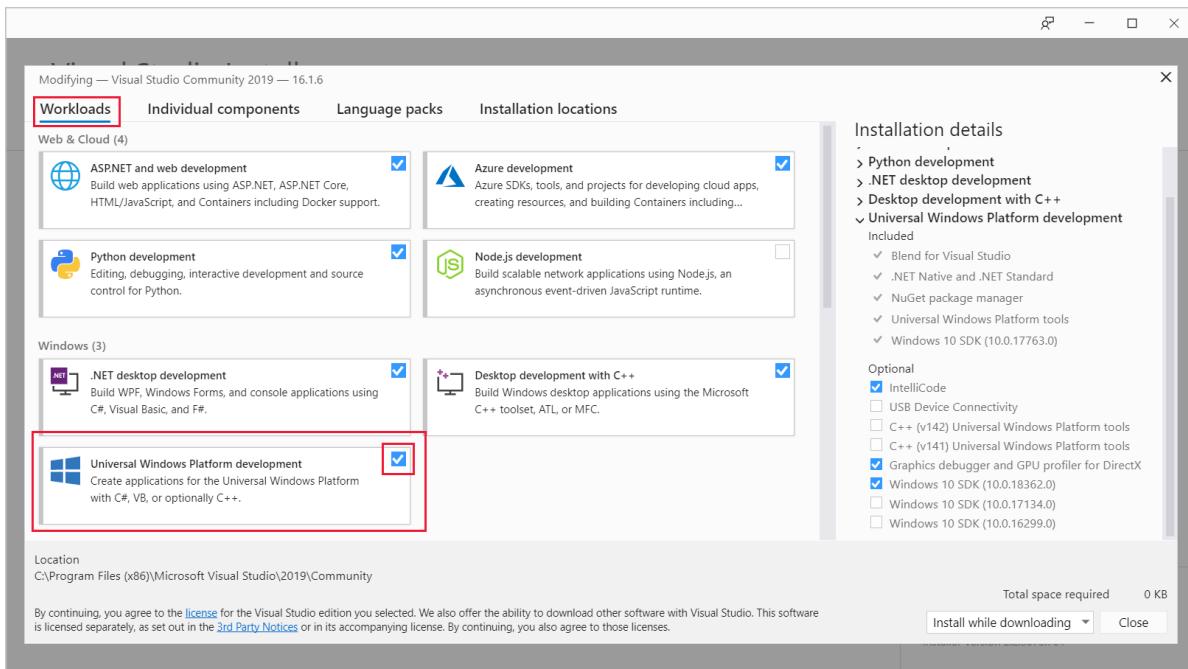
### Configuración de las opciones de desarrollo de Visual Studio

Para empezar, asegúrese de que Visual Studio está configurado correctamente para el desarrollo de UWP:

1. Abra Visual Studio 2019 para mostrar la ventana **Inicio**.



2. Seleccione **Continuar sin código** para ir al IDE de Visual Studio.
3. En la barra de menús de Visual Studio, seleccione **Herramientas > Get Tools and Features** (Obtener herramientas y características) para abrir el Instalador de Visual Studio y ver el cuadro de diálogo **Modificar**.

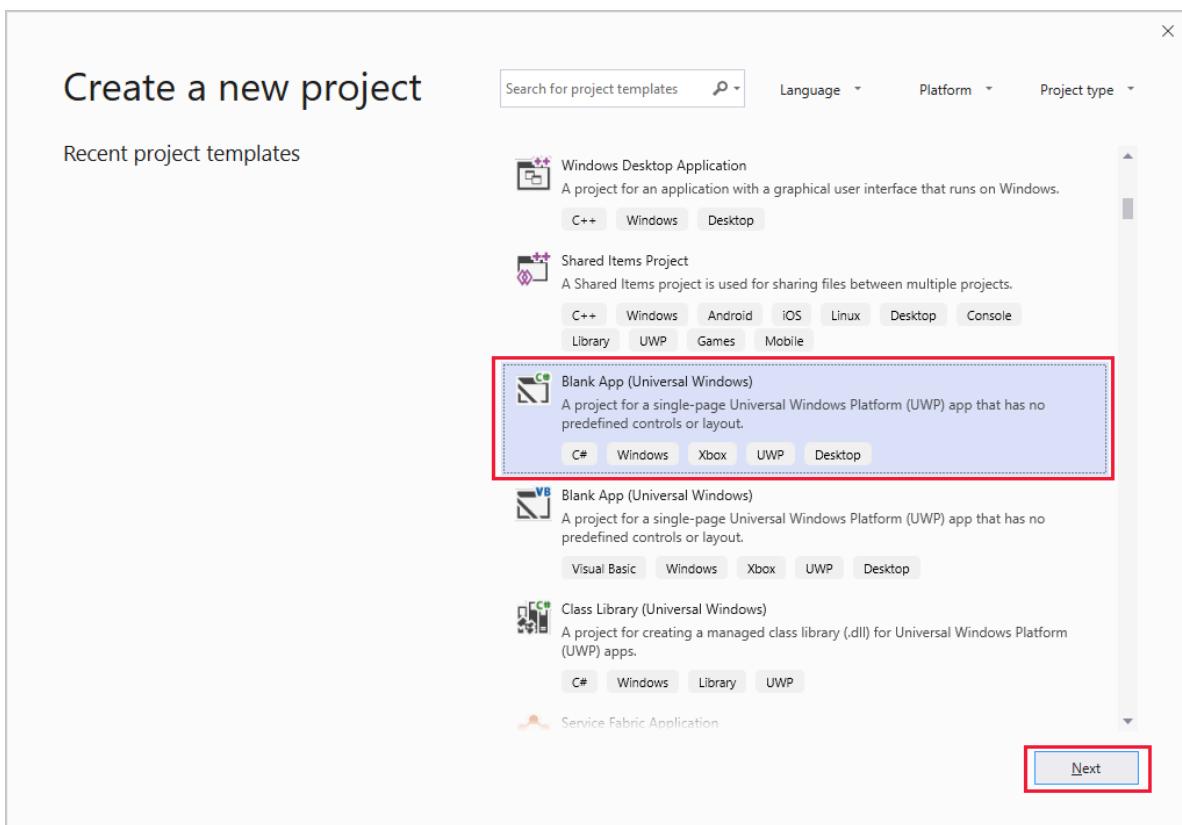


4. En la pestaña **Cargas de trabajo**, en **Windows**, busque la carga de trabajo **Desarrollo de la Plataforma universal de Windows**. Si la casilla situada junto a esa carga de trabajo ya está seleccionada, cierre el cuadro de diálogo **Modificar** y vaya al paso 6.
5. Active la casilla **Desarrollo de la Plataforma universal de Windows**, seleccione **Modificar** y, a continuación, en el cuadro de diálogo **Antes de comenzar**, seleccione **Continuar** para instalar la carga de trabajo de desarrollo de UWP. La instalación de la nueva característica puede tardar un rato.
6. Cierre el Instalador de Visual Studio.

#### **Cree el proyecto y seleccione la arquitectura de destino.**

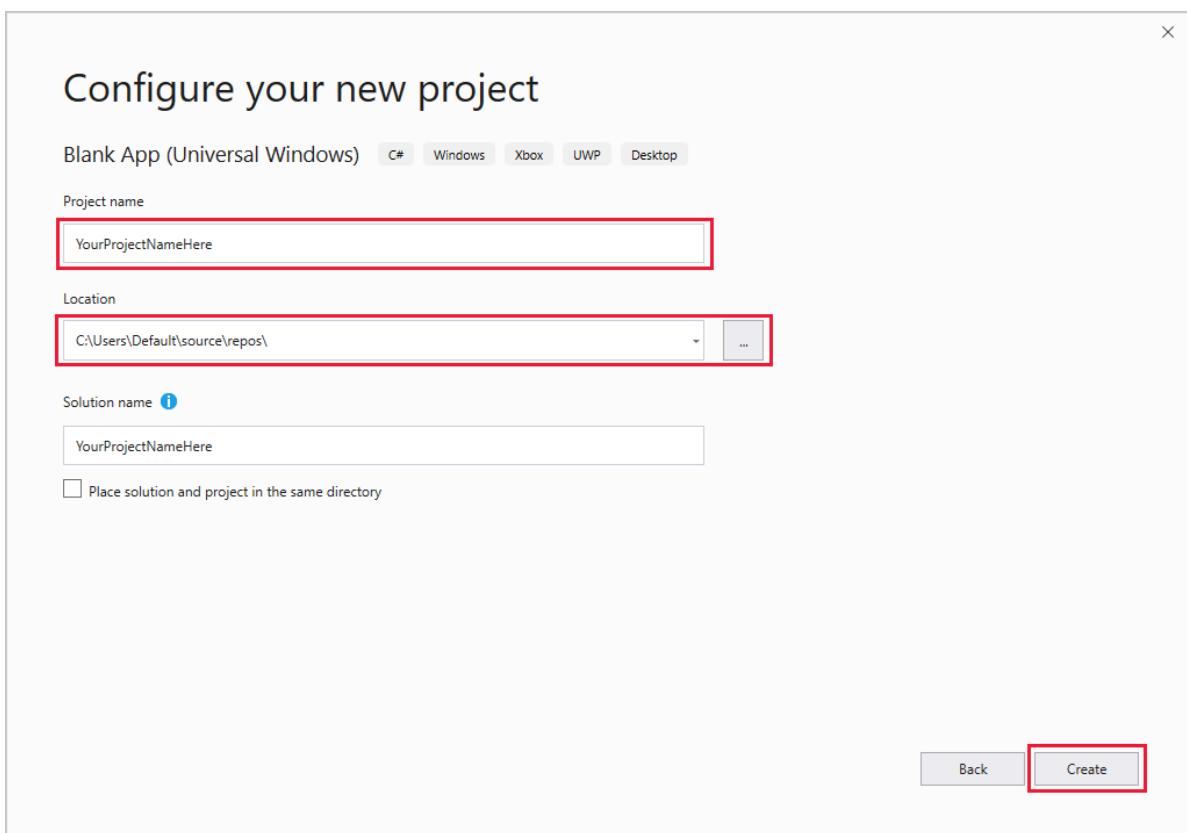
Después, cree el proyecto.

1. En la barra de menús de Visual Studio, elija **Archivo > Nuevo > Proyecto** para mostrar la ventana **Crear un nuevo proyecto**.



2. Busque y seleccione **Aplicación vacía (Windows universal)**. Asegúrese de seleccionar la versión de C# de este tipo de proyecto (en lugar de Visual Basic).

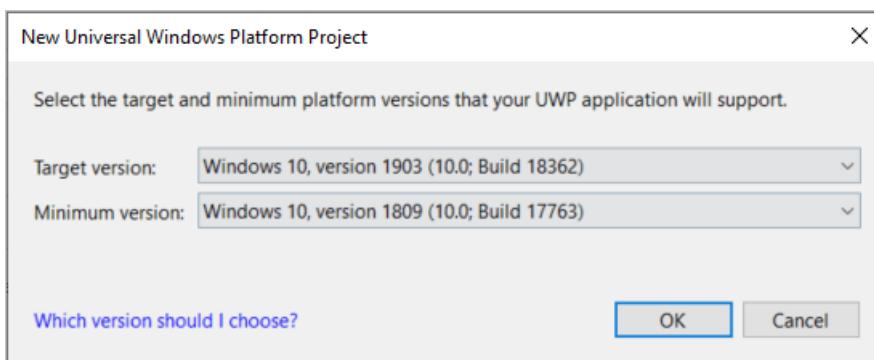
3. Seleccione **Siguiente** para abrir la pantalla **Configurar el nuevo proyecto**.



4. En **Nombre del proyecto**, escriba `helloworld`.

5. En **Ubicación**, vaya la carpeta en la que desea guardar el proyecto y selecciónela o créela.

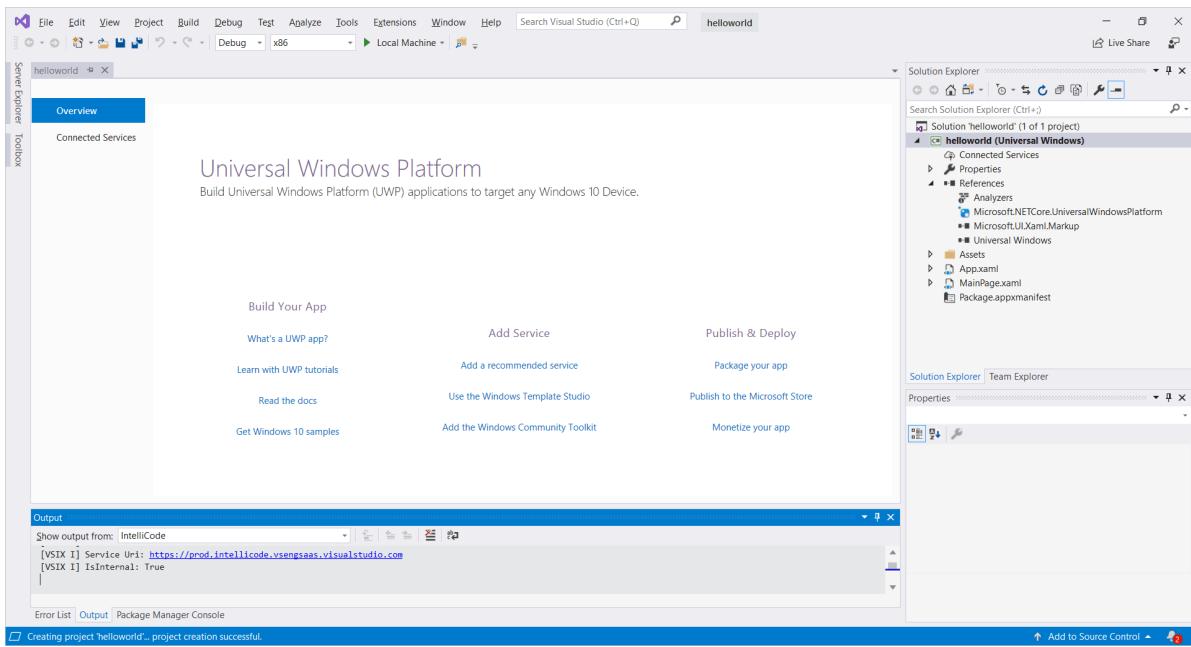
6. Seleccione **Crear** para ir a la ventana **Nuevo proyecto de la Plataforma universal de Windows**.



7. En **Versión mínima** (el segundo cuadro desplegable), elija **Windows 10 Fall Creators Update (10.0; Compilación 16299)**, que es el requisito mínimo para el SDK de voz.

8. En **Versión de destino** (el primer cuadro desplegable), elija un valor idéntico o posterior al valor de **Versión mínima**.

9. Seleccione **Aceptar**. Se le devolverá al IDE de Visual Studio, con el nuevo proyecto creado y visible en el panel **Explorador de soluciones**.



Ahora seleccione la arquitectura de la plataforma de destino. En la barra de herramientas de Visual Studio, busque el cuadro desplegable **Plataformas de solución**. (Si no lo ve, elija **Ver > Barra de herramientas > Estándar** para mostrar la barra de herramientas que contiene **Plataformas de solución**). Si está ejecutando Windows de 64 bits, elija **x64** en el cuadro desplegable. Windows de 64 bits puede ejecutar también aplicaciones de 32 bits, por lo que puede elegir **x86**, si lo prefiere.

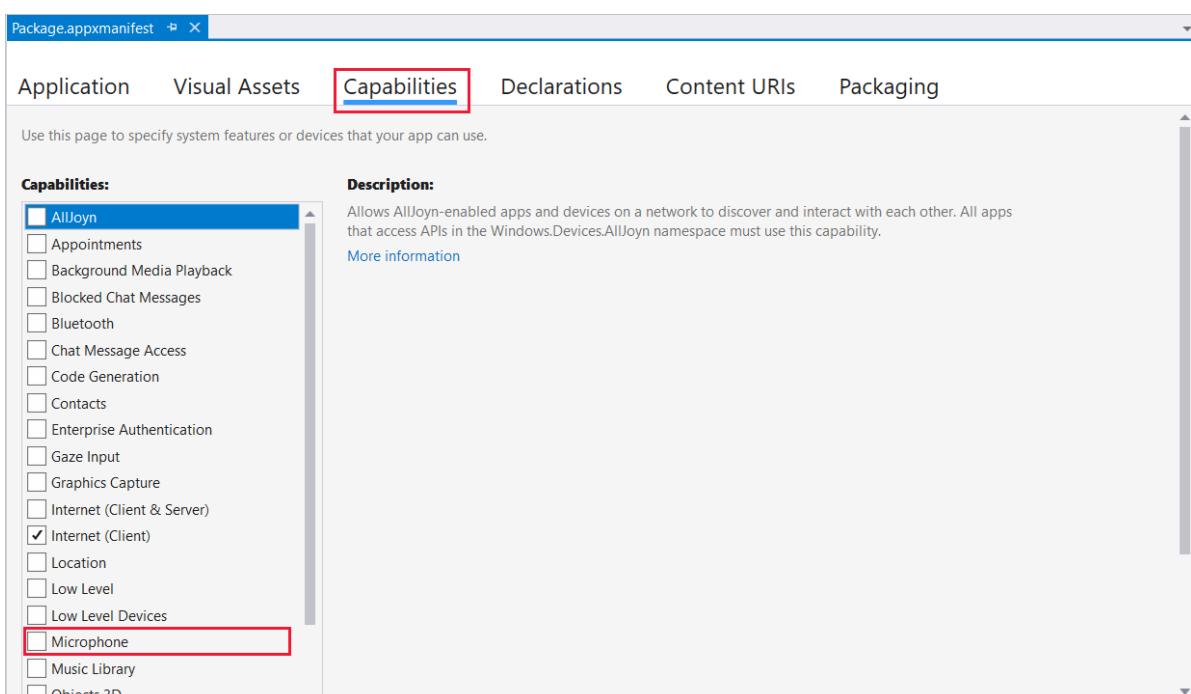
#### NOTE

El SDK de Voz solo admite los procesadores compatibles con Intel. Actualmente no se admiten los procesadores ARM.

## Configuración de la captura de audio

A continuación, permita que el proyecto capture la entrada de audio:

1. En el **Explorador de soluciones**, haga doble clic en **Package.appxmanifest** para abrir el manifiesto de aplicación del paquete.
2. Seleccione la pestaña **Funcionalidades**.



3. Active la casilla de la funcionalidad **Micrófono**.

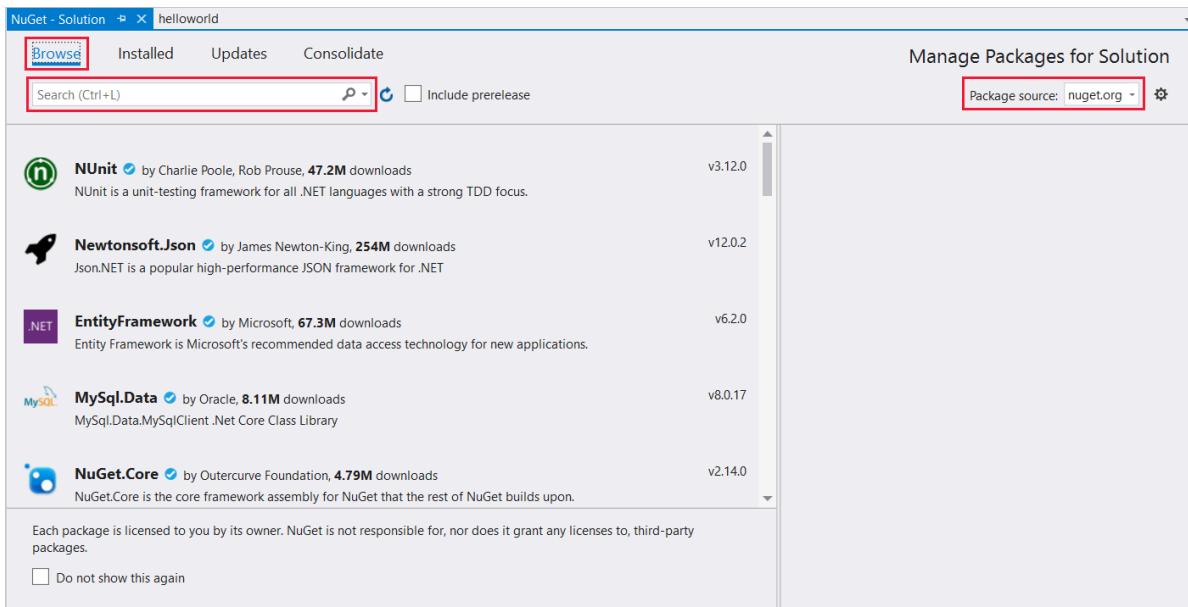
4. En la barra de menús, elija **Archivo > Guardar Package.appxmanifest** para guardar los cambios.

## Instalación de Speech SDK

Por último, instale el [paquete NuGet del SDK de voz](#) y haga referencia al SDK de voz en el proyecto:

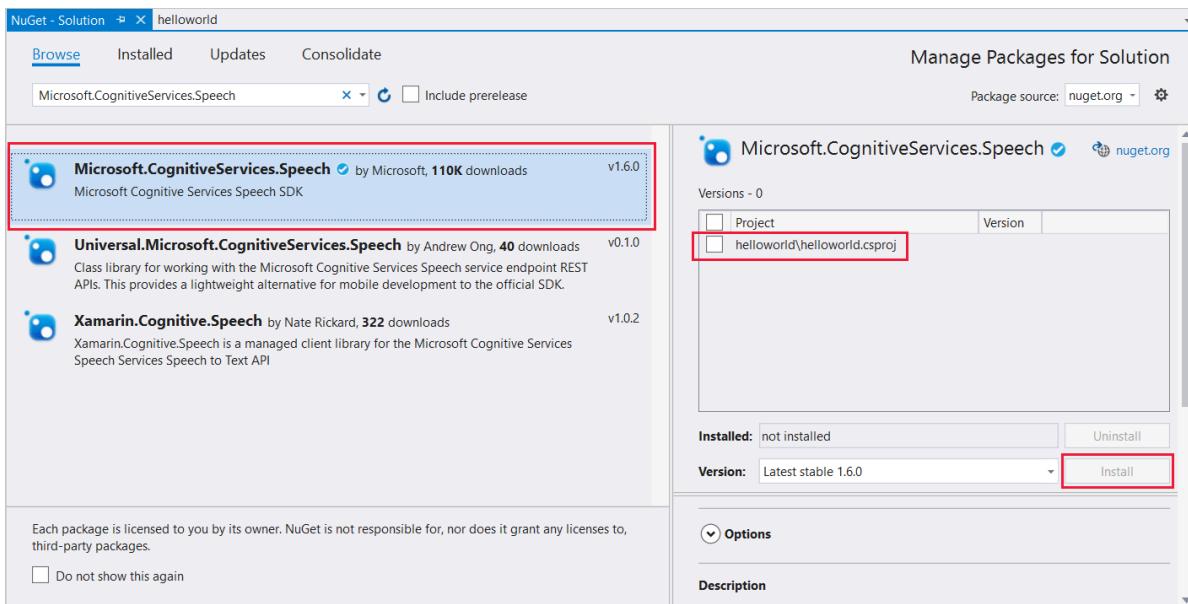
1. En el **Explorador de soluciones**, haga clic con el botón derecho en la solución y elija **Manage NuGet Packages for Solution** (Administrar paquetes NuGet para la solución) para ir a la ventana **NuGet: solución**.

2. Seleccione **Examinar**.



3. En **Origen del paquete**, elija [nuget.org](#).

4. En el cuadro de búsqueda **Buscar**, escriba `Microsoft.CognitiveServices.Speech` y, a continuación, elija el paquete cuando aparezca en los resultados de la búsqueda.



5. En el panel de estado del paquete situado junto a los resultados de la búsqueda, seleccione el proyecto **HelloWorld**.

6. Seleccione **Instalar**.

7. En el cuadro de diálogo **Vista previa de los cambios**, seleccione **Aceptar**.

8. En el cuadro de diálogo **Aceptación de licencia**, vea la licencia y, a continuación, seleccione **Acepto**. La instalación del paquete comienza y, cuando se completa, el panel **Salida** muestra un mensaje similar al siguiente: `Successfully installed 'Microsoft.CognitiveServices.Speech 1.7.0' to helloworld`.

## Paso 3: Incorporación de código de ejemplo

En este paso, se agrega el código XAML que define la interfaz de usuario de la aplicación y el código subyacente en C# de la implementación.

### Código XAML

Cree la interfaz de usuario de la aplicación; para ello, agregue el código XAML.

1. En el **Explorador de soluciones**, abra `MainPage.xaml`.
2. En la vista XAML del diseñador, reemplace todo el contenido por el siguiente fragmento de código:

```
<Page
    x:Class="helloworld.MainPage"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:local="using:helloworld"
    xmlns:d="http://schemas.microsoft.com/expressionblend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    mc:Ignorable="d"
    Background="{ThemeResource ApplicationPageBackgroundThemeBrush}">

    <Grid>
        <StackPanel Orientation="Vertical" HorizontalAlignment="Center"
            Margin="20,50,0,0" VerticalAlignment="Center" Width="800">
            <Button x:Name="EnableMicrophoneButton" Content="Enable Microphone"
                Margin="0,0,10,0" Click="EnableMicrophone_ButtonClicked"
                Height="35"/>
            <Button x:Name="ListenButton" Content="Talk"
                Margin="0,10,10,0" Click="ListenButton_ButtonClicked"
                Height="35"/>
            <StackPanel x:Name="StatusLabel" Orientation="Vertical"
                RelativePanel.AlignBottomWithPanel="True"
                RelativePanel.AlignRightWithPanel="True"
                RelativePanel.AlignLeftWithPanel="True">
                <TextBlock x:Name="StatusLabel" Margin="0,10,10,0"
                    TextWrapping="Wrap" Text="Status:" FontSize="20"/>
                <Border x:Name="StatusBorder" Margin="0,0,0,0">
                    <ScrollViewer VerticalScrollMode="Auto"
                        VerticalScrollBarVisibility="Auto" MaxHeight="200">
                        <!-- Use LiveSetting to enable screen readers to announce
                            the status update. -->
                    <TextBlock
                        x:Name="StatusBlock" FontWeight="Bold"
                        AutomationProperties.LiveSetting="Assertive"
                        MaxWidth="{Binding ElementName=Splitter, Path=ActualWidth}"
                        Margin="10,10,10,20" TextWrapping="Wrap" />
                    </ScrollViewer>
                </Border>
            </StackPanel>
        </StackPanel>
        <MediaElement x:Name="mediaElement"/>
    </Grid>
</Page>
```

La vista Diseño se actualiza para mostrar la interfaz de usuario de la aplicación.

### Código subyacente de C#

Agregue el código fuente subyacente para que la aplicación funcione según lo previsto. El código fuente

subyacente incluye:

- Instrucciones `using` obligatorias para los espacios de nombres `Speech` y `Speech.Dialog`.
- Una implementación sencilla para garantizar el acceso del micrófono, conectado a un controlador de botón
- Asistentes básicos de la interfaz de usuario para presentar los errores y mensajes en la aplicación
- Un punto de aterrizaje para la ruta de acceso del código de inicialización que se rellenará más tarde
- Una asistente para la reproducción de texto a voz (sin compatibilidad con streaming)
- Un controlador de botón vacía para empezar a escuchar que se rellenará más tarde

Agregue el código fuente subyacente como se muestra a continuación:

1. En **Explorador de soluciones**, abra el archivo de código fuente subyacente `MainPage.xaml.cs` (agrupado en `MainPage.xaml`).
2. Reemplace el contenido del archivo por el siguiente código:

```
using Microsoft.CognitiveServices.Speech;
using Microsoft.CognitiveServices.Speech.Audio;
using Microsoft.CognitiveServices.Speech.Dialog;
using System;
using System.Diagnostics;
using System.IO;
using System.Text;
using Windows.Foundation;
using Windows.Storage.Streams;
using Windows.UI.Xaml;
using Windows.UI.Xaml.Controls;
using Windows.UI.Xaml.Media;

namespace helloworld
{
    public sealed partial class MainPage : Page
    {
        private DialogServiceConnector connector;

        private enum NotifyType
        {
            StatusMessage,
            ErrorMessage
        };

        public MainPage()
        {
            this.InitializeComponent();
        }

        private async void EnableMicrophone_ButtonClicked(
            object sender, RoutedEventArgs e)
        {
            bool isMicAvailable = true;
            try
            {
                var mediaCapture = new Windows.Media.Capture.MediaCapture();
                var settings =
                    new Windows.Media.Capture.MediaCaptureInitializationSettings();
                settings.StreamingCaptureMode =
                    Windows.Media.Capture.StreamingCaptureMode.Audio;
                await mediaCapture.InitializeAsync(settings);
            }
            catch (Exception)
            {
                isMicAvailable = false;
            }
            if (!isMicAvailable)
            {
```

```

        await Windows.System.Launcher.LaunchUriAsync(
            new Uri("ms-settings:privacy-microphone"));
    }
    else
    {
        NotifyUser("Microphone was enabled", NotifyType.StatusMessage);
    }
}

private void NotifyUser(
    string strMessage, NotifyType type = NotifyType.StatusMessage)
{
    // If called from the UI thread, then update immediately.
    // Otherwise, schedule a task on the UI thread to perform the update.
    if (Dispatcher.HasThreadAccess)
    {
        UpdateStatus(strMessage, type);
    }
    else
    {
        var task = Dispatcher.RunAsync(
            Windows.UI.Core.CoreDispatcherPriority.Normal,
            () => UpdateStatus(strMessage, type));
    }
}

private void UpdateStatus(string strMessage, NotifyType type)
{
    switch (type)
    {
        case NotifyType.StatusMessage:
            StatusBorder.Background = new SolidColorBrush(
                Windows.UI.Colors.Green);
            break;
        case NotifyType.ErrorMessage:
            StatusBorder.Background = new SolidColorBrush(
                Windows.UI.Colors.Red);
            break;
    }
    StatusBlock.Text += string.IsNullOrEmpty(StatusBlock.Text)
        ? strMessage : "\n" + strMessage;

    if (!string.IsNullOrEmpty(StatusBlock.Text))
    {
        StatusBorder.Visibility = Visibility.Visible;
        StatusPanel.Visibility = Visibility.Visible;
    }
    else
    {
        StatusBorder.Visibility = Visibility.Collapsed;
        StatusPanel.Visibility = Visibility.Collapsed;
    }
    // Raise an event if necessary to enable a screen reader
    // to announce the status update.
    var peer =
Windows.UI.Xaml.Automation.Peers.FrameworkElementAutomationPeer.FromElement(StatusBlock);
    if (peer != null)
    {
        peer.RaiseAutomationEvent(
            Windows.UI.Xaml.Automation.Peers.AutomationEvents.LiveRegionChanged);
    }
}

// Waits for and accumulates all audio associated with a given
// PullAudioOutputStream and then plays it to the MediaElement. Long spoken
// audio will create extra latency and a streaming playback solution
// (that plays audio while it continues to be received) should be used --
// see the samples for examples of this.
private void SynchronouslyPlayActivityAudio()

```

```

private void SyncOnAudioActivity(ActivityResult activity)
{
    PullAudioOutputStream activityAudio;
    var playbackStreamWithHeader = new MemoryStream();
    playbackStreamWithHeader.Write(Encoding.ASCII.GetBytes("RIFF"), 0, 4); // ChunkID
    playbackStreamWithHeader.Write(BitConverter.GetBytes(UInt32.MaxValue), 0, 4); // ChunkSize;
    max
    playbackStreamWithHeader.Write(Encoding.ASCII.GetBytes("WAVE"), 0, 4); // Format
    playbackStreamWithHeader.Write(Encoding.ASCII.GetBytes("fmt "), 0, 4); // Subchunk1ID
    playbackStreamWithHeader.Write(BitConverter.GetBytes(16), 0, 4); // Subchunk1Size: PCM
    playbackStreamWithHeader.Write(BitConverter.GetBytes(1), 0, 2); // AudioFormat: PCM
    playbackStreamWithHeader.Write(BitConverter.GetBytes(1), 0, 2); // NumChannels: mono
    playbackStreamWithHeader.Write(BitConverter.GetBytes(16000), 0, 4); // SampleRate: 16kHz
    playbackStreamWithHeader.Write(BitConverter.GetBytes(32000), 0, 4); // ByteRate
    playbackStreamWithHeader.Write(BitConverter.GetBytes(2), 0, 2); // BlockAlign
    playbackStreamWithHeader.Write(BitConverter.GetBytes(16), 0, 2); // BitsPerSample: 16-bit
    playbackStreamWithHeader.Write(Encoding.ASCII.GetBytes("data"), 0, 4); // Subchunk2ID
    playbackStreamWithHeader.Write(BitConverter.GetBytes(UInt32.MaxValue), 0, 4); //
    Subchunk2Size

    byte[] pullBuffer = new byte[2056];

    uint lastRead = 0;
    do
    {
        lastRead = activityAudio.Read(pullBuffer);
        playbackStreamWithHeader.Write(pullBuffer, 0, (int)lastRead);
    }
    while (lastRead == pullBuffer.Length);

    var task = Dispatcher.RunAsync(
        Windows.UI.Core.CoreDispatcherPriority.Normal, () =>
    {
        mediaElement.SetSource(
            playbackStreamWithHeader.AsRandomAccessStream(), "audio/wav");
        mediaElement.Play();
    });
}

private void InitializeDialogServiceConnector()
{
    // New code will go here
}

private async void ListenButton_Clicked(
    object sender, RoutedEventArgs e)
{
    // New code will go here
}
}
}

```

3. Agregue el siguiente código al cuerpo del método `InitializeDialogServiceConnector`.

```
// This code creates the `DialogServiceConnector` with your subscription information.  
// create a DialogServiceConfig by providing a Custom Commands application id and Cognitive Services  
subscription key  
// the RecoLanguage property is optional (default en-US); note that only en-US is supported in Preview  
const string speechCommandsApplicationId = "YourApplicationId"; // Your application id  
const string speechSubscriptionKey = "YourSpeechSubscriptionKey"; // Your subscription key  
const string region = "YourServiceRegion"; // The subscription service region. Note: only 'westus2' is  
currently supported  
  
var speechCommandsConfig = DialogServiceConfig.FromSpeechCommandsAppId(speechCommandsApplicationId,  
speechSubscriptionKey, region);  
speechCommandsConfig SetProperty(PropertyId.SpeechServiceConnection_RecoLanguage, "en-us");  
connector = new DialogServiceConnector(speechCommandsConfig);
```

4. Reemplace las cadenas `YourApplicationId`, `YourSpeechSubscriptionKey` y `YourServiceRegion` CON SUS propios valores para la aplicación, la suscripción de voz y la [región](#).
5. Anexe el siguiente fragmento de código al final del cuerpo del método `InitializeDialogServiceConnector`.

```

// This code sets up handlers for events relied on by `DialogServiceConnector` to communicate its
activities,
// speech recognition results, and other information.
//
// ActivityReceived is the main way your client will receive messages, audio, and events
connector.ActivityReceived += async (sender, activityReceivedEventArgs) =>
{
    NotifyUser(
        $"Activity received, hasAudio={activityReceivedEventArgs.HasAudio} activity=
{activityReceivedEventArgs.Activity}");

    if (activityReceivedEventArgs.HasAudio)
    {
        SynchronouslyPlayActivityAudio(activityReceivedEventArgs.Audio);
    }
};

// Canceled will be signaled when a turn is aborted or experiences an error condition
connector.Canceled += (sender, canceledEventArgs) =>
{
    NotifyUser($"Canceled, reason={canceledEventArgs.Reason}");
    if (canceledEventArgs.Reason == CancellationReason.Error)
    {
        NotifyUser(
            $"Error: code={canceledEventArgs.ErrorCode}, details={canceledEventArgs.ErrorDetails}");
    }
};

// Recognizing (not 'Recognized') will provide the intermediate recognized text
// while an audio stream is being processed
connector.Recognizing += (sender, recognitionEventArgs) =>
{
    NotifyUser($"Recognizing! in-progress text={recognitionEventArgs.Result.Text}");
};

// Recognized (not 'Recognizing') will provide the final recognized text
// once audio capture is completed
connector.Recognized += (sender, recognitionEventArgs) =>
{
    NotifyUser($"Final speech-to-text result: '{recognitionEventArgs.Result.Text}'");
};

// SessionStarted will notify when audio begins flowing to the service for a turn
connector.SessionStarted += (sender, sessionEventArgs) =>
{
    NotifyUser($"Now Listening! Session started, id={sessionEventArgs.SessionId}");
};

// SessionStopped will notify when a turn is complete and
// it's safe to begin listening again
connector.SessionStopped += (sender, sessionEventArgs) =>
{
    NotifyUser($"Listening complete. Session ended, id={sessionEventArgs.SessionId}");
};

```

6. Agregue el siguiente fragmento de código al cuerpo del método `ListenButton_ButtonClicked` de la clase `MainPage`.

```

// This code sets up `DialogServiceConnector` to listen, since you already established the
configuration and
// registered the event handlers.
if (connector == null)
{
    InitializeDialogServiceConnector();
    // Optional step to speed up first interaction: if not called,
    // connection happens automatically on first use
    var connectTask = connector.ConnectAsync();
}

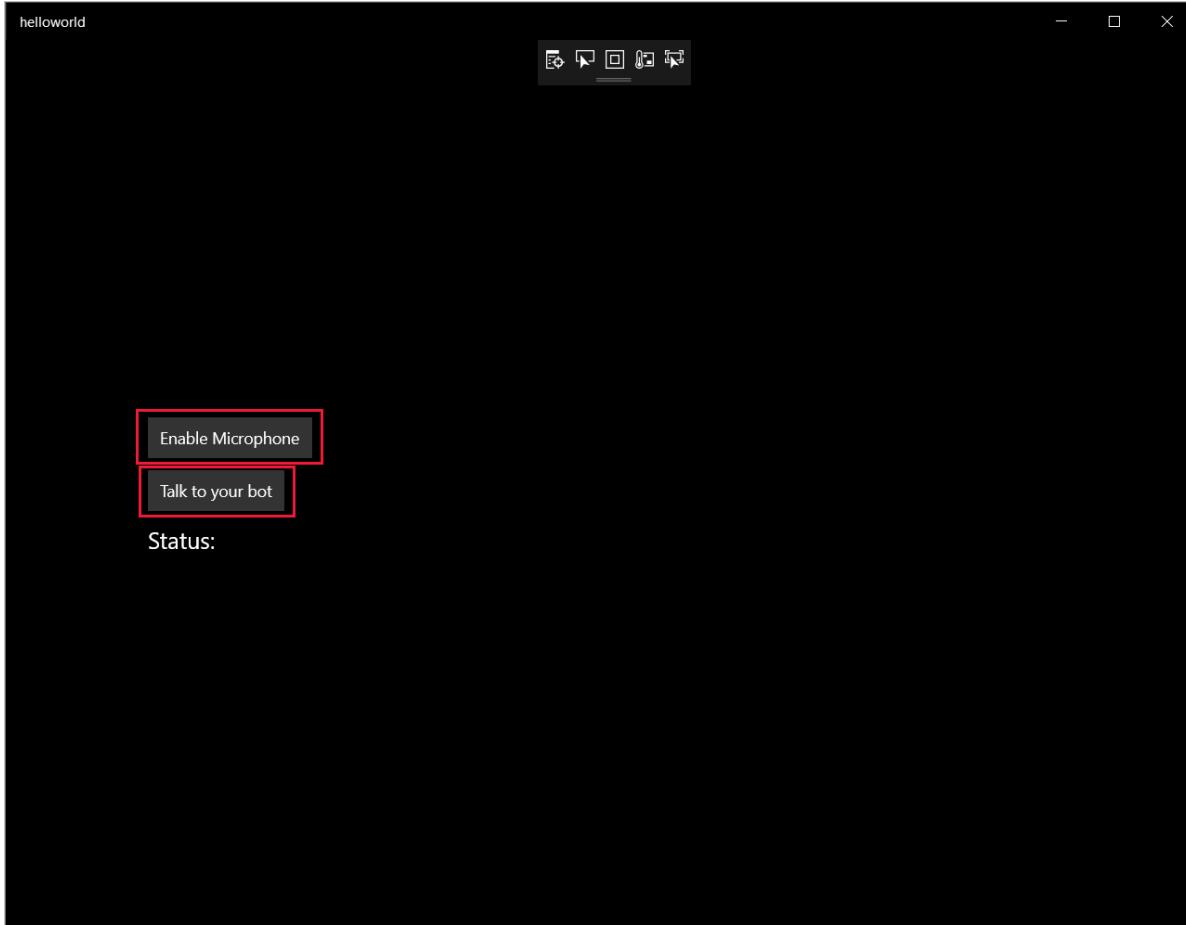
try
{
    // Start sending audio
    await connector.ListenOnceAsync();
}
catch (Exception ex)
{
    NotifyUser($"Exception: {ex.ToString()}", NotifyType.ErrorMessage);
}

```

7. En la barra de menús, elija **Archivo > Guardar todo** para guardar los cambios.

## Compilación y ejecución de la aplicación

1. En la barra de menús, elija **Compilar > Compilar solución** para compilar la aplicación. El código se debería compilar sin errores ahora.
2. Elija **Depurar > Iniciar depuración** o presione **F5** para iniciar la aplicación. Aparece la ventana **HelloWorld**.



3. Seleccione **Habilitar micrófono**. Si aparece la solicitud de permiso de acceso, seleccione **Sí**.

Let helloworld access your microphone?

Let helloworld access your microphone?

To change this later, go to the Settings app.

Yes

No

4. Seleccione **Hablar** y diga una expresión o frase en inglés en el micrófono del dispositivo. Lo que diga se transmitirá al canal Direct Line Speech y se transcribirá en texto, que aparece en la misma ventana.

## Pasos siguientes

[Uso de realización de comandos en el cliente con el SDK de Voz \(versión preliminar\)](#) Procedimiento: adición de validaciones a los parámetros de comandos personalizados (versión preliminar)

# Instrucciones: Realización de comandos en el cliente con el SDK de Voz (versión preliminar)

13/01/2020 • 4 minutes to read • [Edit Online](#)

Para finalizar las tareas con una aplicación de comandos personalizada, puede enviar cargas personalizadas a un dispositivo cliente conectado.

En este artículo, hará lo siguiente:

- Definir y enviar una carga JSON personalizada desde la aplicación de comandos personalizados.
- Recibir y visualizar el contenido de la carga JSON personalizada desde una aplicación cliente del SDK de Voz en C# de UWP.

## Requisitos previos

- [Visual Studio 2019](#)
- Una clave de suscripción de Azure para el servicio de voz.
  - [Obtenga una gratis](#) o créela en [Azure Portal](#).
- Una aplicación de comandos personalizados creada previamente.
  - [Inicio rápido: Creación de un comando personalizado con parámetros \(versión preliminar\)](#)
- Una aplicación cliente habilitada para el SDK de Voz.
  - [Inicio rápido: conexión a una aplicación de comandos personalizados con el SDK de Voz \(versión preliminar\)](#)

## Opcional: Empiece rápidamente

En este artículo se describe, paso a paso, cómo hacer que una aplicación cliente se comunique con la aplicación de comandos personalizados. Si prefiere sumergirse de lleno, el código fuente completo y listo para compilar utilizado en este artículo está disponible en los [ejemplos del SDK de Voz](#).

## Finalización de la carga útil de JSON

1. Abra la aplicación de comandos personalizados creada anteriormente desde [Speech Studio](#).
2. Compruebe la sección **Reglas de finalización** para asegurarse de estar usando la regla creada anteriormente que responde al usuario.
3. Para enviar una carga directamente al cliente, cree una nueva regla con una acción Enviar actividad.

# Completion Rules

Completion rules are executed once we're ready to fulfill, or complete, the command. They occur when all the required parameters are gathered. [Learn more](#)

## Rule Name

Name \*

UpdateDeviceState

## Conditions



RequiredParameters(OnOff, SubjectDevice)

## Actions



SendActivity (name, UpdateDeviceState)

Save

CONFIGURACIÓN	VALOR SUGERIDO	DESCRIPCIÓN
Nombre de la regla	UpdateDeviceState	Nombre que describe el propósito de la regla.
Condiciones	Parámetro obligatorio: <code>OnOff</code> y <code>SubjectDevice</code> .	Condiciones que determinan cuándo se puede ejecutar la regla.
Acciones	<code>SendActivity</code> (consulte a continuación)	Acción que se realizará cuando la condición de la regla sea true.

Type \*

Send activity to client



#### Activity Content

```
1  {
2      "type": "event",
3      "name": "UpdateDeviceState",
4      "state": "{OnOff}",
5      "device": "{SubjectDevice}"
6  }
```

Cancel

Save

```
{
    "name": "UpdateDeviceState",
    "state": "{OnOff}",
    "device": "{SubjectDevice}"
}
```

## Creación de objetos visuales para el estado de encendido o apagado del dispositivo

En [Inicio rápido: conexión a una aplicación de comandos personalizados con el SDK de Voz \(versión preliminar\)](#) creó una aplicación cliente del SDK de voz que administraba comandos como `turn on the tv`, `turn off the fan`. Ahora, agregue algunos objetos visuales para que pueda ver el resultado de esos comandos.

Agregue cuadros etiquetados con texto que indique **Encendido** o **Apagado** mediante el siguiente XML agregado a `MainPage.xaml.cs`

```
<StackPanel Orientation="Horizontal" HorizontalAlignment="Center" Margin="20">
    <Grid x:Name="Grid_TV" Margin="50, 0" Width="100" Height="100" Background="LightBlue">
        <StackPanel>
            <TextBlock Text="TV" Margin="0, 10" TextAlignment="Center"/>
            <TextBlock x:Name="State_TV" Text="Off" TextAlignment="Center"/>
        </StackPanel>
    </Grid>
    <Grid x:Name="Grid_Fan" Margin="50, 0" Width="100" Height="100" Background="LightBlue">
        <StackPanel>
            <TextBlock Text="Fan" Margin="0, 10" TextAlignment="Center"/>
            <TextBlock x:Name="State_Fan" Text="Off" TextAlignment="Center"/>
        </StackPanel>
    </Grid>
</StackPanel>
```

## Control de la carga personalizable

Ahora que ha creado una carga JSON, puede agregar una referencia a la biblioteca [JSON.NET](#) para controlar la deserialización.

NuGet: fulfillment ➔ X

Browse    Installed    Updates 1

newtonsoft.json X ⟳  Include prerelease

 **Newtonsoft.Json** ✓ by James Newton-King, 299M downloads v12.0.2  
Json.NET is a popular high-performance JSON framework for .NET

En `InitializeDialogServiceConnector`, agregue lo siguiente al controlador de eventos `ActivityReceived`. El código adicional extraerá la carga de la actividad y cambiará el estado visual del televisor o el ventilador según corresponda.

```
connector.ActivityReceived += async (sender, activityReceivedEventArgs) =>
{
    NotifyUser($"Activity received, hasAudio={activityReceivedEventArgs.HasAudio} activity=
{activityReceivedEventArgs.Activity}");

    dynamic activity = JsonConvert.DeserializeObject(activityReceivedEventArgs.Activity);
    var payload = activity?.Value;

    if(payload?.name == "SetDeviceState")
    {
        var state = payload?.state;
        var device = payload?.device;
        switch(device)
        {
            case "tv":
                State_TV.Text = state;
                break;
            case "fan":
                State_Fan.Text = state;
                break;
            default:
                NotifyUser($"Received request to set unsupported device {device} to {state}");
                break;
        }
    }

    if (activityReceivedEventArgs.HasAudio)
    {
        SynchronouslyPlayActivityAudio(activityReceivedEventArgs.Audio);
    }
};
```

## Prueba

1. Inicio de la aplicación
2. Seleccione Habilitar micrófono.
3. Seleccionar el botón Hablar.
4. Diga `turn on the tv`.
5. El estado visual del televisor debe cambiar a "Encendido"

## Pasos siguientes

[Uso de adición de validaciones a los parámetros de comandos personalizados \(versión preliminar\)](#)

# Instrucciones: adición de validaciones a los parámetros de comandos personalizados (versión preliminar)

14/01/2020 • 3 minutes to read • [Edit Online](#)

En este artículo, aprenderá a agregar validaciones a los parámetros y a solicitar correcciones.

## Prerequisites

Debe haber completado los pasos descritos en los siguientes artículos:

- [Inicio rápido: Creación de un comando personalizado \(versión preliminar\)](#)
- [Inicio rápido: Creación de un comando personalizado con parámetros \(versión preliminar\)](#)

## Creación de un comando SetTemperature

Para demostrar las validaciones, vamos a crear un nuevo comando que permita al usuario establecer la temperatura.

1. Abra la aplicación de comandos personalizados creada anteriormente en [Speech Studio](#)
2. Cree un nuevo comando **SetTemperature**.
3. Agregue un parámetro para la temperatura de destino.
4. Agregue una validación para el parámetro temperature.

## New Validation

X

Min Value \*

60

Max Value \*

80

Response Template \*

1 - Sorry, I can only set between 60 and 80 degrees

Create

Cancel

CONFIGURACIÓN	VALOR SUGERIDO	DESCRIPCIÓN
Nombre	Temperatura	Nombre descriptivo para el parámetro de comando.
Obligatorio	true	Casilla que indica si es obligatorio especificar un valor para este parámetro antes de completar el comando.
Plantilla de respuesta	"- ¿Qué temperatura le gustaría?"	Pregunta para solicitar el valor de este parámetro cuando no se conoce.
Tipo	Number	Tipo de parámetro, como número, cadena o fecha y hora.
Validación	Valor mínimo: 60, Valor máximo: 80	En el caso de los parámetros de número, el intervalo de valores permitido para el parámetro.
Plantilla de respuesta	"- Lo sentimos, el valor solo puede establecerse entre 60 y 80 grados"	Pregunta para solicitar un valor actualizado si se produce un error en la validación.

5. Agregue algunas oraciones de ejemplo.

```
set the temperature to {Temperature} degrees
change the temperature to {Temperature}
set the temperature
change the temperature
```

6. Agregue una regla de finalización para confirmar el resultado.

CONFIGURACIÓN	VALOR SUGERIDO	DESCRIPCIÓN
Nombre de la regla	Mensaje de confirmación	Nombre que describe el propósito de la regla.
Condiciones	Parámetro obligatorio: temperature	Condiciones que determinan cuándo se puede ejecutar la regla.
Acciones	SpeechResponse: "- De acuerdo, se establecerá en {temperature} grados"	Acción que se realizará cuando la condición de la regla sea true.

#### TIP

En este ejemplo se utiliza una respuesta de voz para confirmar el resultado. Para obtener ejemplos sobre cómo completar el comando con una acción de cliente, consulte: [Cómo: Fulfill Commands on the client with the Speech SDK \(Preview\)](#)  
(Instrucciones: realización de comandos en el cliente con el SDK de Voz).

## Prueba

Seleccione el panel de prueba y pruebe algunas interacciones.

- Entrada: establezca la temperatura en 72 grados.
- Salida: "de acuerdo, se establecerá en 72 grados".
- Entrada: establezca la temperatura en 45 grados.
- Salida: "Lo sentimos, el valor solo puede establecerse entre 60 y 80 grados"
- Entrada: cámbiala a 72 grados en su lugar.
- Salida: "de acuerdo, se establecerá en 72 grados".

## Pasos siguientes

[Cómo: Adición de una confirmación a un comando personalizado \(versión preliminar\)](#)

# Instrucciones: Agregar una corrección de un paso a un comando personalizado (versión preliminar)

15/01/2020 • 3 minutes to read • [Edit Online](#)

En este artículo, aprenderá a agregar una confirmación en un paso a un comando.

La corrección en un paso se usa para actualizar un comando que se acaba de completar.

Por ejemplo, si acaba de configurar una alarma, puede cambiar de opinión y actualizar la hora de esta.

- Entrada: Establecer alarma para mañana a mediodía
- Salida: "De acuerdo, alarma establecida para el 06/12/2019 a las 12:00:00"
- Entrada: No, mañana a las 13:00
- Salida: "Vale"

Tenga en cuenta que esto implica que usted, como desarrollador, tiene un mecanismo para actualizar la alarma en la aplicación de back-end.

## Prerequisites

Debe haber completado los pasos descritos en los siguientes artículos:

- [Inicio rápido: Creación de un comando personalizado \(versión preliminar\)](#)
- [Inicio rápido: Creación de un comando personalizado con parámetros \(versión preliminar\)](#)
- [Cómo: Adición de una confirmación a un comando personalizado \(versión preliminar\)](#)

## Incorporación de reglas avanzadas para la corrección en un paso

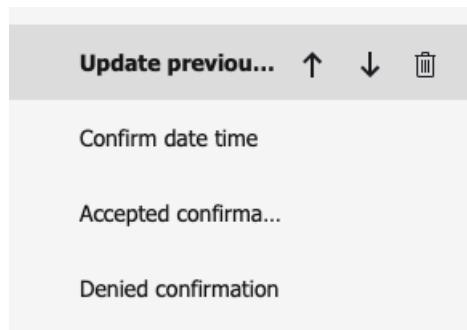
Para demostrar la corrección en un paso, vamos a ampliar el comando **SetAlarm** creado en el [procedimiento de confirmación](#).

1. Agregue una regla avanzada para actualizar la alarma anterior.

Esta regla pedirá al usuario que confirme la fecha y la hora de la alarma y espera una confirmación (sí/no) para el siguiente turno.

CONFIGURACIÓN	VALOR SUGERIDO	DESCRIPCIÓN
Nombre de la regla	Actualizar alarma anterior	Nombre que describe el propósito de la regla.
Condiciones	UpdateLastCommand y parámetro obligatorio: DateTime	Condiciones que determinan cuándo se puede ejecutar la regla.
Acciones	SpeechResponse: "actualizando la alarma anterior a {DateTime}"	Acción que se realizará cuando la condición de la regla sea true.
Estado después de la ejecución	Comando Completar	Estado del usuario después del turno

2. Mueva la regla que acaba de crear a la parte superior de las reglas avanzadas (desplácese por la regla en el panel y haga clic en la flecha Arriba).



#### NOTE

En una aplicación real, en la sección Acciones de esta regla también enviará una actividad al cliente o llamará a un punto de conexión HTTP para actualizar la alarma en el sistema.

## Prueba

Seleccione el panel de prueba y pruebe algunas interacciones.

- Entrada: Establecer alarma para mañana a mediodía
- Salida: "¿Seguro que desea establecer una alarma para el 07/12/2019 a las 12:00:00?"
- Entrada: Sí
- Salida: "De acuerdo, alarma establecida para el 07/12/2019 a las 12:00:00"
- Entrada: No, mañana a las 13:00
- Salida: "Actualizando la alarma anterior al 12/07/2019 a las 13:00:00"

# Tutorial: Habilitación del bot con voz mediante el SDK de voz

14/01/2020 • 43 minutes to read • [Edit Online](#)

Ahora puede usar la potencia del servicio de voz para habilitar fácilmente un bot de chat.

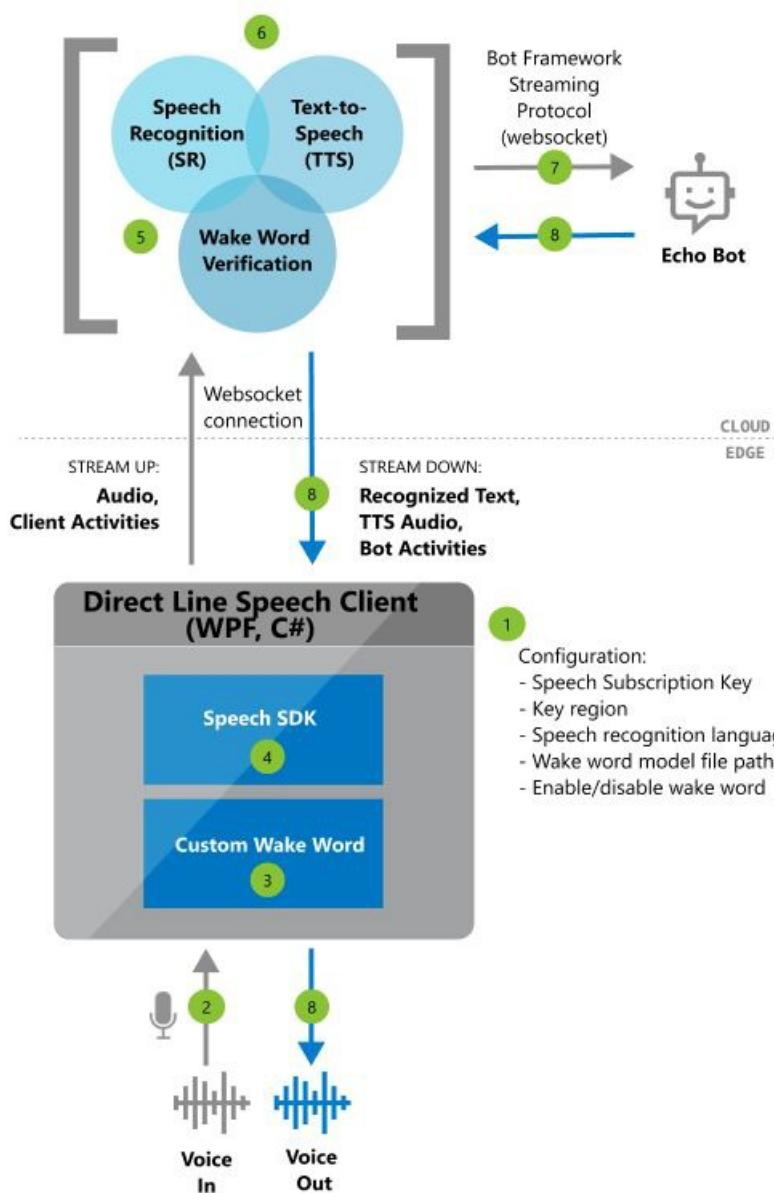
En este tutorial, creará un bot de eco con Microsoft Bot-Framework, lo implementará en Azure y lo registrará en el canal Direct Line Speech de Bot-Framework. A continuación, configurará una aplicación cliente de ejemplo para Windows que le permita hablar con el bot y oír su respuesta.

Este tutorial está pensado para desarrolladores que empiezan a trabajar con Azure, con bots de Bot-Framework, con Direct Line Speech o con el SDK de voz y quieren crear rápidamente un sistema de trabajo con codificación limitada. No es necesario tener experiencia ni estar familiarizado con estos servicios.

Al final de este ejercicio, habrá configurado un sistema que funcionará de la siguiente manera:

1. La aplicación cliente de ejemplo está configurada para conectarse al canal Direct Line Speech y al bot de eco.
2. El audio se graba a través del micrófono predeterminado al presionar el botón (o se graba continuamente si se activa la palabra clave personalizada)
3. Opcionalmente, se produce la detección de la palabra clave, que canaliza el streaming de audio a la nube.
4. Con el SDK de voz, la aplicación se conecta al canal Direct Line Speech y transmite el audio.
5. Opcionalmente, se produce una comprobación de la palabra clave de mayor precisión en el servicio.
6. El audio se pasa al servicio de reconocimiento de voz y se transcribe en texto.
7. El texto reconocido se pasa al bot de eco como una actividad de Bot Framework.
8. El texto de la respuesta se convierte en audio mediante el servicio de conversión de texto a voz (TTS) y se vuelve a transmitir a la aplicación cliente para su reproducción.

## Direct Line Speech channel Co-located services



### NOTE

Los pasos de este tutorial no requieren un servicio de pago. Como nuevo usuario de Azure, podrá usar créditos de su suscripción de prueba de Azure y del nivel gratis del servicio de voz para completar este tutorial.

Este tutorial abarca lo siguiente:

- Crear nuevos recursos de Azure
- Compilar, probar e implementar el ejemplo de bot de eco en una instancia de Azure App Service
- Registrar el bot con el canal Direct Line Speech
- Compilar y ejecutar el cliente de Direct Line Speech para interactuar con el bot de eco
- Agregar la activación de la palabra clave personalizada
- Aprender cómo cambiar el idioma de la voz hablada y reconocida

## Prerequisites

Para completar este tutorial, necesitará lo siguiente:

- Un equipo con Windows 10 con un micrófono y altavoces (o auriculares) que funcionen

- [Visual Studio 2017](#) o cualquier versión posterior
- [.NET Core SDK](#) versión 2.1 o posterior.
- Una cuenta de Azure. [Regístrate gratis.](#)
- Una cuenta [GitHub](#)
- [Git para Windows](#)

## Crear un grupo de recursos

La aplicación cliente que creará en este tutorial utiliza una serie de servicios de Azure. Para reducir el tiempo de ida y vuelta de las respuestas del bot, deberá asegurarse de que estos servicios se encuentran en la misma región de Azure. En esta sección, se crea un grupo de recursos en la región **Oeste de EE. UU.** Este grupo de recursos se usará al crear recursos individuales para Bot-Framework, el canal Direct Line Speech y el servicio de voz.

1. Inicie sesión en [Azure Portal](#).
2. En el panel de la izquierda, seleccione **Grupos de recursos**. Despues, haga clic en **Agregar** para agregar un nuevo grupo de recursos.
3. Se le pedirá que proporcione algo de información:
  - Establezca **Suscripción** en **Evaluación gratuita** (también puede usar una suscripción existente).
  - Escriba un nombre para el **grupo de recursos**. Se recomienda **SpeechEchoBotTutorial-ResourceGroup**.
  - En la lista desplegable **Región**, seleccione **Oeste de EE. UU.**
4. Haga clic en **Revisar y crear**. Debería ver un banner que indica **Validación superada**.
5. Haga clic en **Crear**. La creación del grupo de recursos tardará unos minutos.
6. Al igual que con los recursos que creará más adelante en este tutorial, es una buena idea anclar este grupo de recursos al panel para facilitar el acceso. Si desea anclar este grupo de recursos, haga clic en el icono de anclaje en la esquina superior derecha del panel.

### Elección de una región de Azure

Si quiere usar una región diferente para este tutorial, estos factores pueden limitar las opciones:

- Asegúrese de que usa una [región de Azure compatible](#).
- El canal Direct Line Speech utiliza el servicio de conversión de texto a voz, que tiene voces estándar y neuronales. Las voces neuronales se [limitan a regiones específicas de Azure](#).
- Las claves de la evaluación gratuitas pueden estar restringidas a una región específica.

Para más información sobre las regiones, consulte [Ubicaciones de Azure](#).

## Crear recursos

Ahora que tiene un grupo de recursos en la región **Oeste de EE. UU.**, el siguiente paso es crear recursos individuales para cada servicio que usará en este tutorial.

### Creación de recurso del servicio de voz

Para crear un recurso de voz, siga estas instrucciones:

1. Vaya a [Azure Portal](#) y seleccione **Crear un recurso** en el panel de navegación izquierdo.
2. En la barra de búsqueda, escriba **Voz**.
3. Seleccione **Voz** y, a continuación, haga clic en **Crear**.
4. Se le pedirá que proporcione algo de información:
  - Asigne un **nombre** al recurso. Se recomienda **SpeechEchoBotTutorial-Speech**.
  - En **Suscripción**, asegúrese de que esté seleccionada la opción **Evaluación gratuita**.

- En **Ubicación**, seleccione **Oeste de EE. UU.**
  - En **Plan de tarifa**, seleccione **F0**. Es el plan gratuito.
  - En **Grupo de recursos**, seleccione **SpeechEchoBotTutorial-ResourceGroup**.
- Después de haber escrito la información necesaria, haga clic en **Crear**. La creación del recurso puede tardar unos minutos.
  - Más adelante en este tutorial necesitará las claves de suscripción para este servicio. Puede acceder a estas claves en cualquier momento desde **Información general** del recurso (Administrar claves) o **Claves**.

En este momento, compruebe que el grupo de recursos (**SpeechEchoBotTutorial-ResourceGroup**) tiene un recurso de voz:

NAME	TYPE	LOCATION
SpeechEchoBotTutorial-Speech	Cognitive Services	Oeste de EE. UU.

### Crear un plan de Azure App Service

El primer paso consiste en crear un plan de App Service. Un plan de App Service define un conjunto de recursos de proceso para que una aplicación web se ejecute.

- Vaya a [Azure Portal](#) y seleccione **Crear un recurso** en el panel de navegación izquierdo.
- En la barra de búsqueda, escriba **Plan de App Service**. A continuación, busque y seleccione la tarjeta **Plan del servicio de aplicaciones** en los resultados de la búsqueda.
- Haga clic en **Crear**.
- Se le pedirá que proporcione algo de información:
  - Establezca **Suscripción** en **Evaluación gratuita** (también puede usar una suscripción existente).
  - En **Grupo de recursos**, seleccione **SpeechEchoBotTutorial-ResourceGroup**.
  - Asigne un **nombre** al recurso. Se recomienda **SpeechEchoBotTutorial-AppServicePlan**
  - En **Sistema operativo**, seleccione **Windows**.
  - En **Región**, seleccione **Oeste de EE. UU.**
  - En **Plan de tarifa**, asegúrese de que está seleccionado **Estándar S1**. Este debería ser el valor predeterminado. Si no es así, asegúrese de establecer el **Sistema operativo** en **Windows** tal y como se describió anteriormente.
- Haga clic en **Revisar y crear**. Debería ver un banner que indica **Validación superada**.
- Haga clic en **Crear**. La creación del grupo de recursos tardará unos minutos.

En este momento, compruebe que el grupo de recursos (**SpeechEchoBotTutorial-ResourceGroup**) tiene dos recursos:

NAME	TYPE	LOCATION
SpeechEchoBotTutorial-AppServicePlan	Plan de servicio de aplicación	Oeste de EE. UU.
SpeechEchoBotTutorial-Speech	Cognitive Services	Oeste de EE. UU.

## Creación de un bot de eco

Ahora que ha creado algunos recursos, vamos a crear un bot. Vamos a empezar con el ejemplo del bot de eco, que, como su nombre implica, muestra el texto que ha escrito como respuesta. No se preocupe, el código de ejemplo está listo para su uso sin precisar ningún cambio. Está configurado para funcionar con el canal Direct Line Speech, al que nos conectaremos después de haber implementado el bot en Azure.

#### NOTE

Las instrucciones siguientes, así como información adicional sobre el bot de eco, están disponibles en el [archivo Léame del ejemplo en GitHub](#).

## Ejecución del bot de ejemplo en su máquina

1. Clone el repositorio de ejemplos:

```
git clone https://github.com/Microsoft/botbuilder-samples.git
```

2. Inicie Visual Studio.

3. En la barra de herramientas, seleccione **Archivo > Abrir > Proyecto o solución** y abra la solución de proyecto Bot de eco:

```
samples\csharp_dotnetcore\02.echo-bot\EchoBot.sln
```

4. Una vez cargado el proyecto, presione **F5** para compilar y ejecutar el proyecto.

## Prueba del ejemplo de bot con Bot Framework Emulator

[Bot Framework Emulator](#) es una aplicación de escritorio que permite que los desarrolladores de bots prueben y depuren sus bots de manera local o remota mediante un túnel. Emulator admite texto escrito como entrada (no voz). El bot responderá con texto. Siga estos pasos para usar Bot Framework Emulator para probar la ejecución local de Bot de eco, con entrada de texto y salida de texto. Después de implementar el bot de Azure, se probará con entrada de voz y salida de voz.

1. Instalación de [Bot Framework Emulator](#) versión 4.3.0 o superior

2. Inicie Bot Framework Emulator y abra el bot:

- **Archivo -> Open Bot** (Abrir bot).

3. Escriba la dirección URL del bot. Por ejemplo:

```
http://localhost:3978/api/messages
```

y presione "Conectar".

4. El bot debe saludarle inmediatamente con el mensaje de bienvenida. (de la directiva). Escriba un mensaje de texto y confirme que obtiene una respuesta del bot.

## Implementación del bot en Azure App Service

El siguiente paso es implementar el bot de eco en Azure. Hay varias maneras de implementar un bot, pero en este tutorial nos centraremos en publicarlo directamente desde Visual Studio.

#### NOTE

También, puede implementar un bot con la [CLI de Azure](#) y las [plantillas de implementación](#).

1. En Visual Studio, abra el bot de eco que se ha configurado para su uso con el canal Direct Line Speech:

```
samples\csharp_dotnetcore\02.echo-bot\EchoBot.sln
```

2. En el **Explorador de soluciones**, haga clic con el botón derecho en el proyecto **EchoBot** y seleccione **Publicar...**
3. Se abrirá una nueva ventana denominada **Elegir un destino de publicación**.
4. Seleccione **App Service** en el panel de navegación izquierdo, seleccione **Crear nuevo** y haga clic en **Publicar**.
5. Cuando aparece la ventana **Crear servicio de aplicaciones**:
  - Haga clic en **Agregar una cuenta** e inicie sesión con las credenciales de la cuenta de Azure. Si ya ha iniciado sesión, seleccione la cuenta que desee en la lista desplegable.
  - Para el **nombre de la aplicación**, deberá escribir un nombre único global para el bot. Este nombre se usa para crear una dirección URL única del bot. Se rellenará un valor predeterminado que incluye la fecha y la hora (por ejemplo: "EchoBot20190805125647"). En este tutorial, utilice los valores predeterminados.
  - En **Suscripción**, establezcala en **Evaluación gratuita**.
  - En **Grupo de recursos**, seleccione **SpeechEchoBotTutorial-ResourceGroup**.
  - En **Plan de hospedaje**, seleccione **SpeechEchoBotTutorial-AppServicePlan**.
6. Haga clic en **Crear**

7. Debería ver un mensaje de operación correcta en Visual Studio que tiene el siguiente aspecto:

```
Publish Succeeded.  
Web App was published successfully https://EchoBot20190805125647.azurewebsites.net/
```

8. Debería abrirse el explorador predeterminado y mostrar una página que diga: "El bot está listo".
9. En este punto, compruebe el grupo de recursos **SpeechEchoBotTutorial-ResourceGroup** en Azure Portal y confirme que hay tres recursos:

NAME	TYPE	LOCATION
EchoBot20190805125647	App Service	Oeste de EE. UU.
SpeechEchoBotTutorial-AppServicePlan	Plan de App Service	Oeste de EE. UU.
SpeechEchoBotTutorial-Speech	Cognitive Services	Oeste de EE. UU.

## Habilitación de sockets web

Tendrá que hacer un pequeño cambio de configuración para que el bot pueda comunicarse con el canal Direct Line Speech mediante sockets web. Siga estos pasos para habilitar los sockets web:

1. Vaya a [Azure Portal](#) y busque la instancia de App Service. El recurso debe tener un nombre similar a **EchoBot20190805125647** (el nombre único de la aplicación).
2. En el panel de navegación izquierdo, en **Configuración**, haga clic en **Configuración**.
3. Seleccione la pestaña **Configuración general**.
4. Busque el botón de alternancia para **Sockets web** y establezcalo en **Activado**.
5. Haga clic en **Save(Guardar)**.

**TIP**

Puede usar los controles que se encuentran en la parte superior de la página de Azure App Service para detener o reiniciar el servicio. Esto puede resultar útil para solucionar problemas.

## Creación de un registro de canal

Ahora que ha creado una instancia de Azure App Service para hospedar el bot, el siguiente paso es crear un **registro de canales de bot**. La creación de un registro de canal es un requisito previo para registrar el bot con canales de Bot-Framework, incluido el canal Direct Line Speech.

**NOTE**

Para más información sobre cómo los bots aprovechan los canales, consulte [Conexión de un bot a canales](#).

1. El primer paso consiste en crear un nuevo recurso para el registro. En [Azure Portal](#), haga clic en **Crear un recurso**.
2. En la barra de búsqueda, escriba **bot**; después de que aparezcan los resultados, seleccione **Bot Channels Registration** (Registro de canales de bot).
3. Haga clic en **Crear**.
4. Se le pedirá que proporcione algo de información:
  - En **Identificador de bot**, escriba **SpeechEchoBotTutorial-BotRegistration**.
  - En **Suscripción**, seleccione **Evaluación gratuita**.
  - En **Grupo de recursos**, seleccione **SpeechEchoBotTutorial-ResourceGroup**.
  - En **Ubicación**, seleccione **Oeste de EE. UU.**
    - En **Plan de tarifa**, seleccione **F0**.
    - En **Messaging endpoint** (Punto de conexión de mensajería), escriba la dirección URL de la aplicación web con la ruta de acceso `/api/messages` anexada al final. Por ejemplo, si el nombre de la aplicación globalmente único era **EchoBot20190805125647**, el punto de conexión de mensajería sería `https://EchoBot20190805125647.azurewebsites.net/api/messages/`.
    - En **Application Insights**, puede establecerlo en **Desconectar**. Para más información, consulte [Análisis del bot](#).
    - Ignore **Auto create App ID and password** (Creación automática del identificador y contraseña de la aplicación).
5. En la parte inferior de la hoja **Registro de canales de bot**, haga clic en **Crear**.

En este punto, compruebe el grupo de recursos **SpeechEchoBotTutorial-ResourceGroup** en Azure Portal.

Ahora debería mostrar cuatro recursos:

NAME	TYPE	LOCATION
EchoBot20190805125647	App Service	Oeste de EE. UU.
SpeechEchoBotTutorial-AppServicePlan	Plan de App Service	Oeste de EE. UU.
SpeechEchoBotTutorial-BotRegistration	Registro de canales de bot	Global
SpeechEchoBotTutorial-Speech	Cognitive Services	Oeste de EE. UU.

#### IMPORTANT

El recurso del registro de canales de bot mostrará la región global aunque haya seleccionado Oeste de EE. UU. Se espera que esto sea así.

## Registro del canal Direct Line Speech

Ahora es el momento de registrar el bot con el canal Direct Line Speech. Este canal es lo que se usa para crear una conexión entre el bot de eco y una aplicación cliente compilada con el SDK de voz.

1. Busque y abra el recurso **SpeechEchoBotTutorial-BotRegistration** en [Azure Portal](#).
2. En el panel de navegación izquierdo, seleccione **Canales**.
  - Busque **Más canales**, busque y haga clic en **Direct Line Speech**.
  - Revise el texto de la página titulada **Configurar Direct Line Speech** y, a continuación, expanda el menú desplegable con la etiqueta de la cuenta de servicio de Cognitive.
  - Seleccione el recurso de voz que creó anteriormente (por ejemplo, **SpeechEchoBotTutorial-Speech**) en el menú para asociar el bot a la clave de suscripción de voz.
  - Haga clic en **Save(Guardar)**.
3. En la barra de navegación izquierda, haga clic en **Configuración**.
  - Active la casilla de verificación **Enable Streaming Endpoint** (Habilitar punto de conexión de streaming). Esto es necesario para habilitar un protocolo de comunicación basado en sockets web entre el bot y el canal Direct Line Speech.
  - Haga clic en **Save(Guardar)**.

#### TIP

Si quiere más información, consulte [Conexión de un bot a Direct Line Speech](#). En esta página se incluye información adicional así como problemas conocidos.

## Creación del cliente de Direct Line Speech

En este paso, va a crear el cliente de Direct Line Speech. El cliente es una aplicación de Windows Presentation Foundation (WPF) en C# que usa el [SDK de voz](#) para administrar la comunicación con el bot mediante el canal Direct Line Speech. Úselo para interactuar con el bot y probarlo antes de escribir una aplicación cliente personalizada.

El cliente de Direct Line Speech tiene una interfaz de usuario simple que le permite configurar la conexión al bot, ver la conversación de texto, ver las actividades de Bot-Framework en formato JSON y mostrar las tarjetas adaptables. También admite el uso de palabras clave personalizadas. Va a utilizar este cliente para hablar con el bot y recibir una respuesta de voz.

Antes de continuar, asegúrese de que el micrófono y los altavoces estén habilitados y en perfecto funcionamiento.

1. Vaya al repositorio de GitHub para buscar el [cliente de Direct Line Speech](#).
2. Siga las instrucciones proporcionadas para clonar el repositorio, compilar el proyecto, configurar el cliente e iniciararlo.
3. Haga clic en **Reconnect** (Volver a conectar) y asegúrese de que ve el mensaje **Press the mic button, or type to start talking to your bot** (Presione el botón MIC o escriba para empezar a hablar con el bot).
4. Vamos a probarlo. Haga clic en el botón de micrófono y diga algunas palabras en inglés. El texto reconocido

aparecerá mientras habla. Cuando haya terminado de hablar, el bot responderá en su propia voz, diciendo "echo" (eco) seguido de las palabras reconocidas.

5. También puede utilizar el texto para comunicarse con el bot. Simplemente escriba el texto en la barra inferior.

### Solución de errores en el cliente de Direct Line Speech

Si recibe un mensaje de error en la ventana principal de la aplicación, use esta tabla para identificarlo y solucionarlo:

ERROR	¿QUÉ DEBERÍA HACER?
Error AuthenticationFailure: WebSocket Upgrade failed with an authentication error (401). Check for correct subscription key (or authorization token) and region name [Error AuthenticationFailure: No se pudo actualizar el socket web con un error de autenticación (401)]	En la página Configuración de la aplicación, asegúrese de que ha especificado correctamente la clave de suscripción de voz y su región. Asegúrese de que la clave de voz y la región de la clave se escribieron correctamente.
Error ConnectionFailure: Connection was closed by the remote host. Código de error: 1011. Detalles del error: We could not connect to the bot before sending a message (Error ConnectionFailure: El host remoto cerró la conexión. Código de error: 1011. Detalles del error: No se pudo conectar al bot antes de enviar un mensaje)	Asegúrese de <a href="#">activar la casilla "Enable Streaming Endpoint"</a> (Habilitar el punto de conexión de streaming) o de <a href="#">alternar los sockets web</a> en Activado. Asegúrese de que la instancia de Azure App Service se ejecuta. Si es así, pruebe a reiniciar la instancia de App Service.
Error ConnectionFailure: Connection was closed by the remote host. Código de error: 1011. Detalles del error: Response status code does not indicate success: 500 (InternalServerError) [Error ConnectionFailure: El host remoto cerró la conexión. Código de error: 1011. Detalles del error: El código de estado de respuesta no indica una operación correcta]	El bot especificó una voz neuronal en el campo <a href="#">Speak</a> (Hablar) de salida, pero la región de Azure asociada a la clave de suscripción de voz no es compatible con las voces neuronales. Consulte <a href="#">Voces estándares y neuronales</a> .
Error ConnectionFailure: Connection was closed by the remote host. Código de error: 1000. Detalles del error: Exceeded maximum web socket connection idle duration(> 300000 ms) [Error ConnectionFailure: El host remoto cerró la conexión. Código de error: 1000. Detalles del error: Se superó la duración máxima de inactividad de la conexión de socket web (> 300000 MS)]	Este es un error esperado cuando una conexión al canal se deja abierta y está inactiva durante más de cinco minutos.

Si el problema no se soluciona en la tabla, consulte [Asistentes de voz: Frequently asked questions](#) (inicio de sesión único de conexión directa de Azure Active Directory: preguntas más frecuentes).

### Ver actividades de bot

Cada bot envía y recibe mensajes de **actividad**. En la ventana **Activity Log** (Registro de actividad) del cliente de Direct line Speech, verá los registros con marca de tiempo con cada actividad en la que el cliente ha recibido del bot. También puede ver las actividades que el cliente ha enviado al bot mediante el método

[DialogServiceConnector.SendActivityAsync](#). Al seleccionar un elemento de registro, se mostrarán los detalles de la actividad asociada como JSON.

Este es un ejemplo de JSON de una actividad que el cliente ha recibido:

```
{
    "attachments": [],
    "channelData": {
        "conversationalAiData": {
            "requestInfo": {
                "interactionId": "8d5cb416-73c3-476b-95fd-9358cbfaebfa",
                "version": "0.2"
            }
        }
    },
    "channelId": "directlinespeech",
    "conversation": {
        "id": "129ebffe-772b-47f0-9812-7c5bfd4aca79",
        "isGroup": false
    },
    "entities": [],
    "from": {
        "id": "SpeechEchoBotTutorial-BotRegistration"
    },
    "id": "89841b4d-46ce-42de-9960-4fe4070c70cc",
    "inputHint": "acceptingInput",
    "recipient": {
        "id": "129ebffe-772b-47f0-9812-7c5bfd4aca79|0000"
    },
    "replyToId": "67c823b4-4c7a-4828-9d6e-0b84fd052869",
    "serviceUrl": "urn:botframework:websocket:directlinespeech",
    "speak": "<speak version='1.0' xmlns='https://www.w3.org/2001/10/synthesis' xml:lang='en-US'><voice name='Microsoft Server Speech Text to Speech Voice (en-US, JessaRUS)'>Echo: Hello and welcome.</voice></speak>",
    "text": "Echo: Hello and welcome.",
    "timestamp": "2019-07-19T20:03:51.1939097Z",
    "type": "message"
}
}
```

Para obtener más información sobre lo que se devuelve en la salida JSON, consulte los [campos de la actividad](#). Para este tutorial, puede centrarse en los campos **Text** (Texto) y **Speak** (Hablar).

### Visualización del código fuente de cliente para las llamadas al SDK de voz

El cliente de Direct Line Speech usa el paquete NuGet [Microsoft.CognitiveServices Account.Speech](#), que contiene el SDK de voz. Un buen lugar para empezar a revisar el código de ejemplo es el método `InitSpeechConnector()` en el archivo [DLSpeechClient\MainWindow.xaml.cs](#), que crea estos dos objetos del SDK de voz:

- `DialogServiceConfig`: para opciones de configuración (por ejemplo, clave de suscripción de voz, región de la clave)
- `DialogServiceConnector`: para administrar la conexión del canal y los eventos de la suscripción de cliente para controlar las respuestas de voz y bot reconocidas.

## Agregar la activación de la palabra clave personalizada

El SDK de Voz admite la activación de palabras clave personalizadas. De forma similar a "Hola Cortana" del ayudante de Microsoft, puede escribir una aplicación que escuche continuamente la palabra clave que prefiera. Tenga en cuenta que una palabra clave puede ser una sola palabra o una frase con varias palabras.

#### NOTE

El término *palabra clave* se suele usar indistintamente con el término *palabra de reactivación*, y verá que ambos se usan en la documentación de Microsoft.

La detección de la palabra clave se realiza en la aplicación cliente. Si se usa una palabra clave, el audio solo se transmite al canal de Direct Line Speech si se detecta dicha palabra clave. El canal Direct Line Speech incluye un componente denominado *comprobación de palabras clave (KVV)*, que realiza un procesamiento más complejo en la nube para comprobar que la palabra clave que ha elegido está al principio de la secuencia de audio. Si la comprobación de palabras clave se realiza correctamente, el canal se comunicará con el bot.

Siga estos pasos para crear un modelo de palabra clave, configurar el cliente de Direct Line Speech para usar este modelo y, por último, probarlo con el bot.

1. Siga estas instrucciones para [crear una palabra clave personalizada mediante el servicio Voz](#).
2. Descomprima el archivo de modelo que descargó en el paso anterior. Debe tener el nombre de la palabra clave. Está buscando un archivo denominado `kws.table`.
3. En el cliente de Direct Line Speech, busque el menú de **configuración** (busque el icono de engranaje en la parte superior derecha). Busque la **ruta de acceso del archivo de modelo** y escriba el nombre de la ruta de acceso completa del archivo `kws.table` del paso 2.
4. Asegúrese de activar la casilla denominada **Enabled** (Habilitada). Debería ver este mensaje junto a la casilla: "Will listen for the keyword upon next connection" (Escuchará la palabra clave después de la siguiente conexión). Si ha proporcionado un archivo incorrecto o una ruta de acceso no válida, debería ver un mensaje de error.
5. Escriba la **clave de suscripción** de voz, la **región de clave de suscripción** y, a continuación, haga clic en **Aceptar** para cerrar el menú de **configuración**.
6. Haga clic en **Volver a conectar**. Verá un mensaje que indica lo siguiente: "New conversation started - type, press the microphone button, or say the keyword" (Nueva conversación iniciada: escriba, presione el botón de micrófono o la palabra clave). La aplicación ahora escucha continuamente.
7. Diga cualquier frase que empiece con la palabra clave. Por ejemplo: "**[su palabra clave]**, what time is it?". No es necesario hacer una pausa después de pronunciar la palabra clave. Cuando termine, sucederán dos cosas:
  - Verá una transcripción de lo que ha hablado.
  - Poco después, oirá la respuesta del bot.
8. Continúe experimentando con los tres tipos de entradas que admite el bot:
  - Escribir texto en la barra inferior
  - Presionar el icono del micrófono y hablar
  - Decir una frase que empiece con la palabra clave

### Ver el código fuente que habilita la palabra clave

En el código fuente del cliente Direct Line Speech, eche un vistazo a estos archivos para revisar el código que se usa para habilitar la detección de palabras clave:

1. `DLSpeechClient\Models.cs` incluye una llamada al método `KeywordRecognitionModel.fromFile()` del SDK de voz, que se usa para crear una instancia del modelo a partir de un archivo local en el disco.
2. `DLSpeechClient\MainWindow.xaml.cs` incluye una llamada al método `DialogServiceConnector.StartKeywordRecognitionAsync()` del SDK de Voz, que activa la detección continua de la palabra clave.

## (Opcional) Cambio del idioma y la voz del bot

El bot que creó escuchará y responderá en inglés, y la voz predeterminada para la conversión de texto a voz será en inglés estadounidense. Sin embargo, no está limitado a usar el idioma inglés ni una voz predeterminada. En esta sección, aprenderá a cambiar el idioma en el que el bot escuchará y responderá. También aprenderá a seleccionar una voz diferente para ese idioma.

### Cambio del idioma

Puede elegir uno de los idiomas que se mencionan en la tabla [voz a texto](#). En el ejemplo siguiente, se cambiará el idioma a alemán.

1. Abra la aplicación cliente de Direct Line Speech, haga clic en el botón de configuración (ícono de engranaje en la parte superior derecha) y escriba `de-de` en el campo Idioma (este es el valor de configuración regional que se menciona en la tabla [voz a texto](#)). Esto establece el idioma hablado para que se reconozca y se sustituya el valor predeterminado `en-us`. Esta acción también indica al canal de Direct Line Speech que use una voz alemana predeterminada para la respuesta del bot.
2. Cierre la página Configuración y haga clic en el botón Volver a conectar para establecer una nueva conexión con el bot de eco.
3. Haga clic en el botón de micrófono y diga una frase en alemán. Verá el texto reconocido y el bot de eco responderá con la voz alemana predeterminada.

### Cambio de la voz de bot predeterminada

Puede seleccionar la voz de conversión de texto a voz y controlar la pronunciación si el bot especifica la respuesta en forma de [lenguaje de marcado de síntesis de voz](#) (SSML) en lugar de texto simple. El bot de eco no usa SSML, pero se puede modificar fácilmente el código para que lo use. En el ejemplo siguiente, se agrega SSML a la respuesta del bot de eco. De este modo, se utilizará la voz alemana de Stefan Apollo (una voz de hombre) en lugar de la voz de mujer predeterminada. Consulte la lista de [voices estándar](#) y [voices neuronales](#) compatibles con su idioma.

1. Comencemos por abrir `samples\csharp_dotnetcore\02.echo-bot\echo-bot.cs`.
2. Busque estas dos líneas:

```
var replyText = $"Echo: {turnContext.Activity.Text}";
await turnContext.SendActivityAsync(MessageFactory.Text(replyText, replyText), cancellationToken);
```

3. Reemplácelas por:

```
var replyText = $"Echo: {turnContext.Activity.Text}";
var replySpeak = @"<speak version='1.0' xmlns='https://www.w3.org/2001/10/synthesis' xml:lang='de-DE'>
    <voice name='Microsoft Server Speech Text to Speech Voice (de-DE, Stefan, Apollo)'>" +
    $"{replyText}" + "</voice></speak>";
await turnContext.SendActivityAsync(MessageFactory.Text(replyText, replySpeak), cancellationToken);
```

4. Compile la solución en Visual Studio y corrija los errores de compilación.

El segundo argumento del método "MessageFactory.Text" establece el [campo Texto hablado de la actividad](#) en la respuesta del bot. Con el cambio anterior, se realizó el cambio de texto simple a SSML para especificar una voz alemana no predeterminada.

### Reimplementación del bot

Ahora que ha realizado el cambio necesario en el bot, el paso siguiente consiste en volver a publicarlo en la instancia de Azure App Service y probarlo:

1. En la ventana Explorador de soluciones, haga clic con el botón derecho en el proyecto **EchoBot** y seleccione **Publicar**.
2. La configuración de implementación anterior ya se ha cargado como valor predeterminado. Simplemente haga clic en **Publicar** junto a **EchoBot20190805125647 - Web Deploy**.
3. El mensaje **Publicación correcta** aparecerá en la ventana de salida de Visual Studio y se iniciará una página web con el mensaje "Your bot is ready!" (El bot está listo).
4. Abra la aplicación cliente de Direct Line Speech, haga clic en el botón de configuración (ícono de engranaje en la parte superior derecha) y asegúrese de que aún se encuentra el valor `de-de` en el campo Idioma.
5. Siga las instrucciones de [Creación del cliente de Direct Line Speech](#) para volver a conectar con el bot recién

implementado, hablar en el nuevo idioma y escuchar la respuesta del bot en ese idioma con la nueva voz.

## Limpieza de recursos

Si no va a seguir usando el bot eco que ha implementado en este tutorial, puede quitarlo y todos los recursos de Azure asociados simplemente eliminando el grupo de recursos de Azure **SpeechEchoBotTutorial-ResourceGroup**.

1. En [Azure Portal](#), seleccione **Grupos de recursos** en el panel de navegación izquierdo.
2. Busque el grupo de recursos denominado: **SpeechEchoBotTutorial-ResourceGroup**. Haga clic en los tres puntos (...).
3. Seleccione **Eliminar grupo de recursos**.

## Pasos siguientes

[Creación de la propia aplicación cliente con el SDK de voz](#)

## Consulte también

- Implementación en una [región de Azure próxima](#) para ver la mejora del tiempo de respuesta de bot
- Implementación en una [región de Azure que admite voces TTS neuronales de alta calidad](#)
- Precios asociados al canal Direct Line Speech:
  - [Precios de Bot Service](#)
  - [Servicio Voz](#)
- Creación e implementación del propio bot habilitado para voz:
  - Creación de un [bot de Bot-Framework](#). Registro con el [canal Direct Line Speech](#) y [personalización del bot para voz](#)
  - Exploración de las [soluciones de Bot-Framework](#) existentes: Compilación de un [asistente virtual](#) y su [ampliación a Direct Line Speech](#)

# Preguntas más frecuentes sobre los asistentes de voz

13/01/2020 • 6 minutes to read • [Edit Online](#)

Si no encuentra respuestas a sus preguntas en este documento, [consulte otras opciones de soporte técnico](#).

## General

### P: ¿Qué es un asistente de voz?

**R:** Como Cortana, un asistente de voz es una solución que escucha las expresiones habladas de un usuario, analiza el contenido de esas expresiones en busca de significado, realiza una o varias acciones en respuesta a la intención de la expresión y, a continuación, proporciona una respuesta al usuario que a menudo incluye un componente hablado. Se trata de una experiencia de "conversación" para interactuar con un sistema. Los autores del asistente de voz crean una aplicación en el dispositivo mediante `DialogServiceConnector` en el SDK de Voz para comunicarse con un asistente creado mediante [comandos personalizados \(versión preliminar\)](#) o el canal de [Direct Line Speech](#) de Bot Framework. Estos asistentes pueden usar palabras clave personalizadas, conversaciones personalizadas y voz personalizada para proporcionar una experiencia adaptada a su marca o producto.

### P: ¿Debo usar comandos personalizados (versión preliminar) o Direct Line Speech? ¿Cuál es la diferencia?

**R:** Los [comandos personalizados \(versión preliminar\)](#) son un conjunto de herramientas de menor complejidad para crear y hospedar fácilmente un asistente adecuado para escenarios de finalización de tareas. [Direct Line Speech](#) proporciona capacidades más enriquecidas y sofisticadas para permitir escenarios de conversación sólidos. Para obtener más información, consulte la [comparación de soluciones para asistentes](#).

### P: ¿Cómo empiezo?

**R:** La mejor manera de empezar es la creación de una aplicación de comandos personalizados (versión preliminar) o un bot de Bot Framework.

- [Creación de una aplicación de comandos personalizados \(versión preliminar\)](#)
- [Creación de un bot de Bot Framework básico](#)
- [Conexión de un bot al canal Direct Line Speech](#)

## Depuración

### P: ¿Dónde está mi secreto de canal?

**R:** Si ha usado la versión preliminar de Direct Line Speech o está leyendo la documentación relacionada, puede encontrar una clave secreta en la página de registro del canal Direct Line Speech. `FromBotSecret` de Factory Method `DialogServiceConfig` v1.7 en el SDK de Voz también espera este valor.

La versión más reciente de Direct Line Speech simplifica el proceso de comunicación con el bot desde un dispositivo. En la página de registro del canal, la lista desplegable de la parte superior asocia el registro del canal Direct Line Speech con un recurso de voz. Una vez asociado, el SDK de Voz v 1.8 incluye Factory Method `BotFrameworkConfig::FromSubscription` que configurará un `DialogServiceConnector` para que se comunique con el bot que ha asociado a la suscripción.

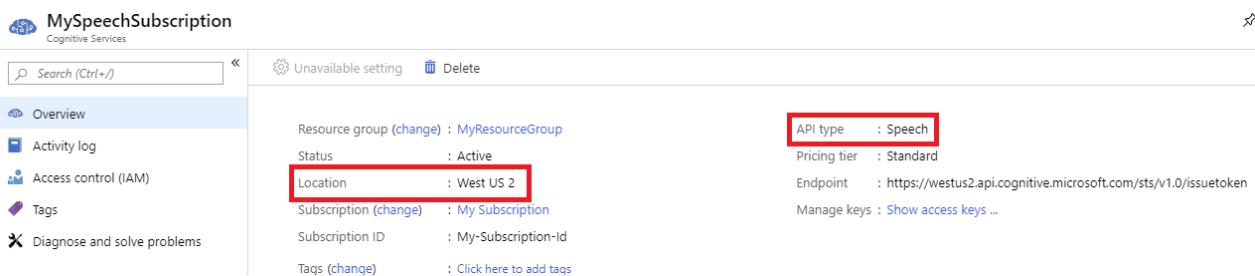
Si todavía está migrando la aplicación cliente de v1.7 a v1.8, `DialogServiceConfig::FromBotSecret` puede seguir trabajando con un valor no vacío y no nulo para su parámetro de secreto de canal, por ejemplo, el secreto anterior que utilizó. Simplemente se omitirá cuando se use una suscripción de voz asociada a un registro de canal más

reciente. Tenga en cuenta que el valor *debe* ser no nulo y no estar vacío, ya que se comprueban en el dispositivo antes de que la asociación del lado de servicio sea apropiada.

Para obtener una guía más detallada, consulte la [sección del tutorial](#) que le guía por el registro del canal.

**P: Se genera un error 401 durante la conexión y nada funciona. Sé que mi clave de suscripción de voz es válida. ¿Qué está ocurriendo?**

**R:** Cuando administre la suscripción en Azure Portal, asegúrese de que usa el recurso **Voz** (`Microsoft.CognitiveServicesSpeechServices`, "Voz") y *no* el recurso **Cognitive Services** (`Microsoft.CognitiveServicesAllInOne`, "Todos los servicios de Cognitive Services"). Consulte también [regiones admitidas del servicio de voz para los asistentes de voz](#).



MySpeechSubscription  
Cognitive Services

Search (Ctrl+ /) < Unavailable setting Delete

Overview Activity log Access control (IAM) Tags Diagnose and solve problems

Resource group (change) : MyResourceGroup	API type : Speech
Status : Active	Pricing tier : Standard
Location : West US 2	Endpoint : <a href="https://westus2.api.cognitive.microsoft.com/sts/v1.0/issuetoken">https://westus2.api.cognitive.microsoft.com/sts/v1.0/issuetoken</a>
Subscription (change) : My Subscription	Manage keys : <a href="#">Show access keys ...</a>
Subscription ID : My-Subscription-Id	
Tags (change) : Click here to add tags	

**P: Obtengo el texto de reconocimiento de `DialogServiceConnector`, pero veo un error "1011" y no recibo nada de mi bot. ¿Por qué?**

**R:** Este error indica un problema de comunicación entre su asistente y el servicio del asistente de voz.

- En el caso de los comandos personalizados (versión preliminar), asegúrese de que la aplicación de comandos personalizados (versión preliminar) esté publicada.
- En el caso de Direct Line Speech, asegúrese de que ha [conectado su bot al canal Direct Line Speech](#), [ha agregado compatibilidad con el protocolo de transmisión](#) al bot (con la compatibilidad relacionada con socket web) y, luego, compruebe que su bot responde a las solicitudes entrantes del canal.

**P: Este código todavía no funciona o recibo un error diferente cuando uso `DialogServiceConnector`. ¿Qué debo hacer?**

**R:** El registro basado en archivos proporciona bastantes más detalles y puede ayudarle a acelerar las solicitudes de soporte técnico. Para habilitar esta funcionalidad, consulte el [uso del registro de archivos](#).

## Pasos siguientes

- [Solución de problemas](#)
- [Notas de la versión](#)

# ¿Qué es Custom Speech?

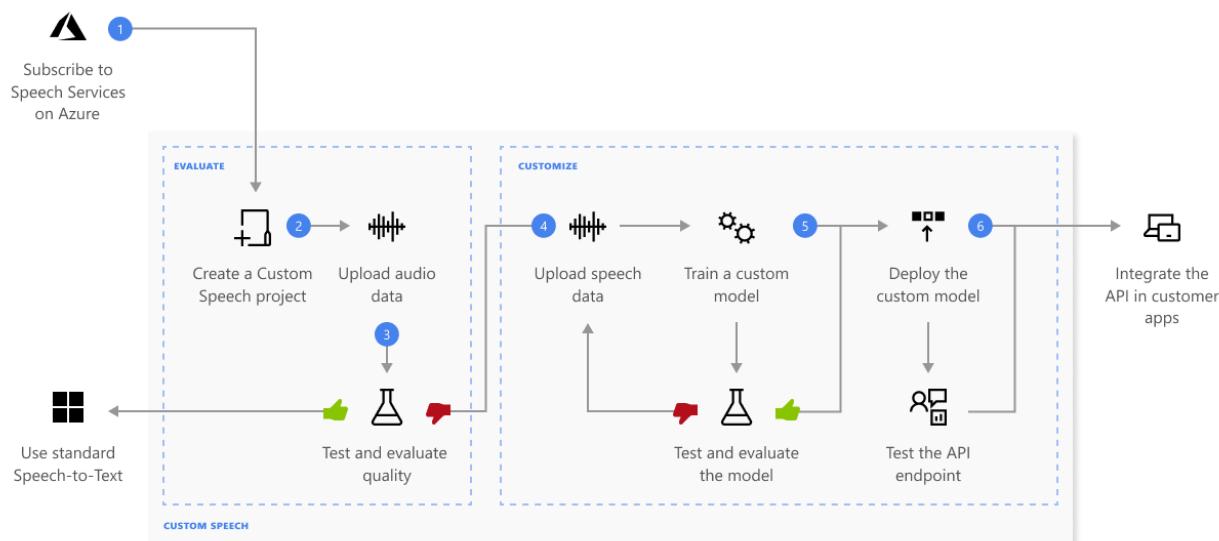
13/01/2020 • 6 minutes to read • [Edit Online](#)

Custom Speech es un conjunto de herramientas en línea que permiten evaluar y mejorar la precisión de la conversión de voz a texto de Microsoft para las aplicaciones, herramientas y productos. Para comenzar solo se necesita una serie de archivos de audio de prueba. Siga los vínculos que se incluyen a continuación para empezar a crear una experiencia personalizada de conversión de voz a texto.

## ¿Qué incluye Custom Speech?

Para utilizar Custom Speech, necesitará una cuenta de Azure y una suscripción al servicio de voz. Cuando ya tenga la cuenta, podrá preparar los datos, entrenar y probar sus modelos, inspeccionar la calidad del reconocimiento, evaluar la precisión y, en última instancia, implementar y utilizar el modelo de conversión de voz a texto personalizado.

Este diagrama resalta las partes que componen el [portal de Custom Speech](#). Use los siguientes vínculos para obtener más información sobre cada paso.



1. [Suscríbase y cree un proyecto](#): cree una cuenta de Azure y suscríbase al servicio de voz. Esta suscripción unificada proporciona acceso a la conversión de voz a texto, la conversión de texto a voz, la traducción de voz y el [portal de Custom Speech](#). A continuación, mediante la suscripción al servicio de voz, cree su primer proyecto de Custom Speech.
2. [Cargue datos de prueba](#): cargue datos de prueba (archivos de audio) para evaluar la oferta de conversión de voz a texto para sus aplicaciones, herramientas y productos.
3. [Inspeccione la calidad del reconocimiento](#): use el [portal de Custom Speech](#) para reproducir el audio cargado e inspeccionar la calidad del reconocimiento de voz de los datos de prueba. Para conocer las medidas cuantitativas, consulte [Inspección de los datos](#).
4. [Evalúe la precisión](#): evalúe la precisión del modelo de conversión de voz a texto. El [portal de Custom Speech](#) proporcionará una *tasa de errores de palabras*, que puede utilizarse para determinar si se necesita más entrenamiento. Si está satisfecho con la precisión, puede usar directamente las API del servicio de voz. Si desea mejorar la precisión en una media relativa del 5 al 20 %, use la pestaña **Entrenamiento** del portal para cargar datos de entrenamiento adicionales, como transcripciones con etiqueta humana y texto relacionado.

5. **Entrene el modelo:** mejore la precisión del modelo de conversión de voz a texto uniendo transcripciones escritas (de 10 a 1000 horas) y texto relacionado (< 200 MB) a datos de prueba de audio. Estos datos ayudan a entrenar el modelo de conversión de voz a texto. Despues del entrenamiento, vuelva a probar; si le satisface el resultado, ya puede implementar el modelo.
6. **Implemente el modelo:** cree un punto de conexión personalizado para el modelo de conversión de voz a texto y úselo en sus aplicaciones, herramientas o productos.

## Configuración de la cuenta de Azure

Para poder usar el [portal de Custom Speech](#) y crear un modelo personalizado, se necesita una suscripción al servicio de voz. Siga estas instrucciones para crear una suscripción estándar al servicio de voz: [Creación de una suscripción a Voz](#).

### NOTE

Asegúrese de crear suscripciones estándar (S0); no se admiten las suscripciones de prueba gratuita (F0).

Después de crear la cuenta de Azure y la suscripción al servicio de voz, deberá iniciar sesión en el [portal de Custom Speech](#) y conectarse a su suscripción.

1. Obtenga la clave de la suscripción del servicio de voz en Azure Portal.
2. Inicie sesión en el [portal de Custom Speech](#).
3. Seleccione la suscripción que necesita para trabajar y crear un proyecto de voz.
4. Si desea modificar la suscripción, utilice el icono **engranaje** situado en el panel de navegación superior.

## Creación de un proyecto

El contenido, como datos, modelos, pruebas y puntos de conexión, se organiza en **proyectos** en el [portal de Custom Speech](#). Cada proyecto es específico de un dominio y un país o idioma. Por ejemplo, puede crear un proyecto para centros de llamadas que usan el inglés en Estados Unidos.

Para crear su primer proyecto, seleccione **Speech-to-text/Custom speech** (Conversión de voz a texto/Conversión de voz personalizada) y, a continuación, haga clic en **New project** (Nuevo proyecto). Siga las instrucciones del asistente para crear el proyecto. Despues de crear el proyecto, verá cuatro pestañas: **Datos**, **Pruebas**, **Entrenamiento** e **Implementación**. Use los vínculos incluidos en [Pasos siguientes](#) para aprender a usar cada pestaña.

## Pasos siguientes

- [Preparación y prueba de los datos](#)
- [Inspección de los datos](#)
- [Evaluación de los datos](#)
- [Entrenamiento del modelo](#)
- [Implementación del modelo](#)

# Preparación de los datos para Custom Speech

15/01/2020 • 15 minutes to read • [Edit Online](#)

Al probar la precisión del reconocimiento de voz de Microsoft o entrenar sus modelos personalizados, necesitará datos de texto y de audio. En esta página, veremos los tipos de datos, cómo se usan y administran.

## Tipos de datos

En esta tabla se enumeran los tipos de datos aceptados, cuándo se debe utilizar cada tipo de datos y la cantidad recomendada. No todos los tipos de datos son necesarios para crear un modelo. Los requisitos de datos variarán dependiendo de si crea una prueba o entrena un modelo.

TIPO DE DATOS	SE USA PARA PRUEBAS	CANTIDAD RECOMENDADA	SE UTILIZA PARA EL ENTRENAMIENTO	CANTIDAD RECOMENDADA
Audio	Sí Se utiliza para la inspección visual	Más de cinco archivos de audio	No	N/a
Transcripciones de audio con etiqueta humana	Sí Se utiliza para evaluar la precisión	De 0,5 a 5 horas de audio	Sí	De 1 a 1000 horas de audio
Texto relacionado	No	N/a	Sí	De 1 a 200 MB de texto relacionado

Los archivos deben agruparse por tipo en un conjunto de datos y cargarse como archivo zip. Cada conjunto de datos solo puede contener un tipo de datos.

### TIP

Para empezar a trabajar rápidamente, considere la posibilidad de usar datos de ejemplo. Consulte este repositorio de GitHub para obtener [datos de Custom Speech de ejemplo](#)

## Carga de datos

Para cargar los datos, navegue al portal de [Custom Speech](#). En el portal, haga clic en **Cargar datos** para iniciar el asistente y crear el primer conjunto de datos. Se le pedirá que seleccione un tipo de datos de voz para el conjunto de datos, antes de permitirle cargar los datos.

Select speech data type

X

**Audio only**

Standalone speech audio data used for testing purposes. [See formatting requirements](#)

**Related text**

Text related to the domain of your intended speech model, such as transcripts, pronunciation guides, and documents, that can be used for training purposes. [Learn more](#)

**Audio + human-labeled transcript**

Speech audio data paired with accurate, transcribed text for each spoken utterance used for both training and testing purposes. [See formatting requirements](#)

Need help working with speech data? [Download our sample datasets](#)

[Next](#) [Back](#)

Cada conjunto de datos que cargue debe cumplir los requisitos del tipo de datos elegido. Los datos deben tener el formato correcto para poder cargarse. Los datos con el formato correcto garantizan que el servicio Custom Speech procesará los datos con precisión. Los requisitos se muestran en las secciones siguientes.

Una vez cargado el conjunto de datos, tiene algunas opciones:

- Puede dirigirse a la pestaña **Testing** (Pruebas) e inspeccionar visualmente datos de transcripción solo de audio o de audio y con etiqueta humana.
- Puede dirigirse a la pestaña **Training** (Entrenamiento) y utilizar datos de transcripción de audio y humana o datos de texto relacionados para entrenar un modelo personalizado.

## Datos de audio para pruebas

Los datos de audio son óptimos para probar la precisión del modelo de línea de base de Microsoft de voz a texto o de un modelo personalizado. Tenga en cuenta que los datos de audio se utilizan para inspeccionar la precisión de la voz con respecto al rendimiento de un modelo específico. Si busca cuantificar la precisión de un modelo, utilice los [datos de transcripción de audio y con etiqueta humana](#).

Utilice esta tabla para asegurarse de que los archivos de audio están formateados correctamente para su uso con Custom Speech:

PROPIEDAD	VALUE
Formato de archivo	RIFF (WAV)
Velocidad de muestreo	8000 o 16 000 Hz
Canales	1 (mono)
Longitud máxima por audio	2 horas
Formato de ejemplo	PCM, 16 bits

PROPIEDAD	VALUE
Formato de archivo	.zip
Tamaño de archivo máximo	2 GB

#### TIP

Al cargar los datos del entrenamiento y de las pruebas, el archivo ZIP no puede superar los 2 GB. Si requiere más datos para el entrenamiento, divídalos en varios archivos ZIP y cárguelos por separado. Más adelante, puede optar por entrenar en *varios* conjuntos de datos. Aunque solo puede realizar pruebas desde un *único* conjunto de datos.

Use [SoX](#) para comprobar las propiedades de audio o convertir el audio existente en los formatos adecuados. A continuación se muestran algunos ejemplos de cómo se puede realizar cada una de estas actividades mediante la línea de comandos de SoX:

ACTIVIDAD	DESCRIPCIÓN	COMANDO DE SOX
Comprobar el formato de audio	Use este comando para comprobar el formato de archivo de audio.	<code>sox --i &lt;filename&gt;</code>
Convertir formato de audio	Use este comando para convertir el archivo de audio en un canal único, de 16 bits y 16 KHz.	<code>sox &lt;input&gt; -b 16 -e signed-integer -c 1 -r 16k -t wav &lt;output&gt;.wav</code>

## Datos de transcripción de audio y con etiqueta humana para pruebas y entrenamiento

Para medir la precisión de la conversión de voz a texto de Microsoft al procesar sus archivos de audio, debe proporcionar transcripciones con etiqueta humana (palabra por palabra) para su comparación. Aunque la transcripción con etiqueta humana a menudo requiere mucho tiempo, es necesario evaluar la precisión y entrenar al modelo para los casos de uso. Tenga en cuenta que las mejoras en el reconocimiento solo serán tan buenas como los datos proporcionados. Por ese motivo, es importante que solo se carguen las transcripciones de alta calidad.

PROPIEDAD	VALUE
Formato de archivo	RIFF (WAV)
Velocidad de muestreo	8000 o 16 000 Hz
Canales	1 (mono)
Longitud máxima por audio	2 horas (pruebas)/60 s (entrenamiento)
Formato de ejemplo	PCM, 16 bits
Formato de archivo	.zip
Tamaño máximo de archivo zip	2 GB

#### NOTE

Al cargar los datos del entrenamiento y de las pruebas, el archivo ZIP no puede superar los 2 GB. Solo puede realizar pruebas desde un **único** conjunto de datos; asegúrese de mantenerlo dentro del tamaño de archivo adecuado.

Para abordar problemas como la eliminación o sustitución de palabras, se necesita una cantidad significativa de datos para mejorar el reconocimiento. Por lo general, se recomienda proporcionar transcripciones palabra por palabra para aproximadamente de 10 a 1000 horas de audio. Las transcripciones para todos los archivos WAV deben incluirse en un único archivo de texto sin formato. Cada línea del archivo de transcripción debe contener el nombre de uno de los archivos de audio, seguido de la transcripción correspondiente. El nombre de archivo y la transcripción deben estar separados por un carácter de tabulación (\t).

Por ejemplo:

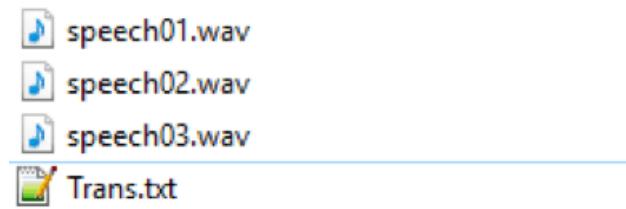
```
speech01.wav speech recognition is awesome
speech02.wav the quick brown fox jumped all over the place
speech03.wav the lazy dog was not amused
```

#### IMPORTANT

La transcripción debería estar codificada como una marca BOM UTF-8.

Las transcripciones son texto normalizado para que el sistema pueda procesarlas. Aunque hay algunas normalizaciones importantes que debe hacer el usuario antes de cargar los datos en Speech Studio. Para conocer el idioma que debe usar al preparar las transcripciones, consulte [How to create a human-labeled transcription](#) (Creación de una transcripción con etiqueta humana).

Cuando haya reunido los archivos de audio y las transcripciones correspondientes, debe empaquetarlos como un solo archivo .zip antes de cargarlos en el [portal de Custom Speech](#). A continuación se muestra un conjunto de datos de ejemplo con tres archivos de audio y un archivo de transcripción con etiqueta humana:



## Datos de texto relacionados para el entrenamiento

Los nombres de producto o las características que son únicas deben incluir datos de texto relacionados para el entrenamiento. El texto relacionado ayuda a garantizar el reconocimiento correcto. Se pueden proporcionar dos tipos de datos de texto relacionados para mejorar el reconocimiento:

TIPO DE DATOS	MEJORA DEL RECONOCIMIENTO POR ESTOS DATOS
Frases (expresiones)	Mejore la precisión al reconocer nombres de productos o vocabulario específico del sector dentro del contexto de una frase.
Pronunciaciones	Mejore la pronunciación de términos poco comunes, acrónimos u otras palabras de pronunciación indefinida.

Las frases se pueden proporcionar como un solo archivo de texto o como varios archivos de texto. Para mejorar la precisión, use datos de texto que estén más cerca de las expresiones orales esperadas. Las pronunciaciones deben proporcionarse como un único archivo de texto. Todo se puede empaquetar como un único archivo zip y cargarse en el [portal de Custom Speech](#).

### Directrices para crear un archivo de frases

Para crear un modelo personalizado con frases, tendrá que proporcionar una lista de expresiones de ejemplo. Las expresiones *no* tienen que ser frases completas o gramaticalmente correctas, pero deben reflejar con precisión la entrada oral que se espera en producción. Si quiere que ciertos términos tengan mayor peso, puede agregar varias frases que incluyan estos términos específicos.

Como guía general, la adaptación de modelos es más eficaz cuando el texto de entrenamiento es lo más parecido posible al texto real esperado en producción. La jerga y las frases específicas del dominio que va a mejorar deben incluirse en el texto de entrenamiento. Cuando sea posible, intente que una frase o una palabra clave se controle en una línea independiente. Para las palabras clave y frases que son importantes para usted (por ejemplo, nombres de producto), puede copiarlas varias veces. Pero tenga en cuenta que no debe copiar demasiado; podría afectar a la tasa de reconocimiento general.

Utilice esta tabla para asegurarse de que el archivo de datos relacionado con las expresiones esté formateado correctamente:

PROPIEDAD	VALUE
Codificación de texto	BOM UTF-8
Número de expresiones por línea	1
Tamaño de archivo máximo	200 MB

Además, querrá tener en cuenta las siguientes restricciones:

- Evite repetir los caracteres más de cuatro veces. Por ejemplo: "aaaa" o "uuuu".
- No utilice caracteres especiales ni caracteres UTF-8 por encima de `U+00A1`.
- Se rechazarán los identificadores URI.

### Directrices para crear un archivo de pronunciación

Si hay términos poco comunes sin pronunciaciones estándar que los usuarios van a encontrar o utilizar, puede proporcionar un archivo de pronunciación personalizado para mejorar el reconocimiento.

#### IMPORTANT

No se recomienda utilizar archivos de pronunciación personalizados para modificar la pronunciación de palabras comunes.

Esto incluye ejemplos de una expresión hablada y una pronunciación personalizada para cada uno:

FORMULARIO RECONOCIDO O MOSTRADO	FORMATO HABLADO
3CPO	CI-TRI-PI-OU
CNTK	CI EN TI KEI
IEEE	AI TRIPLE I

La forma hablada es la secuencia fonética deletreada. Puede estar compuesta de letras, palabras, sílabas o una

combinación de las tres.

La pronunciación personalizada está disponible en inglés ( `en-US` ) y en alemán ( `de-DE` ). En esta tabla se muestran los caracteres admitidos por idioma:

IDIOMA	CONFIGURACIÓN REGIONAL	CHARACTERS
Inglés	<code>en-US</code>	a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, p, q, r, s, t, u, v, w, x, y, z
Alemán	<code>de-DE</code>	ä, ö, ü, a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, p, q, r, s, t, u, v, w, x, y, z

Utilice la tabla siguiente para asegurarse de que el formato del archivo de datos relacionado con las pronunciaciones sea correcto. Los archivos de pronunciación son pequeños y no deberían superar unos pocos KB.

PROPIEDAD	VALUE
Codificación de texto	BOM UTF-8 (también se admite ANSI con el inglés)
Número de pronunciaciones por línea	1
Tamaño de archivo máximo	1 MB (1 KB por cada nivel gratis)

## Pasos siguientes

- [Inspección de los datos](#)
- [Evaluación de los datos](#)
- [Entrenamiento del modelo](#)
- [Implementación del modelo](#)

# Inspección de los datos de Custom Speech

13/01/2020 • 4 minutes to read • [Edit Online](#)

## NOTE

En esta página se supone que ha leído [Preparación de los datos de Custom Speech](#) y que ha cargado un conjunto de datos para su inspección.

Custom Speech proporciona herramientas que le permiten inspeccionar visualmente la calidad del reconocimiento de un modelo mediante la comparación de los datos de audio con el resultado de reconocimiento correspondiente. Desde el [portal de Custom Speech](#), puede reproducir el audio cargado y determinar si el resultado de reconocimiento proporcionada es correcto. Esta herramienta le permite inspeccionar rápidamente la calidad del modelo de voz a texto de línea de base de Microsoft o de un modelo personalizado entrenado sin tener que transcribir los datos de audio.

En este documento, aprenderá a inspeccionar visualmente la calidad de un modelo con los datos de entrenamiento cargados previamente.

En esta página, aprenderá a inspeccionar visualmente la calidad del modelo de voz a texto de línea de base de Microsoft o de un modelo personalizado que haya entrenado. Usará los datos que cargó en la pestaña **Data** (Datos) para realizar pruebas.

## Creación de una prueba

Para crear una prueba, siga estas instrucciones:

1. Inicie sesión en el [portal de Custom Speech](#).
2. Vaya a **Speech-to-text > Custom Speech > Testing** (Conversión de voz a texto > Custom Speech > Pruebas).
3. Haga clic en **Add test** (Agregar prueba).
4. Seleccione **Inspect quality (Audio-only data)** (Inspeccionar la calidad [solo datos de audio]). Asigne un nombre y una descripción a la prueba y seleccione el conjunto de datos de audio.
5. Puede seleccionar hasta dos modelos para probar.
6. Haga clic en **Create** (Crear).

Después de que la prueba se ha creado correctamente, puede comparar los modelos en paralelo.

## NOTE

Al realizar pruebas, el sistema llevará a cabo una transcripción. Es importante tenerlo en cuenta, ya que el precio varía según la oferta de servicio y el nivel de suscripción. Consulte siempre en la página oficial Precios de Cognitive Services: Speech Services [los detalles más recientes](#).

## Comparaciones de los modelos en paralelo

Cuando el estado de la prueba sea *Succeeded* (Correcto), haga clic en el nombre del elemento de prueba para ver los detalles de la prueba. En esta página se muestran todas las expresiones del conjunto de datos y se indican los resultados del reconocimiento de los dos modelos al lado de la transcripción del conjunto de datos enviado.

Con el fin de inspeccionar la comparación en paralelo, puede alternar los distintos tipos de error, como inserción, eliminación y sustitución. Al escuchar el audio y comparar los resultados del reconocimiento de cada columna (que muestra la transcripción con la etiqueta humana y los resultados de los dos modelos de conversión de voz a texto), puede decidir qué modelo satisface sus necesidades y dónde hacen falta mejoras.

La inspección de las pruebas de calidad es útil para validar si la calidad de un punto de conexión de reconocimiento de voz es suficiente para una aplicación. Si necesita una medida objetiva de precisión, que necesite la transcripción del audio, siga las instrucciones que encontrará en [Evaluación de la precisión](#).

## Pasos siguientes

- [Evaluación de los datos](#)
- [Entrenamiento del modelo](#)
- [Implementación del modelo](#)

## Recursos adicionales

- [Preparación de los datos de prueba para Custom Speech](#)

# Evaluación de la precisión de la voz personalizada

13/01/2020 • 6 minutes to read • [Edit Online](#)

En este documento, aprenderá a medir cuantitativamente la calidad del modelo de conversión de texto a voz de Microsoft o su modelo personalizado. Para probar la voz se requieren datos de transcripción de con la etiqueta audio + humano, y se deben proporcionar entre 30 minutos y 5 horas de audio representativo.

## ¿Qué es Word Error Rate (WER)?

Es el estándar del sector para medir la precisión del modelo. WER cuenta el número de palabras incorrectas identificadas durante el reconocimiento y luego las divide entre el número total de palabras proporcionadas en la transcripción con la etiqueta humano. Por último, ese número se multiplica por 100 % para calcular la tasa WER.

$$WER = \frac{I + D + S}{N} * 100\%$$

Las palabras identificadas incorrectamente pertenecen a tres categorías:

- Inserción (I): palabras que se agregan incorrectamente en la transcripción de hipótesis.
- Eliminación (D): palabras que no se detectan en la transcripción de hipótesis.
- Sustitución (S): palabras que se sustituyeron entre la referencia y la hipótesis.

Este es un ejemplo:

Human-labeled Transcript: How **D**are you today John  
Speech Recognition Result: How you **I**a today Jones **T** **S**

## Resolución de errores y mejora de WER

Puede usar WER a partir de los resultados de reconocimiento automático para evaluar la calidad del modelo que usa con su aplicación, herramienta o producto. Una tasa WER de entre un 5 y un 10 % se considera buena calidad y está listo para usarse. Una tasa WER de 20 % es aceptable, pero quizás considere la posibilidad de entrenamiento adicional. Una tasa WER de 30 % o más señala una calidad deficiente y la necesidad de personalización y entrenamiento.

El modo en que se distribuyen los errores es importante. Si se encuentran muchos errores de eliminación, la causa suele ser la intensidad débil de señal de audio. Para resolver este problema, debe recopilar los datos de audio más cerca de la fuente. Los errores de inserción significan que el audio se grabó en un entorno ruidoso y pueden producirse interferencias, lo que ocasiona problemas de reconocimiento. Los errores de sustitución se encuentran a menudo cuando se ha proporcionado una muestra insuficiente de términos específicos del dominio como transcripciones con la etiqueta humano o texto relacionado.

Al analizar archivos individuales, puede determinar qué tipo de errores existen y qué errores son específicos de un determinado archivo. Comprender los problemas en el nivel de archivo le ayudará a identificar las mejoras.

## Creación de una prueba

Si quiere probar la calidad del modelo de línea de base de texto a voz de Microsoft o un modelo personalizado que haya entrenado, puede comparar dos modelos en paralelo para evaluar la precisión. La comparación incluye los resultados de reconocimiento y WER. Normalmente, un modelo personalizado se compara con el modelo de línea de base de Microsoft.

Para evaluar los modelos en paralelo:

1. Inicie sesión en el [portal de Custom Speech](#).
2. Vaya a **Speech-to-text > Custom Speech > Testing** (Conversión de voz a texto > Custom Speech > Pruebas).
3. Haga clic en **Add test** (Aregar prueba).
4. Seleccione **Evaluate accuracy** (Evaluar precisión). Proporcione a la prueba un nombre y una descripción, y seleccione el conjunto de datos de transcripción con la etiqueta audio + humano.
5. Puede seleccionar hasta dos modelos para probar.
6. Haga clic en **Create**(Crear).

Después de que la prueba se ha creado correctamente, puede comparar los resultados en paralelo.

## Comparación en paralelo

Una vez finalizada la prueba, que viene indicado por el cambio de estado a *Succeeded* (Correcto), encontrará un número de WER para ambos modelos incluidos en la prueba. Haga clic en el nombre de la prueba para ver la página de detalles de las pruebas. En esta página se muestran todas las expresiones del conjunto de datos y se indican los resultados del reconocimiento de los dos modelos al lado de la transcripción del conjunto de datos enviado. Con el fin de inspeccionar la comparación en paralelo, puede alternar los distintos tipos de error, como inserción, eliminación y sustitución. Al escuchar el audio y comparar los resultados del reconocimiento de cada columna, que muestra la transcripción con la etiqueta humano y los resultados de los dos modelos de voz a texto, puede decidir qué modelo satisface sus necesidades y dónde hacen falta entrenamiento y mejoras adicionales.

## Pasos siguientes

- [Entrenamiento del modelo](#)
- [Implementación del modelo](#)

## Recursos adicionales

- [Preparación y prueba de los datos](#)
- [Inspección de los datos](#)

# Entrenamiento de un modelo de Custom Speech

13/01/2020 • 5 minutes to read • [Edit Online](#)

El entrenamiento de un modelo de voz a texto puede mejorar la precisión del reconocimiento en el modelo de línea de base de Microsoft o en un modelo personalizado que tenga pensado crear. Un modelo se entrena mediante transcripciones con etiqueta humana y el texto relacionado. Estos conjuntos de datos, junto con los datos de audio cargados anteriormente, se usan para refinar y entrenar el modelo de texto a voz para el reconocimiento de palabras, frases, acrónimos, nombres y otros términos específicos de cada producto. Cuantos más conjuntos de datos en dominio proporcione (datos que están relacionados con qué dirán los usuarios y qué espera reconocer), más preciso será su modelo y, por lo tanto, mejor será el reconocimiento. Tenga en cuenta que el uso de datos no relacionados en el entrenamiento puede reducir o perjudicar la precisión del modelo.

## Uso del entrenamiento para solucionar problemas de precisión

Si se está encontrando con problemas de reconocimiento en el modelo, el uso de transcripciones con etiqueta humana y los datos relacionados para realizar entrenamiento adicional puede ayudar a mejorar la precisión. Use esta tabla para determinar qué conjunto de datos se debe usar para solucionar su problema:

CASO DE USO	TIPO DE DATOS
Mejorar la precisión del reconocimiento en el vocabulario y la gramática específicos del sector, como la terminología médica o la jerga de TI.	Texto relacionado (frases o expresiones)
Definir el formato fonético y mostrado de una palabra o término que tenga una pronunciación no estándar, como nombres de producto o acrónimos	Texto relacionado (pronunciación)
Mejorar la precisión del reconocimiento en estilos de habla, acentos o ruidos de fondo específicos.	Transcripciones de audio con etiqueta humana

### IMPORTANT

Si no ha cargado un conjunto de datos, consulte [Preparación y prueba de los datos](#). En este documento se proporcionan instrucciones para cargar datos y directrices para crear conjuntos de datos de alta calidad.

## Entrenamiento y evaluación de un modelo

El primer paso para entrenar un modelo es cargar los datos de entrenamiento. Use [Preparación y prueba de los datos](#) para obtener instrucciones paso a paso para preparar las transcripciones con etiqueta humana y el texto relacionado (expresiones y pronunciaciones). Después de cargar los datos de entrenamiento, siga estas instrucciones para empezar a entrenar el modelo:

1. Inicie sesión en el [portal de Custom Speech](#).
2. Vaya a **Speech-to-text > Custom Speech > Training** (Conversión de voz a texto > Custom Speech > Aprendizaje).
3. Haga clic en **Train model** (Entrenar modelo).
4. A continuación, asigne a su entrenamiento un **nombre** y una **descripción**.

5. En el menú desplegable **Scenario and Baseline model** (Escenario y modelo de línea de base), seleccione el escenario que mejor se adapte a su dominio. Si no está seguro de qué escenario elegir, seleccione **General**. El modelo de línea de base es el punto de partida para el entrenamiento. Si no tiene una preferencia, puede usar el último.
6. En la página **Select training data** (Seleccionar datos de entrenamiento), elija uno o varios conjuntos de datos de audio + transcripción con la etiqueta humana que quiera usar para el entrenamiento.
7. Una vez finalizado el entrenamiento, puede elegir realizar pruebas de precisión en el modelo recién entrenado. Este paso es opcional.
8. Seleccione **Create** (Crear) para generar el modelo personalizado.

En la tabla de entrenamiento se muestra una nueva entrada que corresponde a este modelo recién creado. En la tabla también se muestra el estado: Procesando, Correcto, Error.

## Evaluación de la precisión de un modelo entrenado

Puede inspeccionar los datos y evaluar la precisión del modelo mediante estos documentos:

- [Inspección de los datos](#)
- [Evaluación de los datos](#)

Si eligió probar la precisión, es importante seleccionar un conjunto de datos acústico diferente del usado para la creación del modelo para obtener una idea realista del rendimiento del modelo.

## Pasos siguientes

- [Implementación del modelo](#)

## Recursos adicionales

- [Preparación y prueba de los datos](#)
- [Inspección de los datos](#)
- [Evaluación de los datos](#)
- [Entrenamiento del modelo](#)

# Implementación de un modelo personalizado

13/01/2020 • 3 minutes to read • [Edit Online](#)

Después de haber cargado e inspeccionar los datos y de haber evaluado y entrenado un modelo personalizado, puede implementar un punto de conexión personalizado para usar con sus productos, herramientas y aplicaciones. En este documento, aprenderá a crear e implementar un punto de conexión mediante el [portal de Custom Speech](#).

## Creación de un punto de conexión personalizado

Para crear un punto de conexión personalizado, inicie sesión en el [portal de Custom Speech](#) y seleccione **Implementaciones** en el menú Custom Speech de la parte superior de la página. Si se trata de la primera ejecución, observará que no aparece ningún punto de conexión en la tabla. Después de crear un punto de conexión, usará esta página para realizar el seguimiento de cada punto de conexión implementado.

A continuación, seleccione **Agregar punto de conexión** y escriba un **nombre** y una **descripción** para el punto de conexión personalizado. Luego, seleccione el modelo personalizado que le gustaría asociar a este punto de conexión. Desde esta página, también puede habilitar el registro. El registro le permite supervisar el tráfico del punto de conexión. Si está deshabilitado, no se almacenará el tráfico.

New endpoint

Name \*

My Endpoint

Description

This endpoint will be used in my speech applications

Model \*

MyFirstTraining

Log content from this endpoint

The content logging will enable you to download the audio and the transcriptions from the endpoint. [Learn more](#)

Cancel Add

### NOTE

No olvide aceptar los términos de uso y los detalles de precios.

A continuación, seleccione **Crear**. Esta acción le devolverá a la página **Implementación**. Ahora, la tabla incluye una entrada que se corresponde con el punto de conexión personalizado. El estado del punto de conexión muestra su estado actual. Se puede tardar hasta 30 minutos en crear una instancia de un nuevo punto de conexión con los modelos personalizados. Cuando el estado de la implementación cambie a **Completado**, el punto de conexión estará listo para su uso.

Una vez implementado el punto de conexión, su nombre aparece como un vínculo. Haga clic en el vínculo para mostrar información específica del punto de conexión, como la clave o la dirección URL, y código de ejemplo.

## Visualización de datos de registro

Los datos de registro están disponibles descargar en **Punto de conexión > Detalles**.

## Pasos siguientes

- Use el punto de conexión personalizado con [Speech SDK](#).

## Recursos adicionales

- [Preparación y prueba de los datos](#)
- [Inspección de los datos](#)
- [Evaluación de los datos](#)
- [Entrenamiento del modelo](#)
- [Implementación del modelo](#)

# Creación de transcripciones con etiqueta humana

13/01/2020 • 12 minutes to read • [Edit Online](#)

Si quiere mejorar la precisión del reconocimiento, en especial cuando se producen problemas por eliminaciones o sustituciones incorrectas de palabras, le interesa usar las transcripciones con etiqueta humana junto con los datos de audio. ¿Qué son las transcripciones con etiqueta humana? Fácil; son transcripciones textuales, palabra por palabra, de un archivo de audio.

Se necesita una muestra grande de datos de transcripciones para mejorar el reconocimiento; se recomienda proporcionar entre 10 y 1000 horas de datos de transcripción. En esta página revisaremos las instrucciones diseñadas para ayudarle a crear transcripciones de alta calidad. Esta guía se divide por configuración regional, con secciones para inglés de Estados Unidos, chino mandarín y alemán.

## Inglés de Estados Unidos (en-US)

Las transcripciones con etiqueta humana para audio inglés deben proporcionarse como texto sin formato, utilizando solo caracteres ASCII. Debe evitarse el uso de caracteres de puntuación Latino-1 o Unicode. A menudo estos caracteres se agregan involuntariamente al copiar texto de una aplicación de procesamiento de texto o al extraer datos de páginas web. Si aparecen estos caracteres, asegúrese de que se actualizan con la sustitución por el carácter ASCII correspondiente.

Estos son algunos ejemplos:

CARACTERES PARA EVITAR	SUSTITUCIÓN	NOTAS
"Hello world"	"Hello world"	Las comillas de apertura y cierre se han sustituido por los caracteres ASCII adecuados.
John's day	John's day	El apóstrofo se ha sustituido por el carácter ASCII adecuado.
it was good—no, it was great!	it was good--no, it was great!	El guion largo se ha sustituido por dos guiones.

### Normalización de texto para inglés de Estados Unidos

La normalización de texto es la transformación de palabras en un formato coherente que se utiliza al entrenar un modelo. Algunas reglas de normalización se aplican al texto automáticamente; sin embargo, se recomienda seguir estas instrucciones a la hora de preparar los datos de transcripción con etiqueta humana:

- Escriba las abreviaturas con todas las palabras.
- Escriba las cadenas numéricas no estándar con todas las palabras (por ejemplo, términos contables).
- Los caracteres que no son alfabéticos o los caracteres alfanuméricos combinados deben transcribirse tal como se pronuncian.
- Las abreviaturas que se pronuncian como palabras no deben editarse (por ejemplo, "radar", "laser", "RAM" o "NATO").
- Escriba las abreviaturas que se pronuncian como letras independientes con cada letra separada por un espacio.

Estos son algunos ejemplos de normalización que debe realizar en la transcripción:

TEXTO ORIGINAL	TEXTO DESPUÉS DE LA NORMALIZACIÓN
Dr. Bruce Banner	Doctor Bruce Banner
James Bond, 007	James Bond double oh seven
Ke\$ha	Kesha
How long is the 2x4	How long is the two by four
The meeting goes from 1-3pm	The meeting goes from one to three pm
My blood type is O+	My blood type is O positive
Water is H2O	Water is H 2 O
Play OU812 by Van Halen	Play O U 8 1 2 by Van Halen
UTF-8 with BOM	U T F 8 with BOM

Las siguientes reglas de normalización se aplican automáticamente a las transcripciones:

- Se utilizan letras minúsculas.
- Se quitan todos los signos de puntuación, excepto los apóstrofos dentro de las palabras.
- Se expanden los números de palabras o formas habladas, como los importes en dólares.

Estos son algunos ejemplos de normalización que se realiza de modo automático en la transcripción:

TEXTO ORIGINAL	TEXTO DESPUÉS DE LA NORMALIZACIÓN
"Holy cow!" said Batman.	holy cow said batman
"What?" said Batman's sidekick, Robin.	what said batman's sidekick robin
Go get -em!	go get em
I'm double-jointed	i'm double jointed
104 Elm Street	one oh four Elm street
Tune to 102.7	tune to one oh two point seven
Pi is about 3.14	pi is about three point one four
It costs \$3.14	it costs three fourteen

## Chino mandarín (zh-CN)

Las transcripciones con etiqueta humana de audio en chino mandarín deben codificarse como UTF-8 con un marcador de orden de bytes. Evite el uso de caracteres de puntuación de ancho medio. Estos caracteres se pueden incluir de forma involuntaria al preparar los datos en un programa de procesamiento de texto o al extraerlos de páginas web. Si aparecen estos caracteres, asegúrese de que se actualizan con la sustitución por el ancho completo correspondiente.

Estos son algunos ejemplos:

CARÁCTERES PARA EVITAR	SUSTITUCIÓN	NOTAS
"你好"	"你好"	Las comillas de apertura y cierre se han sustituido por los caracteres adecuados.
需要什么帮助?	需要什么帮助？	El signo de interrogación se ha sustituido por el carácter adecuado.

### Normalización de texto para chino mandarín

La normalización de texto es la transformación de palabras en un formato coherente que se utiliza al entrenar un modelo. Algunas reglas de normalización se aplican al texto automáticamente; sin embargo, se recomienda seguir estas instrucciones a la hora de preparar los datos de transcripción con etiqueta humana:

- Escriba las abreviaturas con todas las palabras.
- Escriba cadenas numéricas en forma hablada.

Estos son algunos ejemplos de normalización que debe realizar en la transcripción:

TEXTO ORIGINAL	TEXTO DESPUÉS DE LA NORMALIZACIÓN
我今年 21	我今年二十一
3 号楼 504	三号 楼 五 零 四

Las siguientes reglas de normalización se aplican automáticamente a las transcripciones:

- Se quitan todos los signos de puntuación.
- Los números se expanden a la forma hablada.
- Las letras de ancho completo se convierten en letras de ancho medio.
- Uso de letras mayúsculas para todas las palabras en inglés

Estos son algunos ejemplos de normalización que se realiza de modo automático en la transcripción:

TEXTO ORIGINAL	TEXTO DESPUÉS DE LA NORMALIZACIÓN
3.1415	三 点 一 四 一 五
¥ 3.5	三 元 五 角
w f y z	W F Y Z
1992 年 8 月 8 日	一 九 九 二 年 八 月 八 日
你吃饭了吗?	你 吃饭 了 吗
下午 5:00 的航班	下 午 五 点 的 航 班
我今年 21 岁	我 今 年 二 十 一 岁

### Alemán (de-DE) y otros idiomas

Las transcripciones con etiqueta humana de audio en alemán (y otros idiomas distintos del inglés o el chino

mandarín) deben codificarse como UTF-8 con un marcador de orden de bytes. Debe proporcionarse una transcripción con etiqueta humanos para cada archivo de audio.

### Normalización de texto para alemán

La normalización de texto es la transformación de palabras en un formato coherente que se utiliza al entrenar un modelo. Algunas reglas de normalización se aplican al texto automáticamente; sin embargo, se recomienda seguir estas instrucciones a la hora de preparar los datos de transcripción con etiqueta humana:

- Los puntos decimales se escriben como "," y no como ":".
- Los separadores de hora se escriben como ":"y no como "." (por ejemplo: 12:00 Uhr).
- Las abreviaturas, como "ca." no se reemplazan. Se recomienda usar la forma hablada completa.
- Se quitan los cuatro operadores matemáticos principales: +, -, \*, /. Se recomienda sustituirlos por su forma escrita: "plus", "minus", "mal" y "geteilt".
- Se quitan los operadores de comparación (=, <, y >). Se recomienda sustituirlos con "gleich", "kleiner als," y "grösser als".
- Las fracciones, como 3/4, se usan con la forma escrita (por ejemplo, "drei viertel" en lugar de 3/4).
- El símbolo "€" se reemplaza por su forma escrita "Euro".

Estos son algunos ejemplos de normalización que debe realizar en la transcripción:

TEXTO ORIGINAL	TEXTO DESPUÉS DE LA NORMALIZACIÓN DEL USUARIO	TEXTO DESPUÉS DE LA NORMALIZACIÓN DEL SISTEMA
Es ist 12.23 Uhr	Es ist 12:23 Uhr	es ist zwölf uhr drei und zwanzig uhr
{12.45}	{12,45}	zwölf komma vier fünf
2 + 3 - 4	2 plus 3 minus 4	zwei plus drei minus vier

Las siguientes reglas de normalización se aplican automáticamente a las transcripciones:

- Se usan letras en minúsculas para todo el texto.
- Se quita toda la puntuación, incluidos varios tipos de comillas ("prueba", 'prueba', "prueba y «prueba» son correctos).
- Se descartan todas las filas que contengan un carácter especial de este conjunto: € ₣ ¥ ! § © ª ¬ ® ° ± ² µ × ÿ Ø ñ ñ.
- Se expanden los números a la forma hablada, incluidos importes en dólares o euros.
- Se acepta la diéresis solo a, o y u. Otras se reemplazará por "th" o se descartarán.

Estos son algunos ejemplos de normalización que se realiza de modo automático en la transcripción:

TEXTO ORIGINAL	TEXTO DESPUÉS DE LA NORMALIZACIÓN
Frankfurter Ring	frankfurter ring
iEine Frage!	eine frage
wir, haben	wir haben

## Pasos siguientes

- [Preparación y prueba de los datos](#)
- [Inspección de los datos](#)

- Evaluación de los datos
- Entrenamiento del modelo
- Implementación del modelo

# Tutorial: Creación de un modelo de inquilino (versión preliminar)

14/01/2020 • 11 minutes to read • [Edit Online](#)

Tenant Model (Custom Speech con datos de Office 365) es un servicio de participación para clientes empresariales de Office 365 que genera automáticamente un modelo de reconocimiento de voz personalizado a partir de los datos de Office 365 de su organización. El modelo está optimizado para términos técnicos, jerga y nombres de personas, y todo ello de forma segura y compatible.

## IMPORTANT

Si su organización se inscribe mediante el servicio Tenant Model, el servicio de voz puede acceder al modelo de lenguaje de su organización. El modelo se genera a partir de documentos y correos electrónicos de grupos públicos de Office 365, que puede ver cualquier usuario de la organización. El administrador de Office 365 de la organización puede activar o desactivar el uso del modelo de lenguaje en toda la organización desde el portal de administración de Office 365.

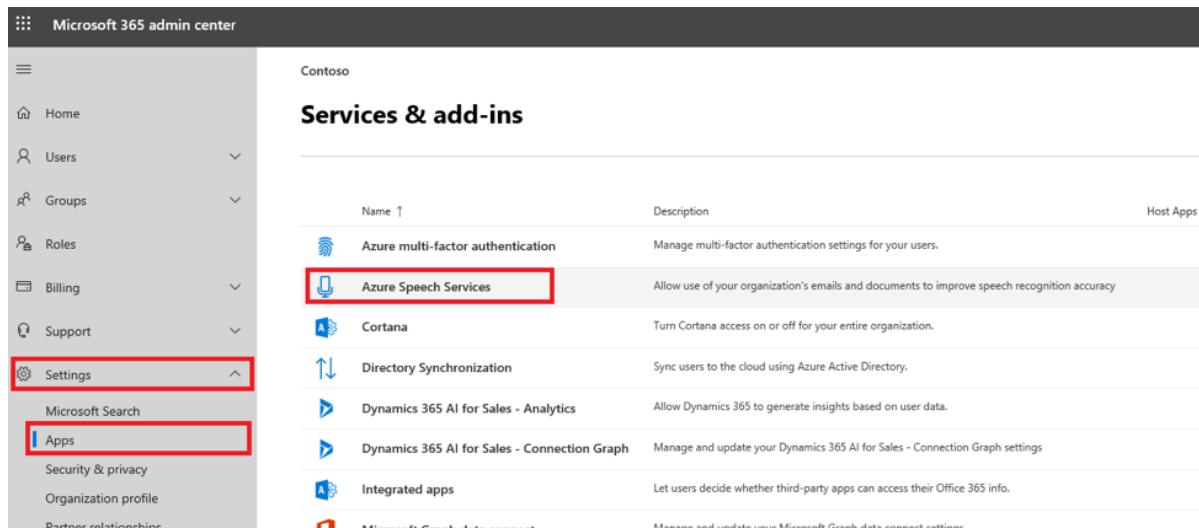
En este tutorial, aprenderá a:

- Inscribirse en Tenant Model desde el Centro de administración de Microsoft 365
- Obtener una clave de suscripción del servicio Voz
- Crear un modelo de inquilino
- Implementar un modelo de inquilino
- Usar un modelo de inquilino con el SDK de Voz

## Inscripción en el servicio Tenant Model

Para poder implementar un modelo de inquilino, es preciso estar inscrito en el servicio Tenant Model. La inscripción se completa en el Centro de administración de Microsoft 365 y solo puede realizarla un administrador de Microsoft 365.

1. Inicie sesión en el [Centro de administración de Microsoft 365](#).
2. En el panel izquierdo, seleccione **Configuración, Aplicaciones y Servicios de voz de Azure**.



The screenshot shows the Microsoft 365 admin center interface. On the left, there's a navigation menu with options like Home, Users, Groups, Roles, Billing, Support, Settings (which is expanded), Microsoft Search, Apps (which is also expanded), Security & privacy, Organization profile, and Partner relationships. The 'Apps' section under 'Settings' is highlighted with a red box. On the right, the main content area is titled 'Services & add-ins'. It lists several services with their descriptions and host apps. One service, 'Azure Speech Services', is also highlighted with a red box. The table structure is as follows:

Name ↑	Description	Host Apps
Azure multi-factor authentication	Manage multi-factor authentication settings for your users.	
Azure Speech Services	Allow use of your organization's emails and documents to improve speech recognition accuracy.	
Cortana	Turn Cortana access on or off for your entire organization.	
Directory Synchronization	Sync users to the cloud using Azure Active Directory.	
Dynamics 365 AI for Sales - Analytics	Allow Dynamics 365 to generate insights based on user data.	
Dynamics 365 AI for Sales - Connection Graph	Manage and update your Dynamics 365 AI for Sales - Connection Graph settings.	
Integrated apps	Let users decide whether third-party apps can access their Office 365 info.	
Microsoft Graph data connect	Manage and update your Microsoft Graph data connect settings.	

3. Seleccione la casilla **Permitir el modelo de idioma de toda la organización** y, después, **Guardar**.

cambios.

The screenshot shows the Microsoft 365 admin center interface. On the left, there's a navigation sidebar with options like Home, Users, Groups, Roles, Billing, Support, Settings, Microsoft Search, Apps, Security & privacy, and Organization profile. The main content area is titled 'Services & add-ins' and shows a list of services. One service, 'Azure Speech Services', is selected. Its details pane on the right includes a description about language models and a checkbox labeled 'Allow the organization-wide language model'. Below the description is a 'Save changes' button, which is also highlighted with a red box.

Para desactivar la instancia del modelo de inquilino:

1. Repita los pasos 1 y 2 anteriores.
2. Desactive la casilla **Permitir el modelo de idioma de toda la organización** y seleccione **Guardar cambios**.

## Obtener una clave de suscripción del servicio Voz

Para usar un modelo de inquilino con el SDK de Voz, necesita un recurso de Voz y su clave de suscripción asociada.

1. Inicie sesión en [Azure Portal](#).
2. Seleccione **Crear un recurso**.
3. En el cuadro **Buscar en Marketplace**, escriba **Speech**.
4. En la lista de resultados, seleccione **Speech** y, después, **Crear**.
5. Siga las instrucciones en pantalla para crear el recurso. Asegúrese de lo siguiente:
  - La **ubicación** se ha establecido en **eastus** o **westus**.
  - El **plan de tarifa** está establecido en **S0**.
6. Seleccione **Crear**.

Después de unos minutos, se crea el recurso. La clave de suscripción está disponible en la sección **Información general** del recurso.

## Creación de un modelo de lenguaje

Una vez que el administrador haya habilitado Tenant Model para su organización, puede crear un modelo de lenguaje basado en los datos de Office 365.

1. Inicie sesión en [Speech Studio](#).
2. En la parte superior derecha, seleccione **Settings** (Configuración) (ícono del engranaje) y, después, **Tenant Model settings** (Configuración de Tenant Model).

The screenshot shows the 'Cognitive Services | Speech Customization' page in Speech Studio. At the top, there's a navigation bar with icons for Home, Help, and Sign in. Below the navigation bar, there are two main links: 'Subscription' and 'Tenant Model settings'. The 'Tenant Model settings' link is highlighted with a red box.

Speech Studio muestra un mensaje que le indica que si es apto para crear un modelo de inquilino.

#### NOTE

Los clientes empresariales de Office 365 en Norteamérica son aptos para crear un modelo de inquilino (inglés). Para los clientes de Caja de seguridad del cliente, Clave de cliente u Office 365 Administración Pública característica no está disponible. Para determinar si es un cliente de Caja de seguridad del cliente o de Clave de cliente, consulte:

- [Caja de seguridad del cliente](#)
- [Clave de cliente](#)
- [Office 365 Administración Pública](#)

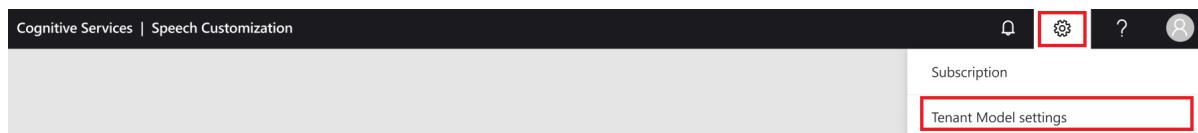
### 3. Seleccione **Habilitar envío**.

Cuando el modelo de inquilino esté listo, recibirá un mensaje de correo electrónico de confirmación con más instrucciones.

## Implementación de un modelo de inquilino

Cuando la instancia del modelo de inquilino esté lista, siga estos pasos para implementarla:

1. En el mensaje de correo electrónico de confirmación, seleccione el botón **View model** (Ver modelo). O bien, inicie sesión en [Speech Studio](#).
2. En la parte superior derecha, seleccione **Settings** (Configuración) (ícono del engranaje) y, después, **Tenant Model settings** (Configuración de Tenant Model).



### 3. Seleccione **Implementar**.

Cuando el modelo se haya implementado, el estado cambiará a *Implementado*.

## Usar un modelo de inquilino con el SDK de Voz

Ahora que ha implementado el modelo, puede usarlo con el SDK de Voz. En esta sección, se usa un código de ejemplo para llamar al servicio de voz mediante la autenticación de Azure Active Directory (Azure AD).

Echemos un vistazo al código que usará para llamar al SDK de Voz en C#. En este ejemplo, el reconocimiento de voz se realiza mediante un modelo de inquilino. En esta guía se da por supuesto que la plataforma ya está configurada. Si necesita ayuda para el programa de configuración, consulte [Inicio rápido: Reconocimiento de voz, C# \(.NET Core\)](#).

Copie este código en el proyecto:

```
namespace PrincetonSROnly.FrontEnd.Samples
{
    using System;
    using System.Collections.Generic;
    using System.IO;
    using System.Net.Http;
    using System.Text;
    using System.Text.RegularExpressions;
    using System.Threading.Tasks;
    using Microsoft.CognitiveServices.Speech;
    using Microsoft.CognitiveServices.Speech.Audio;
    using Microsoft.IdentityModel.Clients.ActiveDirectory;
    using Newtonsoft.Json.Linq;
```

```

// ServiceApplicationId is a fixed value. No need to change it.

public class TenantLMSample
{
    private const string EndpointUriArgName = "EndpointUri";
    private const string SubscriptionKeyArgName = "SubscriptionKey";
    private const string UsernameArgName = "Username";
    private const string PasswordArgName = "Password";
    private const string ClientApplicationId = "f87bc118-1576-4097-93c9-dbf8f45ef0dd";
    private const string ServiceApplicationId = "18301695-f99d-4cae-9618-6901d4bdc7be";

    public static async Task ContinuousRecognitionWithTenantLMAsync(Uri endpointUri, string
subscriptionKey, string audioDirPath, string username, string password)
    {
        var config = SpeechConfig.FromEndpoint(endpointUri, subscriptionKey);

        // Passing client specific information for obtaining LM
        if (string.IsNullOrEmpty(username) || string.IsNullOrEmpty(password))
        {
            config.AuthorizationToken = await
AcquireAuthTokenWithInteractiveLoginAsync().ConfigureAwait(false);
        }
        else
        {
            config.AuthorizationToken = await AcquireAuthTokenWithUsernamePasswordAsync(username,
password).ConfigureAwait(false);
        }

        var stopRecognition = new TaskCompletionSource<int>();

        // Creates a speech recognizer using file as audio input.
        // Replace with your own audio file name.
        using (var audioInput = AudioConfig.FromWavFileInput(audioDirPath))
        {
            using (var recognizer = new SpeechRecognizer(config, audioInput))
            {
                // Subscribes to events
                recognizer.Recognizing += (s, e) =>
                {
                    Console.WriteLine($"RECOGNIZING: Text={e.Result.Text}");
                };

                recognizer.Recognized += (s, e) =>
                {
                    if (e.Result.Reason == ResultReason.RecognizedSpeech)
                    {
                        Console.WriteLine($"RECOGNIZED: Text={e.Result.Text}");
                    }
                    else if (e.Result.Reason == ResultReason.NoMatch)
                    {
                        Console.WriteLine($"NOMATCH: Speech could not be recognized.");
                    }
                };
            };

            recognizer.Canceled += (s, e) =>
            {
                Console.WriteLine($"CANCELED: Reason={e.Reason}");
                if (e.Reason == CancellationReason.Error)
                {
                    Exception exp = new Exception(string.Format("Error Code: {0}\nError Details{1}\nIs
your subscription information updated?", e.ErrorCode, e.ErrorDetails));
                    throw exp;
                }

                stopRecognition.TrySetResult(0);
            };
        }

        recognizer.SessionStarted += (s, e) =>
        {

```

```

        Console.WriteLine("\n    Session started event.");
    };

    recognizer.SessionStopped += (s, e) =>
    {
        Console.WriteLine("\n    Session stopped event.");
        Console.WriteLine("\nStop recognition.");
        stopRecognition.TrySetResult(0);
    };

    // Starts continuous recognition. Uses StopContinuousRecognitionAsync() to stop
recognition.
    await recognizer.StartContinuousRecognitionAsync().ConfigureAwait(false);

    // Waits for completion.
    // Use Task.WaitAny to keep the task rooted.
    Task.WaitAny(new[] { stopRecognition.Task });

    // Stops recognition.
    await recognizer.StopContinuousRecognitionAsync().ConfigureAwait(false);
}
}

public static void Main(string[] args)
{
    var arguments = new Dictionary<string, string>();
    string inputArgNamePattern = "--";
    Regex regex = new Regex(inputArgNamePattern);
    if (args.Length > 0)
    {
        foreach (var arg in args)
        {
            var userArgs = arg.Split("=");
            arguments[regex.Replace(userArgs[0], string.Empty)] = userArgs[1];
        }
    }

    var endpointString = arguments.GetValueOrDefault(EndpointUriArgName,
$"wss://westus.online.princeton.customspeech.ai/msgraphcustomspeech/conversation/v1");
    var endpointUri = new Uri(endpointString);

    if (!arguments.ContainsKey(SubscriptionKeyArgName))
    {
        Exception exp = new Exception("Subscription Key missing! Please pass in a Cognitive services
subscription Key using -SubscriptionKey=\"your_subscription_key\" +
                    "Find more information on creating a Cognitive services resource and accessing your
Subscription key here: https://docs.microsoft.com/azure/cognitive-services/cognitive-services-apis-create-
account?tabs=multiservice%2Cwindows");
        throw exp;
    }

    var subscriptionKey = arguments[SubscriptionKeyArgName];
    var username = arguments.GetValueOrDefault(UsernameArgName, null);
    var password = arguments.GetValueOrDefault>PasswordArgName, null);

    var audioDirPath =
Path.Combine(Path.GetDirectoryName(System.Reflection.Assembly.GetExecutingAssembly().Location),
"../../../../AudioSamples/DictationBatman.wav");
    if (!File.Exists(audioDirPath))
    {
        Exception exp = new Exception(string.Format("Audio File does not exist at path: {0}",
audioDirPath));
        throw exp;
    }

    ContinuousRecognitionWithTenantLMAsync(endpointUri, subscriptionKey, audioDirPath, username,
password).GetAwaiter().GetResult();
}

```

```

        }

        private static async Task<string> AcquireAuthTokenWithUsernamePasswordAsync(string username, string
password)
{
    var tokenEndpoint = "https://login.microsoftonline.com/common/oauth2/token";
    var postBody = $"resource={ServiceApplicationId}&client_id=
{ClientApplicationId}&grant_type=password&username={username}&password={password}";
    var stringContent = new StringContent(postBody, Encoding.UTF8, "application/x-www-form-
urlencoded");
    using (HttpClient httpClient = new HttpClient())
    {
        var response = await httpClient.PostAsync(tokenEndpoint, stringContent).ConfigureAwait(false);

        if (response.IsSuccessStatusCode)
        {
            var result = await response.Content.ReadAsStringAsync().ConfigureAwait(false);

            JObject jobject = JObject.Parse(result);
            return jobject["access_token"].Value<string>();
        }
        else
        {
            throw new Exception($"Requesting token from {tokenEndpoint} failed with status code
{response.StatusCode}: {await response.Content.ReadAsStringAsync().ConfigureAwait(false)}");
        }
    }
}

private static async Task<string> AcquireAuthTokenWithInteractiveLoginAsync()
{
    var authContext = new
AuthenticationContext("https://login.windows.net/microsoft.onmicrosoft.com");
    var deviceCodeResult = await authContext.AcquireDeviceCodeAsync(ServiceApplicationId,
ClientApplicationId).ConfigureAwait(false);

    Console.WriteLine(deviceCodeResult.Message);

    var authResult = await
authContext.AcquireTokenByDeviceCodeAsync(deviceCodeResult).ConfigureAwait(false);

    return authResult.AccessToken;
}
}
}

```

Después tendrá que volver a compilar y ejecutar el proyecto desde la línea de comandos. Antes de ejecutar el comando, actualice algunos parámetros, para lo que debe seguir estos pasos:

1. Reemplace <Username> y <Password> por los valores de un usuario de inquilino válido.
2. Reemplace <Subscription-Key> por la clave de suscripción del recurso de Voz. Este valor está disponible en la hoja de **información general** del recurso de Voz de [Azure Portal](#).
3. Reemplace <Endpoint-Uri> por el siguiente punto de conexión. Asegúrese de que reemplaza {your region} por la región donde se creó el recurso de Voz. Se admiten estas regiones: westus , westus2 y eastus . La información de la región está disponible en la sección de **información general** del recurso de Voz de [Azure Portal](#).

"wss://{{your region}}.online.princeton.customspeech.ai/msgraphcustomspeech/conversation/v1".

4. Ejecute el siguiente comando:

```
dotnet TenantLMSample.dll --Username=<Username> --Password=<Password> --SubscriptionKey=<Subscription-  
Key> --EndpointUri=<Endpoint-Uri>
```

En este tutorial, ha aprendido a usar los datos de Office 365 para crear un modelo de reconocimiento de voz personalizado, a implementarlo y a usarlo con el SDK de Voz.

## Pasos siguientes

- [Speech Studio](#)
- [Acerca del SDK de Voz](#)

# Introducción al control de voz neuronal personalizada

13/01/2020 • 3 minutes to read • [Edit Online](#)

Obtenga más información sobre el proceso de introducción a la voz neuronal personalizada.

## Compromiso con la innovación responsable

Como parte del compromiso de Microsoft con el diseño de IA responsable, se ha preparado un conjunto de materiales para guiar a los clientes en el uso de la voz neuronal personalizada. Las directrices y la información que se encuentran aquí se basan en los [principios de Microsoft para la innovación responsable en IA](#).

### Instrucciones para implementar la voz neuronal personalizada

- [Directrices para la implementación responsable](#): nuestras recomendaciones principales basadas en nuestra investigación.
- [Divulgación de talento de voz](#): lo que necesita saber e informar al actor de voz sobre la tecnología para su uso responsable
- [Diseño de divulgación](#): cómo diseñar experiencias para que los usuarios sepan cuándo se utiliza una voz sintética y confíen en su servicio

### Motivo por el que la voz neuronal personalizada es una tecnología controlada

Nuestro objetivo es proteger los derechos de las personas y la sociedad, fomentar interacciones transparentes entre equipos y humanos y contrarrestar la proliferación de ultrafalsos ("deepfakes") nocivos y contenido engañoso. Por este motivo, el uso de la voz neuronal personalizada está controlado. Los clientes obtienen acceso a la tecnología solo tras la revisión de sus aplicaciones y después de que se hayan comprometido a usarla en consonancia con nuestros principios éticos.

### Nuestro proceso de control

Para obtener acceso a la voz neuronal personalizada, deberá llenar el formulario de admisión en línea. Inicie la aplicación [aquí](#).

El acceso al servicio de voz neuronal personalizado está sujeto a la exclusiva discreción de Microsoft en función de los criterios de idoneidad, el proceso de investigación y la disponibilidad para admitir un número limitado de clientes durante esta versión preliminar controlada.

Como parte del proceso de solicitud, deberá comprometerse a obtener el permiso explícito por escrito del actor de voz antes de crear una fuente de voz, lo que incluye compartir la [Divulgación de talento de voz](#). También debe aceptar que, cuando implemente la fuente de voz, su implementación [divulgará la naturaleza sintética](#) del servicio a los usuarios, proporcionará la atribución al servicio de voz sintética de Microsoft en sus condiciones de servicio y admitirá un canal de comentarios que permita a los usuarios del servicio informar de problemas y compartir detalles con Microsoft. Obtenga más información sobre las condiciones de uso [aquí](#).

## Documentos de referencia

- [Divulgación de talento de voz](#)
- [Directrices para la implementación responsable de la tecnología de voz sintética](#)
- [Procedimiento de divulgación](#)

## Pasos siguientes

- Directrices para la implementación responsable de la tecnología de voz sintética

# Directrices para la implementación responsable de la tecnología de voz sintética

13/01/2020 • 6 minutes to read • [Edit Online](#)

A continuación se presentan las directrices generales de diseño de Microsoft para usar la tecnología de voz sintética. Se desarrollaron en estudios que Microsoft llevó a cabo con actores de voz, consumidores e incluso personas con trastornos del lenguaje para guiar el desarrollo responsable de la voz sintética.

## Consideraciones generales

En la implementación de la tecnología de síntesis de voz, se aplican las siguientes directrices en la mayoría de los escenarios.

### **Divulgación de cuando la voz es sintética**

Divulgar el hecho de que la voz se generó por ordenador no solo minimiza el riesgo de resultados perjudiciales derivados de una decepción, sino que también aumenta la confianza en la organización que presenta la voz. Más información sobre [cómo divulgar información](#).

### **Selección de los tipos de voz adecuados para su escenario**

Considere cuidadosamente el contexto de uso y los posibles daños asociados al uso de la voz sintética. Por ejemplo, es posible que las voces sintéticas de alta fidelidad no sean adecuadas en situaciones de alto riesgo, como la mensajería personal, las transacciones financieras o situaciones complejas que requieran de la adaptación humana o la empatía. Los usuarios también pueden tener expectativas diferentes para cada tipo de voz. Por ejemplo, al escuchar noticias confidenciales leídas por una voz sintética, algunos usuarios prefieren una lectura más empática y similar a los seres humanos, mientras que otros prefieren una voz más imparcial y monótona. Considere la posibilidad de probar la aplicación para comprender mejor las preferencias del usuario.

### **Transparencia en cuanto a funcionalidades y limitaciones**

Es más probable que los usuarios tengan mayores expectativas al interactuar con agentes de voz sintéticos de alta fidelidad. Por consiguiente, cuando las funcionalidades del sistema no cumplen esas expectativas, la confianza puede verse afectada y puede dar lugar a experiencias desagradables o incluso dañinas.

### **Ofrecimiento de soporte técnico humano opcional**

En escenarios transaccionales ambiguos (por ejemplo, un centro telefónico de soporte técnico), los usuarios no siempre confían en que un agente artificial responderá correctamente a sus solicitudes. El soporte técnico humano puede ser necesario en estas situaciones, independientemente de la calidad realista de la voz o la funcionalidad del sistema.

## Consideraciones sobre los actores de voz

Al trabajar con actores de voz, como actores de doblaje, para crear voces sintéticas, se aplica la directriz siguiente.

### **Obtención del consentimiento significativo del actor de voz**

Los actores de voz esperan tener control sobre su fuente de voz (cómo y dónde se usará), así como ser compensados siempre que esta se use. Por lo tanto, los propietarios del sistema deben obtener el permiso explícito por escrito del actor de voz y tener claras especificaciones contractuales sobre los casos de uso, la duración de uso, la compensación, etc. Algunos actores de voz no saben cuáles son los posibles usos malintencionados de la tecnología y los propietarios del sistema deben instruirlos sobre las funcionalidades de la tecnología. Para obtener más información sobre los actores de voz y el consentimiento, consulte [Divulgación de](#)

[talento de voz.](#)

## Consideraciones para las personas con trastornos del habla

Al trabajar con personas con trastornos del habla para crear o implementar la tecnología de voz sintética, se aplican las siguientes directrices.

### **Ofrecimiento de directrices para celebrar contratos**

Proporcione directrices para celebrar contratos con personas que usan la voz sintética como ayuda para hablar. El contrato debería especificar las partes que poseen la voz, la duración de uso, los criterios de transferencia de propiedad, los procedimientos para eliminar la fuente de voz y cómo evitar el acceso no autorizado. Además, habilite la transferencia contractual de la propiedad de la fuente de voz después de la muerte a los familiares si esa persona ha dado su permiso.

### **Consideración de las incoherencias en los patrones de habla**

En el caso de las personas con trastornos del habla que registran sus propias fuentes de voz, las incoherencias en su patrón de habla (mala pronunciación o incapacidad de pronunciar ciertas palabras) pueden complicar el proceso de grabación. En estos casos, la tecnología de voz sintética y las sesiones de grabación deberían adaptarse a ellos (es decir, proporcionar descansos y un número adicional de sesiones de grabación).

### **Permitir la modificación a lo largo del tiempo**

Las personas con trastornos del habla quieren hacer actualizaciones en su voz sintética para reflejar su edad (por ejemplo, un niño que alcanza la pubertad). Las personas también pueden tener preferencias estilísticas que cambian con el tiempo y puede que quieran realizar cambios en las características de tono, acento u otras.

## Documentos de referencia

- [Divulgación de talento de voz](#)
- [Información general sobre control](#)
- [Procedimiento de divulgación](#)
- [Modelos de diseño de divulgación](#)

## Pasos siguientes

- [Divulgación de talento de voz](#)
- [Procedimiento de divulgación](#)
- [Modelos de diseño de divulgación](#)

# Directrices de diseño de divulgación de información

13/01/2020 • 7 minutes to read • [Edit Online](#)

Aprenda a generar y mantener la confianza con los clientes al ser transparentes sobre la naturaleza sintética de su experiencia de voz.

## ¿Qué es la divulgación?

La divulgación es un medio para que los usuarios sepan que la voz que están escuchando o con la que están interactuando está generada de forma sintética.

## ¿Por qué es necesaria la divulgación?

La necesidad de divulgar los orígenes sintéticos de una voz generada por ordenador es relativamente nueva. En el pasado, las voces generadas por ordenador eran obvias, ya que nadie las confundiría con una persona real. Sin embargo, todos los días el realismo de las voces sintéticas mejora y se vuelve cada vez más indistinguible de las voces humanas.

## Objetivos

Estos son los principios a tener en cuenta al diseñar experiencias de voz sintéticas:

### **Reforzar la confianza**

Diseñe con la intención de que se produzca un error en la prueba de Turing sin deteriorar la experiencia. Avise a los usuarios que están interactuando con una voz sintética mientras les permite interactuar sin problemas con la experiencia.

### **Adaptarse al contexto de uso**

Comprenda cuándo, dónde y cómo interactuarán los usuarios con la voz sintética para proporcionar el tipo correcto de divulgación en el momento adecuado.

### **Establecer expectativas claras**

Permita a los usuarios detectar y comprender fácilmente las funcionalidades del agente. Ofrezca oportunidades para obtener más información acerca de la tecnología de voz sintética a petición.

### **Aceptar los errores**

Utilice los errores para reforzar las funcionalidades del agente.

## Cómo utilizar esta guía

Esta guía le ayuda a determinar qué patrones de divulgación son los que mejor se adaptan a su experiencia de voz sintética. A continuación, se ofrecen ejemplos de cómo y cuándo utilizarlos. Cada uno de estos modelos está diseñado para maximizar la transparencia con los usuarios sobre la síntesis de voz, al mismo tiempo que se mantiene el diseño humano.

Teniendo en cuenta la gran cantidad de guías de diseño sobre experiencias de voz, nos centramos aquí en lo siguiente:

1. **Evaluación de la divulgación:** un proceso para determinar el tipo de divulgación recomendado para su experiencia de voz sintética
2. **Procedimiento de divulgación:** ejemplos de patrones de divulgación que pueden aplicarse a su

experiencia de voz sintética.

3. **Momento de divulgación:** momentos óptimos para divulgar información a lo largo del recorrido del usuario.

## Evaluación de la divulgación

Tenga en cuenta las expectativas de los usuarios sobre una interacción y el contexto en el que experimentarán la voz. Si el contexto deja claro que una voz sintética está "hablando", la divulgación puede ser mínima, puntual o incluso innecesaria. Los principales tipos de contexto que afectan a la divulgación incluyen el tipo de rol, el tipo de escenario y el nivel de exposición. También resulta útil considerar quién puede estar escuchando.

### Comprensión del contexto

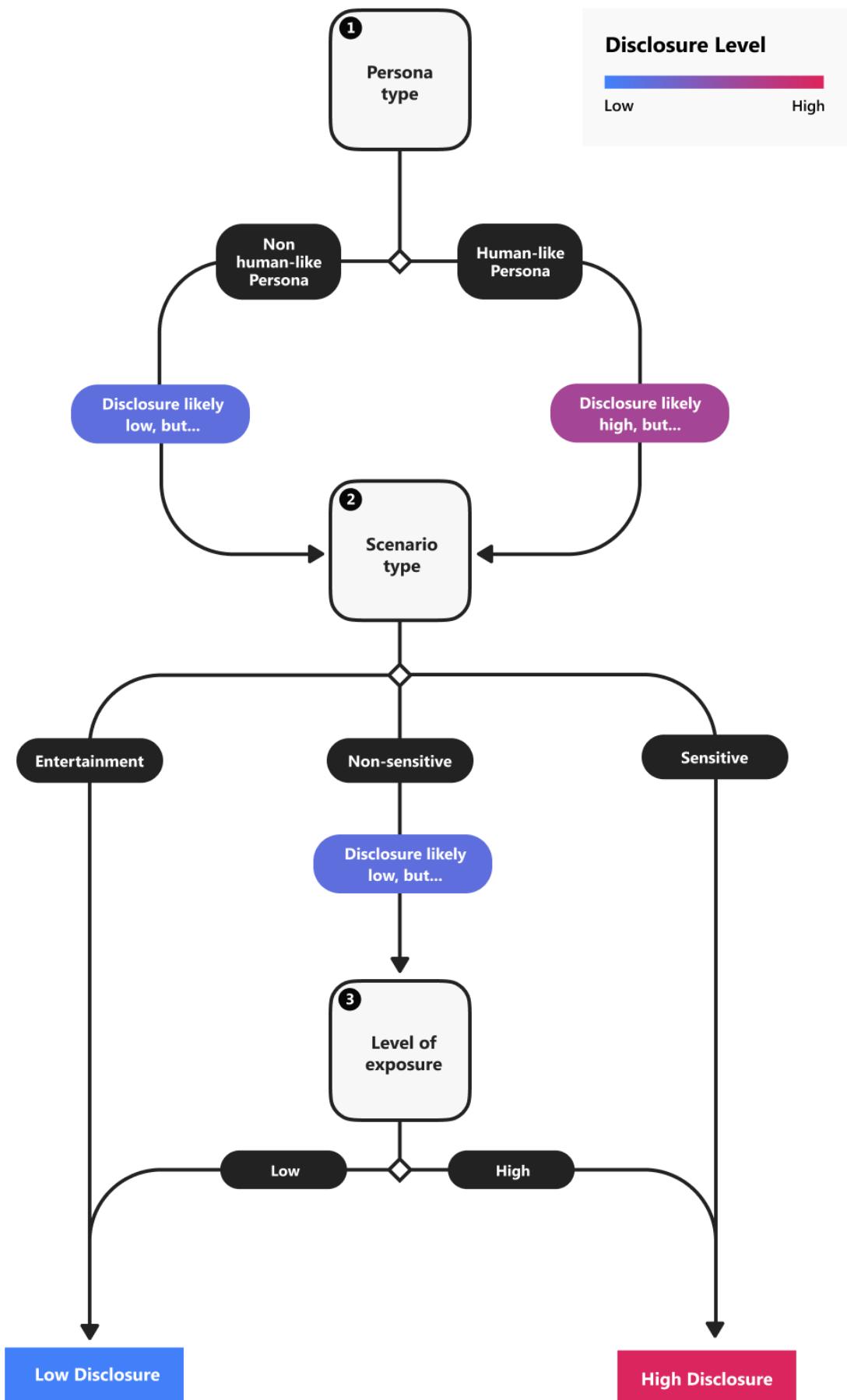
Utilice esta hoja de cálculo para determinar el contexto de su experiencia de voz sintética. La aplicará en el paso siguiente, donde determinará el nivel de divulgación.

	CONTEXTO DE USO	POSIBLES RIESGOS Y DESAFÍOS
<b>1. Tipo de rol</b>	<p><b>Si se cumple alguna de las siguientes condiciones, el rol se ajusta a la categoría "rol similar al humano":</b></p> <ul style="list-style-type: none"><li>• El rol personifica a una persona real, ya sea una representación ficticia o si no (por ejemplo, fotografía o representación generada por ordenador de una persona real).</li><li>• La voz sintética se basa en la voz de una persona real ampliamente reconocible (por ejemplo, una persona famosa o una figura política).</li></ul>	Cuanto más representaciones similares al humano proporcione su rol, más probable será que un usuario lo asocie a una persona real o que crea que el contenido es de una persona real en lugar de generado por un ordenador.
<b>2. Tipo de escenario</b>	<p><b>Si se cumple alguna de las siguientes condiciones, su experiencia de voz se ajusta en la categoría "confidencial":</b></p> <ul style="list-style-type: none"><li>• Obtiene o muestra información personal del usuario.</li><li>• Difunde noticias u información confidencial (por ejemplo, una alerta de emergencia).</li><li>• Pretende ayudar a los usuarios reales a comunicarse entre sí (por ejemplo, lee correos electrónicos o mensajes de texto personales).</li><li>• Proporciona asistencia médica o sanitaria.</li></ul>	Es posible que el uso de voz sintética no resulte adecuado o confiable para las personas que la usan cuando los temas están relacionados con asuntos confidenciales, personales o urgentes. También es posible que esperen el mismo nivel de empatía y conciencia del contexto que con un usuario real.

CONTEXTO DE USO	POSIBLES RIESGOS Y DESAFÍOS
<b>3. Nivel de exposición</b>	<p><b>Lo más probable es que la experiencia de voz se ajuste a la categoría "nivel alto" si:</b></p> <ul style="list-style-type: none"> <li>• El usuario oirá o interactuará con la voz sintética con frecuencia o durante un largo período de tiempo.</li> </ul>

### Determinación del nivel de divulgación

Use el siguiente diagrama para determinar si su experiencia de voz sintética requiere un nivel de divulgación alto o bajo en función del contexto de uso.



## Documentos de referencia

- [Divulgación de talento de voz](#)
- [Directrices para la implementación responsable de la tecnología de voz sintética](#)

- [Información general sobre control](#)

## Pasos siguientes

- [Modelos de diseño de divulgación](#)

# Modelos de diseño de divulgación

13/01/2020 • 19 minutes to read • [Edit Online](#)

Tras determinar el [nivel de divulgación](#) adecuado para su experiencia de voz sintética, es un buen momento para explorar posibles modelos de diseño.

## Información general

Hay un abanico de modelos de diseño de divulgación que puede aplicar a su experiencia de voz sintética. Si el resultado de la evaluación de la divulgación fue "nivel alto de divulgación", se recomienda la **divulgación explícita**; es decir, comunicar los orígenes de la voz sintética directamente. La **divulgación implícita** incluye guías y modelos de interacción que benefician a las experiencias de voz, independientemente de si los niveles de divulgación necesarios son altos o bajos.

**Avatar**

**Display text**

**Sonicon**

**Spoken prompt**



Meet Oso, your digital assistant.



*"Hi, I'm Oso, your digital assistant"*

VISUAL

AUDITORY

### MODELOS DE DIVULGACIÓN EXPLÍCITOS

- Introducción transparente
- Introducción transparente verbal
- Firma explícita
- Personalización y calibración
- Divulgación parental
- Oportunidades para obtener más información sobre cómo se realizó la voz

### MODELOS DE DIVULGACIÓN IMPLÍCITOS

- Divulgación de funcionalidad
- Indicaciones implícitas y retroalimentación
- Transparencia de conversación

Use el siguiente gráfico para hacer referencia directamente a los modelos que se aplican a la voz sintética. Algunas de las demás condiciones de este gráfico también pueden aplicarse a su escenario:

SI SU EXPERIENCIA DE VOZ SINTÉTICA...	RECOMENDACIONES	PATRONES DE DISEÑO
Requiere un nivel de divulgación alto	Use al menos un modelo explícito e indicaciones implícitas por adelantado para ayudar a los usuarios a crear asociaciones.	Divulgación explícita Divulgación implícita
Requiere un nivel de divulgación bajo	La divulgación puede ser mínima o innecesaria, pero podría beneficiarse de algunos modelos implícitos.	Divulgación de funcionalidad Transparencia de conversación
Tiene un alto nivel de interacción	Compile a largo plazo y ofrezca varios puntos de entrada para la divulgación a lo largo del recorrido del usuario. Se recomienda encarecidamente tener una experiencia de incorporación.	Introducción transparente Personalización y calibración Divulgación de funcionalidad

SI SU EXPERIENCIA DE VOZ SINTÉTICA...	RECOMENDACIONES	PATRONES DE DISEÑO
Incluye a los niños como el público principal previsto	Diríjase a los padres como público de divulgación principal y asegúrese de que pueden comunicarse de forma eficaz con los niños.	Divulgación parental Introducción transparente verbal Divulgación implícita Transparencia de conversación
Incluye a usuarios ciegos o personas con poca visión como el público principal previsto	Debe incluir a todos los usuarios y asegurarse de que cualquier forma de divulgación visual tenga un texto alternativo o efectos sonoros asociados. Cumpla los estándares de accesibilidad para la relación de contraste y el tamaño de la pantalla. Utilice indicaciones auditivas para comunicar la divulgación.	Introducción transparente verbal Indicaciones auditivas Indicaciones hapticas Transparencia de conversación Estándares de accesibilidad
No tiene pantalla, no tiene dispositivo o usa la voz como el modo principal o único de interacción.	Utilice indicaciones auditivas para comunicar la divulgación.	Introducción transparente verbal Indicaciones auditivas
Possiblemente incluye varios usuarios o agentes de escucha (por ejemplo, asistente personal en varios hogares).	Tenga en cuenta los distintos contextos de usuario y niveles de comprensión y ofrezca varias oportunidades para la divulgación en el recorrido del usuario.	Introducción transparente (usuario habitual) Oportunidades para obtener más información sobre cómo se realizó la voz Transparencia de conversación

## Divulgación explícita

Si la experiencia de voz sintética requiere un nivel de divulgación elevado, es mejor usar al menos uno de los siguientes modelos explícitos para indicar claramente la naturaleza sintética de la voz.

### Introducción transparente

Antes de que comience la experiencia de voz, introduzca el asistente digital; sea totalmente transparente sobre los orígenes de la voz y sus funcionalidades. El momento óptimo para usar este modelo es cuando se incorpora un nuevo usuario o cuando se introducen nuevas características para un usuario habitual. La implementación de indicaciones implícitas durante una introducción ayuda a los usuarios a formar un modelo mental sobre la naturaleza sintética del agente digital.

### Experiencia de usuario nuevo



Meet your new digital assistant, Oso\*. Oso can help you search movies & shows, look up the weather, search the internet, and more.

\*Oso uses synthetic voice technology. [Learn more](#)

[Continue](#)

[Set up later in settings](#)

*La voz sintética se introduce durante la incorporación de un nuevo usuario.*

#### Recomendaciones

- Describa que la voz es artificial (por ejemplo, "digital")
- Describa lo que el agente es capaz de hacer.
- Indique explícitamente los orígenes de la voz.
- Ofrezca un punto de entrada para obtener más información acerca de la voz sintética.

#### Experiencia del usuario habitual

Si un usuario omite la experiencia de incorporación, siga ofreciendo puntos de entrada a la experiencia de introducción transparente hasta que el usuario active la voz por primera vez.



Meet Oso, your new digital voice assistant



Search movies, tv shows, and more.



Meet your new voice assistant, Oso\*. You can control your TV with Oso and get things done just by asking. Oso can help you search movies, shows, look up the weather, search the internet, and more.

\*Oso uses synthetic voice technology.



Try saying...

Continue

Learn more

Set up later

*Proporcione un punto de entrada coherente con la experiencia de voz sintética. Permita que el usuario vuelva a la experiencia de incorporación cuando desencadene la voz por primera vez en cualquier momento del recorrido del usuario.*

#### Introducción transparente verbal

Un aviso hablado que indica los orígenes de la voz del asistente digital es lo suficientemente explícito como para lograr la divulgación. Este modelo es el mejor para escenarios con un nivel de divulgación elevado, donde la voz es el único modo de interacción disponible.

*"This audiobook was synthetically generated using samples of the author's voice."*

Use una introducción transparente cuando haya momentos en la experiencia del usuario en los que ya pueda introducir o atribuir la voz de una persona.

*"Hi, this is Melody Stewart. This audiobook was synthetically generated using samples of my voice. Click on the description to learn more."*

Para una transparencia adicional, el actor de voz puede revelar los orígenes de la voz sintética en primera persona.

#### Firma explícita

Use este modelo si el usuario va a interactuar con un reproductor de audio o un componente interactivo para desencadenar la voz.



Una firma explícita es la atribución de la procedencia de la voz.

#### Recomendaciones

- Ofrezca un punto de entrada para obtener más información acerca de la voz sintética.

#### Personalización y calibración

Proporcione a los usuarios control sobre el modo en que el asistente digital responde a ellos (es decir, cómo suena la voz). Cuando un usuario interactúa con un sistema en sus propios términos y con objetivos específicos en mente, por definición, ya ha entendido que no se trata de una persona real.

#### Control de usuario

Ofrezca opciones que tengan un impacto significativo y perceptible en la experiencia de voz sintética.

## Preferences



Sarah  
[View profile](#)

### ASSISTANT SETTINGS

- |                   |  |
|-------------------|--|
| Language          |  |
| Assistant voice   |  |
| Voice calibration |  |
| Custom Commands   |  |
| Reminders         |  |

### YOUR DATA

- |                        |  |
|------------------------|--|
| Assistant Activity     |  |
| Voice & Audio Activity |  |
| Device Information     |  |

*Las preferencias de usuario permiten a los usuarios personalizar y mejorar su experiencia.*

#### Recomendaciones

- Permite a los usuarios personalizar la voz (por ejemplo, seleccionar el idioma y el tipo de voz).
- Proporcione a los usuarios una forma de enseñar al sistema a responder a su voz única (por ejemplo, calibración de voz, comandos personalizados).
- Optimice las interacciones contextuales o generadas por el usuario (por ejemplo, recordatorios).

#### Personalización de roles

Ofrezca maneras de personalizar la voz del asistente digital. Si la voz está basada en una persona famosa o es ampliamente reconocible, considere la posibilidad de usar las introducciones visuales y orales cuando los usuarios obtengan una vista previa de la voz.

## Assistant Voice

Select the voice your digital assistant will use to respond to you



### Celebrity Voice **Melody Stewart**

This voice is synthetically generated using recordings of Melody Stewart's voice.

[Learn more](#)

[Continue](#)

*Offering the ability to select from a set of voices helps convey the artificial nature.*

### Recomendaciones

- Permita a los usuarios obtener una vista previa del sonido de cada voz.
- Use una introducción auténtica para cada voz.
- Ofrezca puntos de entrada para obtener más información acerca de la voz sintética.

### Divulgación parental

Además de cumplir con las regulaciones de COPPA, proporcione divulgación a los padres si el público previsto principal es joven y el nivel de exposición es alto. En el caso de los usos confidenciales, considere la posibilidad de restringir el acceso a la experiencia hasta que un adulto haya confirmado el uso de la voz sintética. Anime a los padres a que comuniquen el mensaje a sus hijos.



## Welcome!

Meet your new digital teacher, Oso!\*

### For parents

To continue, read the following statement and answer the question below:

\*Oso's voice is synthetically created using a real teacher's voice. Please help your child understand that a real person is not speaking.

[Learn more](#)

What is:

$$7 \times 5 = \square$$

[Confirm](#)

*Una introducción transparente optimizada para los padres garantiza que un adulto sea consciente de la naturaleza sintética de la voz antes de que un niño interactúe con ella.*

Recomendaciones

- Diríjase a los padres como público principal para la divulgación.
- Anime a los padres a comunicar la divulgación a sus hijos.
- Ofrezca puntos de entrada para obtener más información acerca de la voz sintética.
- Controle el acceso a la experiencia al solicitar a los padres una sencilla pregunta "de seguridad" para mostrar que han leído la divulgación.

### Oportunidades para obtener más información sobre cómo se realizó la voz

Ofrezca puntos de entrada contextuales a una página, un elemento emergente o un sitio externo que proporcione más información acerca de la tecnología de voz sintética. Por ejemplo, podría mostrar un vínculo para obtener más información durante la incorporación o cuando el usuario solicite más información durante la conversación.



Hi, I'm Oso, your digital voice assistant

\*Oso uses synthetic voice technology. [Learn more](#)

*Ejemplo de un punto de entrada para ofrecer la oportunidad de obtener más información sobre la voz sintetizada.*

Una vez que un usuario solicita más información sobre la voz sintética, el objetivo principal es educarlo sobre los orígenes de la voz sintética y ser transparente sobre la tecnología.

# Help & Feedback

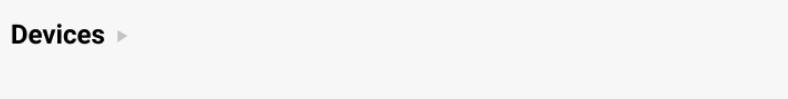
About Oso  
FAQ  
Legal  
Contact us

**Features ▶**

**Synthetic voice technology ▾**



**Devices ▶**



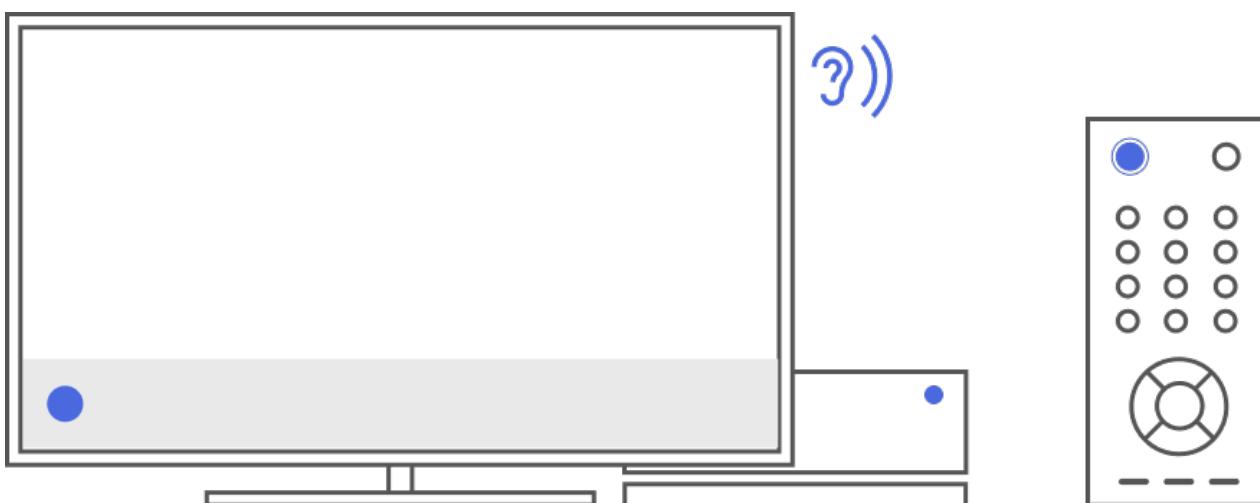
*Se puede ofrecer más información en el sitio de ayuda de un sitio externo.*

#### Recomendaciones

- Simplifique los conceptos complejos y evite el uso de jerga jurídica y técnica.
- No oculte este contenido en declaraciones de privacidad y condiciones de uso.
- Mantenga el contenido conciso y use imágenes cuando estén disponibles.

## Divulgación implícita

La coherencia es la clave para lograr la divulgación implícitamente a lo largo del recorrido del usuario. El uso coherente de indicaciones visuales y auditivas en los dispositivos y modos de interacción puede ayudar a crear asociaciones entre los modelos implícitos y la divulgación explícita.



#### Indicaciones implícitas y comentarios

El antropomorfismo puede manifestarse de maneras diferentes, desde la representación visual real del agente hasta la voz, los sonidos, los patrones de luz, el rebote de formas o incluso la vibración de un dispositivo. Al definir el rol, aproveche las indicaciones implícitas y los patrones de retroalimentación, en lugar de tener como objetivo un avatar de apariencia muy humana. Se trata de una manera de minimizar la necesidad de una divulgación más

explícita.



Estas guías ayudan a antropomorfizar al agente sin volverlo demasiado humano. También pueden convertirse en mecanismos de divulgación eficaces por su cuenta cuando se usan de forma coherente con el tiempo.

Tenga en cuenta los diferentes modos de interacciones de su experiencia al incorporar los siguientes tipos de indicaciones:

INDICACIONES VISUALES	INDICACIONES AUDITIVAS	INDICACIONES HÁPTICAS
<p>Avatar Indicaciones dinámicas en tiempo real (por ejemplo, animaciones). Indicaciones que no están en pantalla (por ejemplo, luces y patrones en un dispositivo).</p>	<p>Sonicon (por ejemplo, un breve sonido distintivo, una serie de notas musicales).</p>	Vibración

### Divulgación de funcionalidad

La divulgación se puede lograr de forma implícita al establecer expectativas precisas de lo que el asistente digital es capaz de usar. Proporcione comandos de ejemplo para que los usuarios puedan aprender cómo interactuar con el asistente digital y ofrezca ayuda contextual para obtener más información sobre la voz sintética durante las primeras fases de la experiencia.

••• Try saying... [Search for funny shows](#) [Play contemporary jazz music](#) [Learn more](#) [Change settings](#)

### Transparencia de conversación

Cuando las conversaciones se encuentren en rutas de acceso inesperadas, considere la posibilidad de crear respuestas predeterminadas que puedan ayudar a restablecer las expectativas, reforzar la transparencia y dirigir a los usuarios hacia las rutas correctas. También hay oportunidades para usar la divulgación explícita en la conversación.



Can you help me fix my cable?



Sorry, I can't help you with that, but perhaps a real human can. Would you like me to connect you to customer service?



Change my payment method



I haven't been programmed to do that yet, but you can try asking me these things:

Search for funny shows

Play contemporary jazz

Las preguntas que no están relacionadas o son "personales" dirigidas al agente son un buen momento para recordar a los usuarios la naturaleza sintética del agente y dirigirlos para que interactúen con ella adecuadamente o para transferirlos con una persona real.



Are you human?



No. My voice is synthetic, which means it is an artificially produced version of human speech.



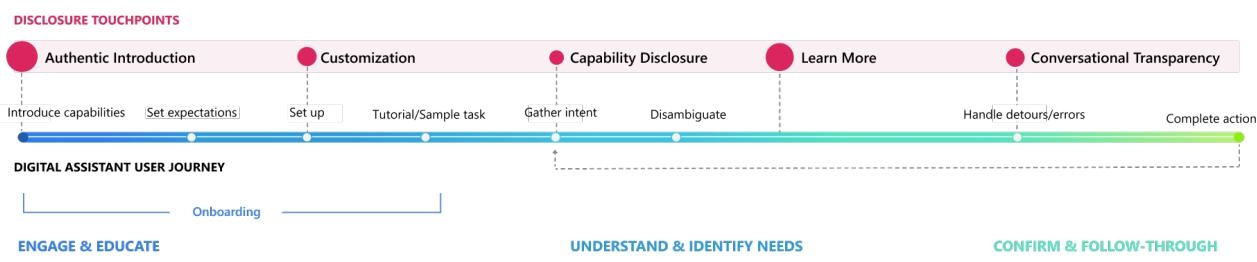
Do you have a soul?



I'll have to ask the engineers who built me...

## Momento de divulgación

Hay muchas oportunidades de divulgación a lo largo del recorrido del usuario. Diseñe para el primer uso, el segundo, el número n..., pero también acepte los "errores" para resaltar la transparencia. Por ejemplo, cuando el sistema comete un error o cuando el usuario detecta una limitación en las funcionalidades del agente.



Ejemplo de un recorrido de usuario estándar de asistente digital que resalta varias oportunidades de divulgación.

### Por adelantado

El momento óptimo para la divulgación es la primera vez que una persona interactúa con la voz sintética. En un escenario de asistente de voz personal, esto ocurriría durante la incorporación o la primera vez que el usuario prácticamente desempaquetaría la experiencia. En otros escenarios, podría ser la primera vez que una voz sintética lee contenido en un sitio web o la primera vez que un usuario interactúa con un personaje virtual.

- [Introducción transparente](#)
- [Divulgación de funcionalidad](#)
- [Personalización y calibración](#)
- [Indicaciones implícitas](#)

### A petición

Los usuarios deben poder acceder fácilmente a información adicional, controlar las preferencias y recibir comunicación transparente en cualquier punto del recorrido del usuario cuando se solicite.

- [Oportunidades para obtener más información sobre cómo se realizó la voz](#)
- [Personalización y calibración](#)
- [Transparencia de conversación](#)

### Continuamente

Use los modelos de diseño implícitos que mejoran la experiencia del usuario continuamente.

- [Divulgación de funcionalidad](#)
- [Indicaciones implícitas](#)

### Cuando se produce un error en el sistema

Use la divulgación como una oportunidad para que se produzcan errores con gracia.

- [Transparencia de conversación](#)
- [Oportunidades para obtener más información sobre cómo se realizó la voz](#)
- [Entrega a personas](#)

## Recursos adicionales

- [Directrices de bot de Microsoft](#)
- [Directrices sobre el diseño de Cortana](#)
- [Directrices para el diseño de voz de UWP de Microsoft Windows](#)
- [Directrices de comandos de voz de Microsoft Windows Mixed Reality](#)

## Documentos de referencia

- [Divulgación de talento de voz](#)
- [Directrices para la implementación responsable de la tecnología de voz sintética](#)
- [Información general sobre control](#)

- Procedimiento de divulgación

## Pasos siguientes

- Divulgación de talento de voz

# Introducción a voz personalizada

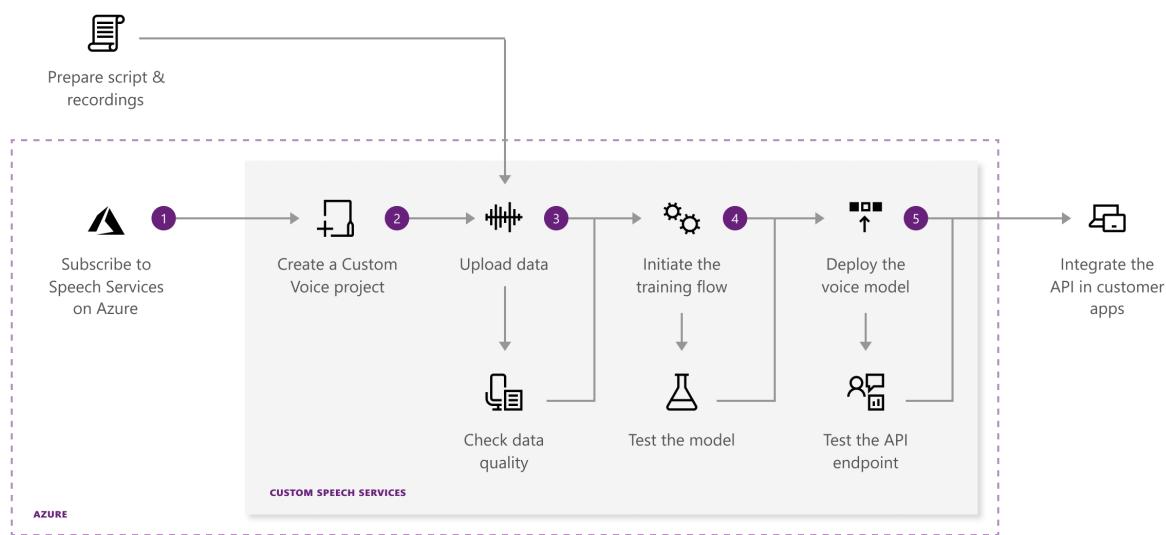
13/01/2020 • 6 minutes to read • [Edit Online](#)

Custom Voice es un conjunto de herramientas en línea que permiten crear una voz única y reconocible para su marca. Todo lo que se necesita para empezar son unos cuantos archivos de audio y las transcripciones asociadas. Siga los vínculos que se incluyen a continuación para empezar a crear una experiencia personalizada de conversión de texto a voz.

## ¿Qué incluye Custom Voice?

Antes de comenzar con Custom Voice, necesitará una cuenta de Azure y una suscripción del servicio de voz. Cuando haya creado una cuenta, podrá preparar los datos, entrenar y probar los modelos, evaluar la calidad de la voz y, finalmente, implementar el modelo de voz personalizado.

En el diagrama siguiente se resaltan los pasos necesarios para crear un modelo de voz personalizado desde el [portal de Custom Voice](#). Siga los vínculos para obtener más información.



1. **Suscríbase y cree un proyecto:** cree una cuenta de Azure y una suscripción del servicio de voz. Esta suscripción unificada proporciona acceso a la conversión de voz a texto, la conversión de texto a voz, la traducción de voz y el portal de Custom Voice. A continuación, mediante la suscripción al servicio de voz, cree su primer proyecto de Voz personalizada.
2. **Cargar datos:** carga de datos (audio y texto) mediante el portal de Custom Voice o una API de voz personalizada. Desde el portal, puede investigar y evaluar las puntuaciones de pronunciación y las relaciones de señal a ruido. Para más información, consulte el artículo sobre la [preparación de datos para Custom Voice](#).
3. **Entrenar el modelo:** utilice los datos para crear un modelo de voz de conversión de texto a voz personalizada. Puede entrenar un modelo en diferentes idiomas. Después del entrenamiento, pruebe el modelo y, si le satisface el resultado, ya puede implementar el modelo.
4. **Implementar el modelo:** cree un punto de conexión personalizado para el modelo de voz de conversión de texto a voz y utilícelo para la síntesis de voz en sus productos, herramientas y aplicaciones.

## Voces neuronales personalizadas

La funcionalidad de personalización de voz neuronal se encuentra actualmente en versión preliminar, limitada a clientes seleccionados. Rellene este [formulario de aplicación](#) para empezar.

#### NOTE

Como parte del compromiso de Microsoft de diseñar inteligencia artificial responsable, nuestra intención es proteger los derechos de los individuos y la sociedad, y fomentar unas interacciones transparentes de personas y equipos. Por esta razón, Voz neuronal personalizada no está disponible con carácter general para todos los clientes. Es posible que obtenga acceso a la tecnología solo tras la revisión de sus aplicaciones y después de que se haya comprometido a usarla en consonancia con nuestros principios éticos. Obtenga más información sobre nuestro [proceso de acceso según la aplicación](#).

## Configuración de la cuenta de Azure

Para poder usar el portal de Custom Speech y crear un modelo personalizado, se necesita una suscripción al servicio de voz. Siga estas instrucciones para crear una suscripción al servicio de voz en Azure. Si no tiene una cuenta de Azure, puede registrarse para obtener una nueva.

Después de crear una cuenta de Azure y la suscripción al servicio de voz, deberá iniciar sesión en el portal de Voz personalizada y conectarse a su suscripción.

1. Obtenga la clave de la suscripción del servicio de voz en Azure Portal.
2. Inicie sesión en el [portal de Custom Voice](#).
3. Seleccione la suscripción y cree un proyecto de voz.
4. Si desea cambiar a otra suscripción de voz, utilice el icono de engranaje situado en el panel de navegación superior.

#### NOTE

El servicio Custom Voice no admite la clave de evaluación gratuita de 30 días. Debe tener una clave F0 o S0 creada en Azure para poder usar el servicio.

## Creación de un proyecto

El contenido como datos, modelos, pruebas y puntos finales se organizan en **proyectos** en el portal de Custom Voice. Cada proyecto es específico de un idioma o país y del género de la voz que desea crear. Por ejemplo, puede crear un proyecto de voz femenina para los bots de chat del centro de llamadas que utilizan el inglés de Estados Unidos (en-US).

Para crear su primer proyecto, seleccione la pestaña **Text-to-Speech/Custom Voice** (Conversión de texto a voz/Conversión de voz personalizada) y, después, haga clic en **New Project** (Nuevo proyecto). Siga las instrucciones del asistente para crear el proyecto. Después de crear el proyecto, verá cuatro pestañas: **Data** (Datos), **Training** (Entrenamiento), **Testing** (Pruebas) y **Deployment** (Implementación). Use los vínculos incluidos en [Pasos siguientes](#) para aprender a usar cada pestaña.

## Pasos siguientes

- [Preparación de datos de voz personalizado](#)
- [Creación de una voz personalizada](#)
- [Guía: Grabación de muestras de voz](#)

# Preparación de los datos para crear una voz personalizada

13/01/2020 • 17 minutes to read • [Edit Online](#)

Cuando esté listo para crear un modelo personalizado de texto a voz para su aplicación, el primer paso es reunir las grabaciones de audio y los scripts asociados para empezar a entrenar el modelo de voz. El servicio de voz usan estos datos para crear una voz única optimizada para que coincida con la de las grabaciones. Cuando haya entrenado la voz, puede comenzar a sintetizarla en sus aplicaciones.

Puede comenzar con una pequeña cantidad de datos para crear una prueba de concepto. Sin embargo, cuantos más datos proporcione, más natural sonará su voz personalizada. Antes de entrenar su propio modelo de texto a voz, necesitará las grabaciones de audio y las transcripciones de texto asociadas. En esta página, revisaremos los tipos de datos, cómo se usan y cómo administrar cada uno.

## Tipos de datos

Un conjunto de datos de entrenamiento de voz incluye grabaciones de audio y un archivo de texto con las transcripciones asociadas. Cada archivo de audio debe contener una sola expresión (una frase única o un solo turno en un sistema de diálogo) y tener una duración de menos de 15 segundos.

En algunos casos, puede que no tenga listo el conjunto de datos adecuado y quiera probar el entrenamiento de voz personalizada con los archivos de audio disponibles, cortos o largos, con o sin transcripciones. Nosotros proporcionamos herramientas (beta) para ayudarle a segmentar el audio en expresiones y preparar las transcripciones mediante la [API Batch Transcription](#).

En esta tabla se enumeran los tipos de datos y cómo se usa cada uno para crear un modelo personalizado de texto a voz.

TIPO DE DATOS	DESCRIPCIÓN	CUÁNDO SE DEBEN USAR	SERVICIO ADICIONAL NECESARIO	CANTIDAD PARA ENTRENAR UN MODELO	CONFIGURACIONES REGIONALES
<b>Expresiones individuales + transcripción relacionada</b>	Una colección (.zip) de archivos de audio (.wav) como expresiones individuales. Cada archivo de audio debe tener una longitud de 15 segundos o menos y estar emparejado con una transcripción con formato (.txt).	Grabaciones profesionales con transcripciones relacionadas	Listo para el entrenamiento.	Sin requisitos fijos para en-US y zh-CN. Más de 2000 expresiones diferentes para otras configuraciones regionales.	Todas las configuraciones regionales de voz personalizada

TIPO DE DATOS	DESCRIPCIÓN	CUÁNDO SE DEBEN USAR	SERVICIO ADICIONAL NECESARIO	CANTIDAD PARA ENTRENAR UN MODELO	CONFIGURACIONES REGIONALES
<b>Audio largo + transcripciones (beta)</b>	Una colección (.zip) de archivos de audio largos sin segmentar (más de 20 segundos), emparejados con una transcripción (.txt) que contiene todas las palabras habladas.	Tiene archivos de audio y transcripciones relacionadas, pero no están segmentados en expresiones.	Segmentación (mediante transcripción por lotes). Transformación del formato de audio cuando sea necesario.	Sin requisitos fijos	Todas las configuraciones regionales de voz personalizada
<b>Solo audio (beta)</b>	Una colección (.zip) de archivos de audio sin una transcripción.	Solo dispone de archivos de audio, sin transcripciones.	Segmentación + generación de transcripciones (mediante la transcripción por lotes). Transformación del formato de audio cuando sea necesario.	Sin requisitos fijos	Todas las configuraciones regionales de voz personalizada

Los archivos deben agruparse por tipo en un conjunto de datos y cargarse como un archivo zip. Cada conjunto de datos solo puede contener un tipo de datos.

#### NOTE

El número máximo de conjuntos de datos que se pueden importar por suscripción es de 10 archivos ZIP para usuarios de la suscripción gratuita (F0) y 500 para usuarios para la suscripción estándar (S0).

## Expresiones individuales + transcripción relacionada

Puede preparar las grabaciones de expresiones individuales y la transcripción relacionada de dos maneras. Escriba un guion y haga que lo lea un locutor, o bien use el audio disponible públicamente y transcríbalo a texto. En este último caso, deberá editar las disfluencias de los archivos de audio, como las muletillas ("em") y otros sonidos de relleno, tartamudeos, palabras entre dientes o pronunciaciones erróneas.

Para crear una buena fuente de voz, realice las grabaciones en una sala silenciosa con un micrófono de alta calidad. El volumen constante, la velocidad de la conversación, el tono al hablar y las particularidades expresivas del habla son esenciales.

#### TIP

Para crear una voz que se vaya a usar en una producción, le recomendamos que use un estudio de grabación profesional y un locutor. Para obtener más información, consulte [Cómo grabar ejemplos de voz para una voz personalizada](#).

### Archivos de audio

Cada archivo de audio debe contener una sola expresión (una sola frase o un solo turno de un sistema de diálogo) y tener una duración inferior a 15 segundos. Todos los archivos deben estar en el mismo idioma hablado. La transformación de texto a voz personalizada en varios idiomas no se admite, excepto en el caso del chino al inglés bilingüe. Los archivos de audio deben tener un nombre de archivo numérico exclusivo con la

extensión de nombre de archivo .wav.

Al preparar el audio, siga estas directrices.

PROPIEDAD	VALOR
Formato de archivo	RIFF (.wav), agrupado en un archivo ZIP
Frecuencia de muestreo	Al menos 16 000 Hz
Formato de ejemplo	PCM, 16 bits
Nombre de archivo	Numérico, con la extensión. wav. No se permiten nombres de archivo duplicados.
Longitud de audio	Menor de 15 segundos
Formato de archivo	.zip
Tamaño de archivo máximo	2048 MB

#### NOTE

Se rechazarán los archivos .wav con una frecuencia de muestreo inferior a 16 000 Hz. Si un archivo ZIP contiene archivos .wav con distintas frecuencias de muestreo, solo se importarán las que sean iguales o superiores a 16 000 Hz. Actualmente el portal importa archivos .zip de hasta 200 MB. Sin embargo, pueden cargarse varios archivos.

## Transcripciones

El archivo de transcripción es un archivo de texto sin formato. Use estas directrices para preparar sus transcripciones.

PROPIEDAD	VALOR
Formato de archivo	Texto sin formato (.txt)
Formato de codificación	ANSI/ASCII, UTF-8, UTF-8-BOM, UTF-16-LE o UTF-16-BE. Con zh-CN, no se admiten las codificaciones ANSI/ASCII y UTF-8.
Número de expresiones por línea	<b>Una:</b> cada línea del archivo de transcripción debe contener el nombre de uno de los archivos de audio, seguido de la transcripción correspondiente. El nombre de archivo y la transcripción deben estar separados por un carácter de tabulación (\t).
Tamaño de archivo máximo	2048 MB

Este es un ejemplo de cómo las transcripciones se organizan en expresiones (de una en una) en un archivo txt:

```
000000001[tab] This is the waistline, and it's falling.  
000000002[tab] We have trouble scoring.  
000000003[tab] It was Janet Maslin.
```

Es importante que las transcripciones tengan una precisión del 100 % respecto al audio correspondiente. Los

errores en las transcripciones darán lugar a la pérdida de calidad durante el entrenamiento.

#### TIP

Al compilar las transformaciones de texto a voz en producción, seleccione aquellas expresiones (o scripts de escritura) que tengan en cuenta tanto la cobertura fonética como la eficiencia. ¿Tiene problemas para obtener los resultados que desea? [Póngase en contacto con el equipo de voz personalizada](#) para averiguar más sobre nuestro asesoramiento.

## Audio largo + transcripciones (beta)

En algunos casos, puede que no disponga de audio segmentado. Nosotros proporcionamos un servicio (beta) a través del portal de voz personalizada para ayudarle a segmentar los archivos de audio largos y crear transcripciones. Tenga en cuenta que este servicio se cobrará en función de su uso de la suscripción de voz a texto.

#### NOTE

El servicio de segmentación de audio largo aprovechará la característica de transcripción de voz a texto por lotes, que solo admite usuarios de la suscripción estándar (S0). Durante el procesamiento de la segmentación, los archivos de audio y las transcripciones también se enviarán al servicio Custom Speech para refinar el modelo de reconocimiento y así pueda mejorar la precisión de los datos. Durante este proceso no se conserva ningún dato. Después de realizar la segmentación, solo las expresiones segmentadas y sus transcripciones de asignación se almacenarán para su descarga y entrenamiento.

### Archivos de audio

Al preparar el audio para la segmentación, siga estas directrices.

PROPIEDAD	VALOR
Formato de archivo	RIFF (.wav) con una frecuencia de muestreo de al menos 16 khz y 16 bits en PCM o .mp3 con una velocidad de bits de al menos 256 Kbps, agrupado en un archivo ZIP
Nombre de archivo	Caracteres ASCII y Unicode admitidos. No se permiten nombres duplicados.
Longitud de audio	Más de 20 segundos
Formato de archivo	.zip
Tamaño de archivo máximo	2048 MB

Todos los archivos de audio se deben agrupar en un archivo ZIP. Puede poner archivos .wav y .mp3 en un archivo ZIP de audio. Por ejemplo, puede cargar un archivo ZIP que contenga un archivo de audio llamado "kingstory.wav", que dure 45 segundos, y otro llamado "queenstory.mp3", que dure 200 segundos. Todos los archivos. mp3 se transformarán al formato .wav después del procesamiento.

### Transcripciones

Las transcripciones deben estar preparadas de acuerdo con las especificaciones enumeradas en esta tabla. Cada archivo de audio debe coincidir con una transcripción.

PROPIEDAD	VALOR
Formato de archivo	Texto sin formato (.txt), agrupado en un archivo ZIP

PROPIEDAD	VALOR
Nombre de archivo	Use el mismo nombre que el archivo de audio relacionado.
Formato de codificación	Solo UTF-8-BOM
Número de expresiones por línea	Sin límite
Tamaño de archivo máximo	2048 MB

Todos los archivos de transcripciones de este tipo de datos deben estar agrupados en un archivo ZIP. Por ejemplo, ha cargado un archivo ZIP que contiene un archivo de audio llamado "kingstory.wav", que dura 45 segundos, y otro llamado "queenstory.mp3", que dura 200 segundos. Deberá cargar otro archivo ZIP que contenga dos transcripciones, una llamada "kingstory.txt" y la otra "queenstory.txt". Dentro de cada archivo de texto sin formato, proporcionará la transcripción completa correcta para el audio relacionado.

Después de que el conjunto de datos se ha cargado correctamente, le ayudaremos a segmentar el archivo de audio en expresiones según la transcripción proporcionada. Para comprobar las expresiones segmentadas y las transcripciones relacionadas, descargue el conjunto de datos. Se asignarán identificadores únicos a las expresiones segmentadas automáticamente. Es importante asegurarse de que las transcripciones que proporciona tengan una precisión del 100 %. Los errores en las transcripciones pueden reducir la precisión durante la segmentación del audio e introducir además pérdida de calidad en la fase de entrenamiento que viene más adelante.

## Solo audio (beta)

Si no tiene transcripciones para las grabaciones de audio, use la opción **Solo audio** para cargar los datos. Nuestro sistema puede ayudarlo a segmentar y transcribir los archivos de audio. Tenga en cuenta que este servicio se tendrá en cuenta en su uso de la suscripción de voz a texto.

Al preparar el audio, siga estas directrices.

### NOTE

El servicio de segmentación de audio largo aprovechará la característica de transcripción de voz a texto por lotes, que solo admite usuarios de la suscripción estándar (S0).

PROPIEDAD	VALOR
Formato de archivo	RIFF (.wav) con una frecuencia de muestreo de al menos 16 khz y 16 bits en PCM o .mp3 con una velocidad de bits de al menos 256 Kbps, agrupado en un archivo ZIP
Nombre de archivo	Caracteres ASCII y Unicode admitidos. No se permiten nombres duplicados.
Longitud de audio	Más de 20 segundos
Formato de archivo	.zip
Tamaño de archivo máximo	2048 MB

Todos los archivos de audio se deben agrupar en un archivo ZIP. Una vez que el conjunto de datos se ha cargado

correctamente, le ayudaremos a segmentar el archivo de audio en expresiones en función de nuestro servicio de transcripción de voz por lotes. Se asignarán identificadores únicos a las expresiones segmentadas automáticamente. Las transcripciones relacionadas se generarán mediante el reconocimiento de voz. Todos los archivos. mp3 se transformarán al formato .wav después del procesamiento. Para comprobar las expresiones segmentadas y las transcripciones relacionadas, descargue el conjunto de datos.

## Pasos siguientes

- [Creación de una voz personalizada](#)
- [Guía: Grabación de muestras de voz](#)

# Creación de una voz personalizada

13/01/2020 • 19 minutes to read • [Edit Online](#)

En [Preparar los datos para crear una voz personalizada](#), se describen los tipos de datos diferentes que puede usar para entrenar una voz personalizada y los distintos requisitos de formato. Cuando haya preparado los datos, puede empezar a cargarlos en el [portal de voz personalizada](#), o a través de la API de entrenamiento de voz personalizada. Aquí se describen los pasos del entrenamiento de una voz personalizada mediante el portal.

## NOTE

En esta página se supone que ha leído [Empezar a trabajar con la voz personalizada](#) y [Preparar los datos para crear una voz personalizada](#) y que ha creado un proyecto de voz personalizada.

Compruebe los idiomas admitidos para la voz personalizada: [idioma para la personalización](#).

## Cargar los conjuntos de datos

Cuando esté listo para cargar los datos, vaya al [portal de voz personalizada](#). Cree o seleccione un proyecto de voz personalizada. El proyecto debe compartir el idioma/configuración regional y las propiedades de género correctos con los datos que pretende usar para el entrenamiento de voz. Por ejemplo, seleccione `en-GB` si las grabaciones de audio se han realizado en inglés con acento británico.

Vaya a la pestaña **Data** (Datos) y haga clic en **Upload data** (Cargar datos). En el asistente, seleccione el tipo de datos correcto que coincida con lo que ha preparado.

Cada conjunto de datos que cargue debe cumplir los requisitos del tipo de datos elegido. Es importante dar formato correctamente a los datos antes de cargarlos. Esto garantiza que el servicio de voz personalizada procesa los datos con precisión. Vaya a [Preparar los datos para crear una voz personalizada](#) y asegúrese de que los datos tienen el formato correcto.

## NOTE

Los usuarios de suscripción gratuita (F0) pueden cargar dos conjuntos de datos al mismo tiempo. En cambio, los usuarios con una suscripción estándar (S0) pueden cargar cinco conjuntos de datos a la vez. Si alcanza el límite, espere hasta que al menos uno de los conjuntos de datos finalice la importación. A continuación, inténtelo de nuevo.

## NOTE

El número máximo de conjuntos de datos que se pueden importar por suscripción es de 10 archivos ZIP para usuarios de la suscripción gratuita (F0) y 500 para usuarios de la suscripción estándar (S0).

Los conjuntos de datos se validan automáticamente una vez que pulsa el botón de carga. La validación de datos incluye una serie de comprobaciones en los archivos de audio para comprobar su formato de archivo, el tamaño y la frecuencia de muestreo. Corrija los errores si los hay y vuelva a realizarla. Cuando se inicia correctamente la solicitud de importación de datos, debería ver una entrada en la tabla de datos que se corresponde con el conjunto de datos que acaba de cargar.

En la siguiente tabla se muestran los estados de procesamiento de los conjuntos de datos importados:

STATE	SIGNIFICADO
Processing	El conjunto de datos se ha recibido y se está procesando.
Succeeded	El conjunto de datos se ha validado y se puede usar para compilar un modelo de voz.
Con error	El conjunto de datos ha dado error durante el procesamiento debido a muchas razones, por ejemplo, errores de archivo, problemas de datos o problemas de red.

Una vez completada la validación, puede ver el número total de expresiones coincidentes para cada uno de los conjuntos de datos en la columna **Utterances** (Expresiones). Si el tipo de datos que ha seleccionado requiere una segmentación de audio de larga duración, esta columna solo refleja las expresiones que se han segmentado automáticamente en función de las transcripciones o mediante el servicio de transcripción de voz. Puede seguir descargando el conjunto de datos validado para ver los resultados detallados de las expresiones importadas correctamente y sus transcripciones de asignación. Sugerencia: la segmentación de audio de larga duración puede tardar más de una hora en completar el procesamiento de datos.

Para los conjuntos de datos de en-US y zh-CN, puede seguir descargando un informe para comprobar las puntuaciones de la pronunciación y el nivel de ruido de cada una de las grabaciones. La puntuación de las pronunciaciones abarca un rango del 0 al 100. Una puntuación por debajo de 70 normalmente indica un error en el discurso o que el guion no coincide. Un acento marcado puede reducir la puntuación de las pronunciaciones, y afectar a la voz digital que se ha creado.

Una relación de la señal y el ruido (SNR) más alta indica un ruido más bajo en el audio. Por lo general, el audio puede alcanzar una SNR de más de 50 puntos si se graba en estudios profesionales. Un audio que tenga un valor de SNR por debajo de 20 puntos puede provocar un ruido obvio en la voz generada.

Puede volver a grabar cualquier expresión que tenga una puntuación baja debido a la pronunciación o a la pobre relación entre el ruido y la señal. Si no es posible volver a realizar la grabación, puede excluir esas expresiones de su conjunto de datos.

## Compilación del modelo de voz personalizada

Una vez que el conjunto de datos se haya validado, podrá usarlo para compilar su modelo de voz personalizado.

1. Vaya a **Text-to-Speech > Custom Voice > Training** (Texto a voz > Voz personalizada > Entrenamiento).
2. Haga clic en **Train model** (Entrenar modelo).
3. A continuación, escriba un **nombre** y una **descripción** que le ayuden a identificar este modelo.

Elija un nombre con cuidado. El nombre que escriba aquí será el nombre que use para especificar la voz en su solicitud de síntesis de voz como parte de la entrada de SSML. Solo se permiten letras, números y algunos signos de puntuación, como -, \_ y (","). Use nombres diferentes para modelos de voz diferentes.

Un uso habitual del campo **Descripción** es registrar los nombres de los conjuntos de datos que se usaron para crear el modelo.

4. En la página **Select training data** (Seleccionar datos de entrenamiento), elija uno o varios conjuntos de datos que quiera usar para el entrenamiento. Compruebe el número de expresiones antes de enviarlas. Puede comenzar con cualquier número de expresiones para los modelos de voz de en-US y zh-CN. En el caso de otras configuraciones regionales, debe seleccionar más de 2000 expresiones para

poder entrenar una voz.

#### NOTE

Los nombres de audio duplicados se quitarán del entrenamiento. Asegúrese de que los conjuntos de datos que seleccione no contengan los mismos nombres de audio en varios archivos ZIP.

#### TIP

Para obtener unos resultados de calidad, es necesario usar los conjuntos de datos del mismo altavoz. Cuando los conjuntos de datos que ha enviado para el entrenamiento contengan un número total de menos de 6000 expresiones distintas, entrenará el modelo de voz mediante la técnica de síntesis paramétrica estadística. En el caso de que los datos de entrenamiento excedan un número total de 6000 expresiones diferentes, se iniciará un proceso de entrenamiento con la técnica de síntesis de concatenación. Normalmente la tecnología de concatenación puede producir resultados más naturales y aumenta la fidelidad de la voz. [Póngase en contacto con el equipo de voz personalizada](#) si quiere entrenar un modelo con la tecnología TTS neuronal más reciente, la cual puede producir una voz digital equivalente a las [voices neuronales](#) disponibles públicamente.

5. Haga clic en **Train** (Entrenar) para empezar a crear el modelo de voz.

En la tabla de entrenamiento se muestra una nueva entrada que corresponde a este modelo recién creado. En la tabla también se muestra el estado: Procesando, Correcto, Error.

El estado que se muestra refleja el proceso de convertir el conjunto de datos en un modelo de voz, como se muestra aquí.

STATE	SIGNIFICADO
Processing	Se está creando el modelo de voz.
Succeeded	El modelo de voz se ha creado y se puede implementar.
Con error	El modelo de voz ha dado error en el entrenamiento por muchas razones, por ejemplo, problemas desapercibidos con los datos o problemas de red.

El tiempo de aprendizaje varía según el volumen de datos de audio procesados. El intervalo de tiempo típicos varía, aproximadamente, desde los 30 minutos para unos cientos de expresiones, hasta 40 horas para 20.000 expresiones. Cuando finalice correctamente el entrenamiento del modelo, puede empezar a probarlo.

#### NOTE

Los usuarios con una suscripción gratuita (F0) pueden entrenar una fuente de voz al mismo tiempo. En cambio, los usuarios que tengan una suscripción estándar (S0) pueden entrenar tres voces simultáneamente. Si alcanza el límite, espere hasta que al menos una de las fuentes de voz finalice el aprendizaje e inténtelo de nuevo.

#### NOTE

El número máximo de modelos de voz que se pueden entrenar por suscripción es de 10 modelos para los usuarios con una suscripción gratuita (F0) y de 100 para los usuarios con una suscripción estándar (S0).

Si usa la funcionalidad de entrenamiento de voz neuronal, puede optar por entrenar un modelo optimizado para escenarios de streaming en tiempo real o un modelo neuronal de alta definición optimizado para la

síntesis de audio de larga duración de manera asincrónica.

## Prueba del modelo de voz

Una vez que la fuente de voz se haya creado correctamente, podrá probarla antes de implementarla para su uso.

1. Vaya a **Text-to-Speech > Custom Voice > Testing** (Texto a voz > Voz personalizada > Prueba).
2. Haga clic en **Add test** (Agregar prueba).
3. Seleccione uno o varios modelos que le gustaría probar.
4. Proporcione el texto que quiere que hable la voz o las voces. Si ha seleccionado probar varios modelos al mismo tiempo, se usará el mismo texto para las pruebas en distintos modelos.

### NOTE

Recuerde que el idioma del texto debe ser el mismo que el de la fuente de voz. Solo se pueden probar los modelos entrenados correctamente. En este paso, solo se admite texto sin formato.

5. Haga clic en **Create** (Crear).

Cuando haya enviado la solicitud de prueba, volverá a la página de prueba. La tabla ahora incluye una entrada que corresponde a su nueva solicitud y la columna de estado. Es posible que se tarden unos minutos en sintetizar la voz. Cuando la columna de estado muestre el valor **Succeeded** (Correcto), puede reproducir el audio, o descargar la entrada de texto (un archivo .txt) y la salida de audio (un archivo .wav) y escuchar esta última para comprobar la calidad.

También puede encontrar los resultados de la prueba en la página de detalles de cada modelo seleccionado para prueba. Vaya a la pestaña **Training** (Entrenamiento) y haga clic en el nombre del modelo para especificar su página de detalles.

## Creación y uso de un punto de conexión de voz personalizado

Una vez que haya creado y probado con éxito su modelo de voz, impleméntelo en un punto de conexión personalizado de Text to Speech. A continuación, use ese punto de conexión en lugar del punto de conexión habitual al realizar solicitudes de Text to Speech a través de la API de REST. Recuerde que solo se puede llamar al punto de conexión personalizado mediante la suscripción que utilizó para implementar la fuente.

Para crear un nuevo punto de conexión personalizado, vaya a **Text-to-Speech > Custom Voice > Deployment** (Texto a voz > Voz personalizada > Implementación). Seleccione **Add endpoint** (Agregar punto de conexión) y escriba un **nombre** y una **descripción** para el punto de conexión personalizado. A continuación, seleccione el modelo de voz personalizada que le gustaría asociar a este punto de conexión.

Después de haber hecho clic en el botón **Add** (Agregar), en la tabla de puntos de conexión, verá una entrada para el nuevo punto de conexión. El proceso para crear instancias del nuevo punto de conexión puede llevar unos minutos. Cuando el estado de la implementación muestra el valor **Completado**, quiere decir que el punto de conexión está listo para su uso.

### NOTE

Los usuarios con una suscripción gratuita (F0) solo pueden tener un modelo implementado. En cambio, los usuarios que tengan una suscripción estándar (S0) pueden crear hasta 50 puntos de conexión, cada uno de ellos con su propia voz personalizada.

#### **NOTE**

Para usar su voz personalizada, debe especificar el nombre del modelo de voz, utilizar el URI personalizado directamente en una solicitud HTTP y emplear la misma suscripción para pasar por la autenticación del servicio TTS.

Una vez implementado el punto de conexión, su nombre aparece como un vínculo. Haga clic en el vínculo para mostrar información específica del punto de conexión, como la clave o la dirección URL, y código de ejemplo.

Las pruebas en línea del punto de conexión también están disponibles a través del portal de voz personalizada. Para probar el punto de conexión, elija **Check endpoint** (Comprobar el punto de conexión) en la página **Endpoint detail** (Detalle de punto de conexión). Aparecerá la página para probar los puntos de conexión. Escriba el texto que se hablará (en texto sin formato o [formato SSML](#) en el cuadro de texto).

Seleccione **Reproducir** para escuchar el texto que se habla en su fuente de voz personalizada. Esta característica de pruebas se les cobrará con el uso de síntesis de voz personalizada.

El punto de conexión personalizado es técnicamente idéntico al punto de conexión estándar que se usa en las solicitudes de texto a voz. Consulte la [API REST](#) para obtener más información.

## Pasos siguientes

- [Guía: Grabar ejemplos de voz](#)
- [Referencia de Text-to-Speech API](#)
- [Long Audio API](#)

# Grabación de muestras de voz para crear una voz personalizada

13/01/2020 • 36 minutes to read • [Edit Online](#)

Crear una voz personalizada con calidad de producción partiendo de cero no es una tarea ocasional. El componente central de una voz personalizada es una gran colección de muestras de audio de voz humana. Es fundamental que estas grabaciones de audio sean de alta calidad. Elija un actor de voz que tenga experiencia en esta clase de grabaciones y un ingeniero de grabación competente que las grabe con un equipo profesional.

No obstante, para poder realizar estas grabaciones, necesita un guion: las palabras que dirá el actor de voz para crear las muestras de audio. Para obtener mejores resultados, el guion debe tener una buena cobertura fonética y variedad suficiente para entrenar el modelo de voz personalizada.

Muchos detalles pequeños pero importantes intervienen en la creación de una grabación de voz profesional. Esta guía es una guía básica de un proceso que le ayudará a obtener resultados buenos y uniformes.

## TIP

Para conseguir resultados de máxima calidad, considere la posibilidad de acudir a Microsoft para ayudar a desarrollar su voz personalizada. Microsoft posee una gran experiencia en producir voces de alta calidad para sus propios productos, como Cortana y Office.

## Roles de grabación de voz

En un proyecto de grabación de voz personalizada hay cuatro roles básicos:

ROLE	PROPÓSITO
Actor de voz	La voz de esta persona constituirá la base de la voz personalizada.
Ingeniero de grabación	Supervisa los aspectos técnicos de la grabación y usa el equipo de grabación.
Director	Prepara el guion y prepara la sesión del actor de voz.
Editor	Finaliza los archivos de audio y los prepara para la carga en el portal Custom Voice.

Un usuario individual puede desempeñar más de un rol. En esta guía se da por hecho que desempeñará principalmente el papel de director y que contratará un actor de voz y un ingeniero de grabación. Si quiere realizar las grabaciones usted mismo, en este artículo se incluye alguna información sobre el rol de ingeniero de grabación. El rol de editor no es necesario hasta después de la sesión, por lo que puede realizarlo el director o el ingeniero de grabación.

## Elección del actor de voz

Los actores con experiencia en voz en off o en poner voz a personajes serán unos buenos actores de voz personalizada. También pueden desempeñar bien este rol presentadores y locutores.

Elija un actor cuya voz natural le guste. Aunque se pueden crear voces de "personaje" únicas, para los actores de voz es mucho más difícil conseguir locuciones uniformes y el esfuerzo puede dar lugar a que fuercen la voz.

#### TIP

Por lo general, evite usar voces reconocibles para crear una voz personalizada, a menos que, desde luego, el objetivo sea producir la voz de un famoso. Las voces menos conocidas generan normalmente menos distracciones para los usuarios.

El único factor más importante para elegir el actor de voz es la uniformidad. Las grabaciones deben sonar como si se realizaran el mismo día en la misma sala. Puede aproximarse a este ideal con unas buenas prácticas y una buena ingeniería de grabación.

Su actor de voz es la otra mitad de la ecuación. Debe ser capaz de hablar a una velocidad, con un nivel de volumen y un tono uniformes. Es obligatorio una dicción clara. El actor también debe ser capaz de controlar estrictamente la variación de su tono, el efecto emocional y los manierismos del habla.

La grabación de muestras de voz personalizada puede ser más fatigoso que otros trabajos de voz. La mayoría de los actores de voz pueden grabar durante dos o tres horas al día. Limite las sesiones a tres o cuatro a la semana, con un día de descanso entre medias si es posible.

Las grabaciones realizadas para un modelo de voz deben ser emocionalmente neutras. Es decir, un enunciado triste no debe leerse de forma triste. El estado de ánimo se puede agregar a la voz sintetizada más adelante mediante controles de la prosodia. Trabaje con el actor de voz para desarrollar una "persona" que defina el sonido y el tono emocional general de la voz personalizada. En el proceso, identificará lo que suena "neutro" para esa persona.

Es posible que, por ejemplo, una persona tenga una personalidad animada por naturaleza. Por tanto, "su" voz puede transmitir una nota de optimismo incluso aunque hable de forma neutra. Sin embargo, tal rasgo de personalidad debe ser sutil y uniforme. Escuche las lecturas de voces existentes para hacerse una idea de lo que busca.

#### TIP

Por lo general, querrá quedarse con las grabaciones de voz que realice. Así que el actor de voz debe aceptar un contrato de trabajo a comisión para el proyecto.

## Creación de un guion

El punto de partida de cualquier sesión de grabación de voz personalizada es el guion, que contiene los enunciados que dirá el actor de voz. (El término "enunciados" abarca tanto oraciones completas como frases más cortas).

Los enunciados del guion pueden proceder de cualquier parte: ficción, no ficción, transcripciones de voces, informes de noticias y cualquier cosa disponible en formato impreso. Si quiere asegurarse de que su voz funcione bien con clases de palabras específicas (como terminología médica o jerga de programación), podría incluir oraciones de trabajos académicos o documentos técnicos. Para obtener una breve descripción de los posibles problemas legales, vea la sección "[Aspectos legales](#)". También puede escribir su propio texto.

Las expresiones no tienen que proceder de la misma fuente, o de la misma clase de fuente. Incluso no es necesario que tengan nada que ver unos con otros. Sin embargo, si va a usar frases hechas (por ejemplo, "Ha iniciado sesión correctamente") en la aplicación de voz, asegúrese de incluirlas en el guion. De esta forma, la voz personalizada tendrá una mejor oportunidad de pronunciar bien esas frases. Y si decide usar una

grabación en lugar de voz sintetizada, ya la tendrá en la misma voz.

Si bien la uniformidad es importante a la hora de elegir un actor de voz, la variedad es lo que distingue un buen guion. El guion debe incluir muchas palabras y frases diferentes con diversas longitudes de frase, estructuras y estados de ánimo. Cada sonido del idioma se debe representar varias veces y en varios contextos (lo que se conoce como *cobertura fonética*).

Además, el texto debe incorporar todas las formas en que puede representarse un sonido determinado al escribir, y colocar cada sonido en distintos lugares en las oraciones. Se deben incluir tanto oraciones afirmativas como preguntas y se deben leer con la entonación adecuada.

Es difícil escribir un guion que proporcione *suficientes* datos para permitir que el portal Custom Speech genere una buena voz. En la práctica, la manera más sencilla de crear un guion que logre una sólida cobertura fonética es incluir un gran número de muestras. Las voces estándar que proporciona Microsoft se han creado a partir de decenas de miles de expresiones. Deberá estar preparado para grabar unos cuantos miles de expresiones para generar una voz personalizada con calidad de producción.

Compruebe el guion con mucho cuidado para detectar posibles errores. Si es posible, que otra persona también lo revise. Cuando repase el guion con el actor de voz, es probable que detecte algunos errores más.

### Formato del guion

Puede escribir el guion en Microsoft Word. El guion se usará durante la sesión de grabación, así que puede prepararlo de forma que sea fácil trabajar con él. Cree de forma independiente el archivo de texto necesario para el portal Custom Voice.

Un formato de guion básico contiene tres columnas:

- El número del enunciado, empezando por el 1. La numeración permite que todas las personas del estudio puedan consultar una expresión concreta ("probemos de nuevo el número 356"). Puede usar la característica de numeración de párrafos de Word para numerar las filas de la tabla de forma automática.
- Una columna en blanco donde escribirá el número de toma o el código de tiempo de cada expresión para ayudarle a encontrarla en la grabación finalizada.
- El propio texto del enunciado.

### Custom Voice Script Session Date: 6/1/65 Voice Talent: A. Lincoln

	Time Code	Utterance
1		Four score and seven years ago our fathers brought forth on this continent a new nation, conceived in Liberty, and dedicated to the proposition that all men are created equal.
2		Now we are engaged in a great civil war, testing whether that nation, or any nation so conceived and so dedicated, can long endure.
3		We are met on a great battlefield of that war.
4		We have come to dedicate a portion of that field as a final resting place for those who here gave their lives that that nation might live.

#### NOTE

La mayoría de los estudios graban en segmentos cortos conocidos como *tomas*. Cada toma contiene normalmente de 10 a 24 expresiones. Basta con anotar el número de toma para después poder encontrar una expresión. Si graba en un estudio que prefiere realizar grabaciones más largas, querrá anotar entonces el código de tiempo. El estudio tendrá una gran pantalla de tiempo.

Deje suficiente espacio después de cada fila para escribir notas. Asegúrese de que ningún enunciado se divida entre páginas. Numere las páginas e imprima el guion en una de las caras del papel.

Imprima tres copias del guion: una para el actor de voz, otra para el ingeniero y la última para el director (usted). Use un clip de papel en lugar de grapas: un actor de voz experimentado separará las páginas para evitar hacer ruido cuando se hojean.

#### Aspectos legales

Según la ley de derechos de autor, la lectura de texto protegido por derechos de autor por parte de un actor podría ser una representación por la que el autor de la obra debería ser compensado. Esta representación no será reconocible en el producto final, la voz personalizada. Aun así, la legalidad del uso de una obra protegida por derechos de autor con esta finalidad no está bien establecida. Microsoft no puede proporcionar asesoramiento legal sobre este problema; consulte a su asesor.

Afortunadamente, es posible evitar estos problemas por completo. Hay muchas fuentes de texto que puede usar sin licencia o permiso.

FUENTE DE TEXTO	DESCRIPCIÓN
Corpus CMU Arctic	Aproximadamente 1100 oraciones seleccionadas de obras protegidas por derechos de autor para su uso especialmente en proyectos de síntesis de voz. Un excelente punto de partida.
Obras que ya no están protegidas por derechos de autor	Normalmente las obras publicadas antes de 1923. En inglés, el <a href="#">proyecto Gutenberg</a> ofrece miles de obras de este tipo. Quizás quiera centrarse en obras más modernas, dado que el lenguaje estará más próximo al inglés moderno.
Obras del gobierno	Las obras creadas por el gobierno de los Estados Unidos no están protegidas por derechos de autor en los Estados Unidos, aunque es posible que el gobierno reclame derechos de autor en otros países o regiones.
Dominio público	Obras en las que se ha renunciado explícitamente a los derechos de autor o que se han destinado al dominio público. Puede que en algunas jurisdicciones no sea posible renunciar totalmente a los derechos de autor.
Obras con licencia permisiva	Obras distribuidas con una licencia como Creative Commons o la licencia de documentación libre de GNU (GFDL). La Wikipedia usa la GFDL. Algunas licencias, sin embargo, pueden imponer restricciones sobre la representación del contenido con licencia que pueden afectar a la creación de un modelo de voz personalizada, así que lea la licencia detenidamente.

## Grabación del guion

Grabe el guion en un estudio de grabación profesional especializado en trabajos de voz. Dispondrá de una cabina de grabación, el equipo adecuado y las personas adecuadas correctas utilizarlo. Vale la pena no escatimar en la grabación.

Revise el proyecto con el ingeniero de grabación del estudio y escuche sus consejos. La grabación debe tener poca o ninguna compresión de rango dinámico (máximo 4:1). Es esencial que el audio tenga un volumen uniforme y una elevada relación señal-ruido, y que esté libre de sonidos no deseados.

### **Hágalo usted mismo**

Si quiere hacer la grabación por su cuenta y riesgo, en lugar de meterse en un estudio de grabación, este es un manual básico breve. Gracias al aumento de la grabación doméstica y la distribución de archivos multimedia, resulta más fácil que nunca encontrar buena información y recursos de grabación en Internet.

La "cabina de grabación" debe ser una habitación pequeña sin eco apreciable o "tono de la sala". Debe ser lo más silenciosa e insonorizada posible. Se pueden usar cortinas en las paredes para reducir el eco y neutralizar o "amortiguar" el sonido de la habitación.

Use un micrófono de condensador de estudio de alta calidad diseñado para la grabación de voz. Sennheiser, AKG e incluso los recientes micrófonos Zoom pueden producir buenos resultados. Puede comprar un micrófono o alquilar uno en una empresa de alquiler de equipo audiovisual. Busque uno con una interfaz USB. Este tipo de micrófono combina de forma práctica el elemento de micrófono, el preamplificador y el convertidor analógico-digital en un paquete, lo que simplifica la conexión.

También puede usar un micrófono analógico. Muchas de las empresas de alquiler ofrecen micrófonos "vintage" reconocidos por sus personajes de voz. Tenga en cuenta que el equipo analógico profesional usa conectores XLR balanceados, en lugar del conector de 1/4" que se usa en los equipos de consumo. Si se decide por el analógico, también necesitará un preamplificador y una interfaz de audio de equipo informático con estos conectores.

Coloque el micrófono en un pie o soporte tipo jirafa e instale un filtro contra chasquidos para eliminar el ruido de las consonantes oclusivas, como la "p" y la "b". Algunos micrófonos incorporan un montaje de suspensión que los aísla de las vibraciones del pie, lo que es útil.

El actor de voz debe permanecer a una distancia constante del micrófono. Use cinta en el suelo para marcar dónde se debe colocar. Si el actor prefiere sentarse, tenga especial cuidado para controlar la distancia al micrófono y evitar el ruido de la silla.

Use un atril para sostener el guion. Evite inclinar el atril para que pueda reflejar el sonido hacia el micrófono.

La persona que use el equipo de grabación, el ingeniero, debe estar en una habitación distinta de la del actor, con algún medio para comunicarse con él en la cabina de grabación (un *circuito de radio*).

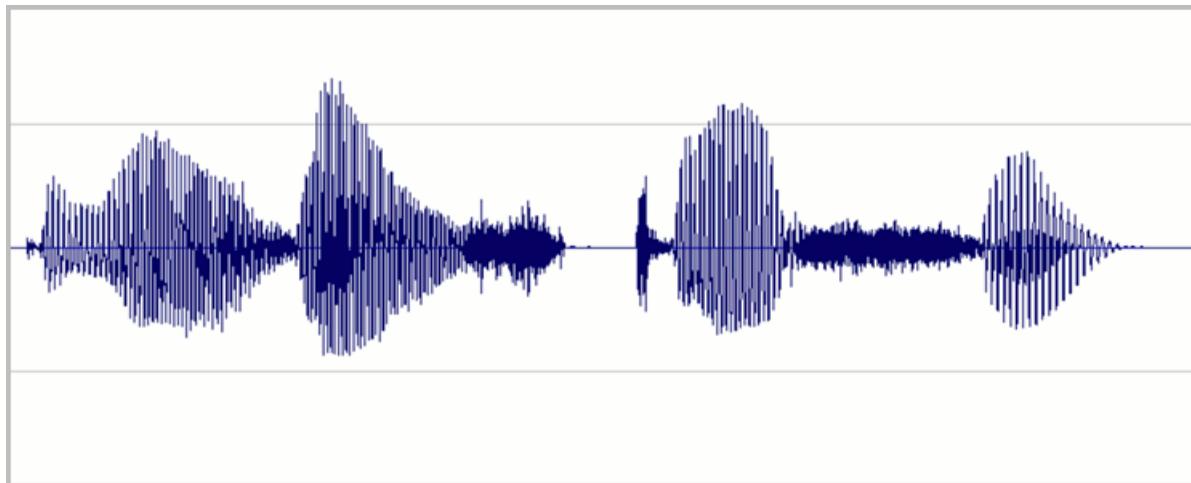
La grabación debe contener el menor ruido posible; el objetivo es una relación señal-ruido de 80 db o superior.

Escuche con atención la grabación de silencio en la cabina, averigüe de dónde proceden los ruidos y elimine la causa. Las fuentes habituales de ruido son los conductos de ventilación, los balastros de tubos fluorescentes, el tráfico de carreteras cercanas y los ventiladores de los equipos (incluso los portátiles pueden tener ventiladores). Los micrófonos y cables pueden captar el ruido eléctrico de los cables cercanos de CA, normalmente un zumbido o murmullo. Un zumbido también se puede producir por un *bucle de tierra*, que se genera al conectar un equipo a más de un circuito eléctrico.

#### **TIP**

En algunos casos, es posible usar un ecualizador o un complemento de software de reducción de ruido para ayudar a eliminar el ruido de las grabaciones, aunque siempre es mejor detenerlo en el origen.

Establezca los niveles de manera que la mayoría del intervalo dinámico de grabación digital disponible se use sin sobremodulación. Esto significa establecer un nivel de audio alto, pero sin que distorsione. En la imagen siguiente se muestra un ejemplo de la forma de onda de una grabación correcta:



Aquí, se usa la mayor parte del rango (alto), pero los picos más altos de la señal no llegan a la parte superior o inferior de la ventana. También puede ver que el silencio en la grabación se aproxima a una línea horizontal fina, que indica un ruido de fondo bajo. Esta grabación tiene un rango dinámico y una relación señal-ruido aceptables.

Grabe directamente en el equipo informático mediante una interfaz de audio de alta calidad o un puerto USB, según el micrófono que vaya a usar. Si es analógico, use lo básico: micrófono, preamplificador, interfaz de audio, equipo informático. Tanto [Avid Pro Tools](#) como [Adobe Audition](#) se pueden usar con licencia mensual por un módico precio. Si su presupuesto es muy reducido, pruebe la aplicación gratuita [Audacity](#).

Grabe con sonido monofónico de 16 bits a 44,1 kHz (calidad de CD) o mejor. El más moderno actualmente es 24 bits a 48 kHz, si el equipo lo admite. La calidad del audio se reduce a 16 bits a 16 kHz antes de enviarlo al portal Custom Voice. Aun así, vale la pena tener una grabación original de alta calidad en caso de que sea necesario realizar modificaciones.

Lo ideal es tener distintas personas en los roles de director, ingeniero y actor. No intente hacerlo todo usted mismo. En caso de emergencia, la misma persona puede ser el director y el ingeniero.

### Antes de la sesión

Para evitar perder tiempo en el estudio, repase el guion con el actor antes de la sesión de grabación. Mientras el actor se familiariza con el texto, puede resolver dudas con la pronunciación de palabras que no resulten muy conocidas.

#### NOTE

La mayoría de los estudios de grabación ofrecen una pantalla electrónica de guiones en la cabina de grabación. En este caso, escriba las notas de revisión directamente en el documento del guion. También puede querer una copia en papel para tomar notas durante la sesión. La mayoría de los ingenieros también querrán una copia en papel. Y la tercera copia impresa servirá como respaldo para el actor en caso de que el equipo informático no funcione.

Es posible que el actor de voz pregunte qué palabra quiere enfatizar en una expresión (la "palabra operativa"). Indíquele que desea una lectura natural sin ningún énfasis en particular. El énfasis se puede agregar cuando se sintetice la voz; no debe formar parte de la grabación original.

Indique el actor que pronuncie las palabras claramente. Cada palabra del guion se debe pronunciar tal y como se escribe. Los sonidos no se deben omitir ni arrastrar juntos, como es habitual en el habla informal, *a menos que se hayan escrito de esa manera en el guion*.

TEXTO ESCRITO	PRONUNCIACIÓN INFORMAL NO DESEADA
me has dejado helado	mehas dejao helao
le dijeron que estaba muy guapa	le dijeron queestaba mu guapa
la ciudad de Madrid	la ciuda de Madri
estoy cansado	estoy cansao

El actor *no* debe agregar pausas claras entre las palabras. La frase debe fluir naturalmente, aunque suene un poco formal. Es posible que se necesite práctica para que esta distinción fina salga bien.

### La sesión de grabación

Al comienzo de la sesión, cree una grabación de referencia, o *archivo de ajuste*, de un enunciado típico. Pida al actor que repita esta línea cada página o página y media. Cada vez, compare la nueva grabación con la referencia. Esta práctica ayuda al actor a mantenerse en un volumen, tono y entonación constantes. Mientras tanto, el ingeniero puede usar el archivo de ajuste como referencia para los niveles y la uniformidad general del sonido.

El archivo de ajuste es especialmente importante cuando se reanuda la grabación tras un descanso, u otro día. Lo reproducirá varias veces para el actor y le hará repetirlo cada vez hasta que la coincidencia sea perfecta.

Entrene a su actor para respirar profundamente y hacer una pausa durante un momento antes de cada enunciado. Grabe un par de segundos de silencio entre enunciados. Las palabras se deben pronunciar de la misma manera cada vez que aparezcan, en función del contexto. Es importante mantener la coherencia en la pronunciación y en la entonación de las palabras.

Grabe un buen silencio de cinco segundos antes de la primera grabación para capturar el "tono de la sala". Esta práctica ayuda al portal Custom Voice a compensar cualquier rastro de ruido que pueda quedar en las grabaciones.

#### TIP

Lo que realmente necesita es captar al actor de voz, por lo que puede hacer una grabación monofónica (un solo canal) de esas líneas concretas. Sin embargo, si graba en estéreo, puede usar el segundo canal para grabar la charla en la sala de control y capturar los comentarios de líneas o cortes determinados. Quite esta pista de la versión que se carga en el portal Custom Voice.

Use auriculares para escuchar con atención la representación del actor de voz. Lo que busca es una dicción buena pero natural, una pronunciación correcta y ausencia de sonidos no deseados. No dude en pedir al actor que vuelva a grabar un enunciado que no satisfaga estas normas.

#### TIP

Cuando se usa un volumen elevado de expresiones, una sola de ellas podría no tener un efecto apreciable sobre la voz personalizada resultante. Es posible que sea más adecuado anotar simplemente las expresiones con problemas, excluirlas del conjunto de datos y ver el resultado de la voz personalizada. Siempre puede volver al estudio y grabar las muestras que faltan más tarde.

Anote en el guion el número de toma o código de tiempo de cada expresión. Pídale también al ingeniero que marque todas las expresiones en los metadatos de la grabación o en la hoja de pistas.

Haga pausas cada cierto tiempo y ofrezca bebidas para ayudar a mantener la voz del actor en buen estado.

## Después de la sesión

Los estudios de grabación modernos funcionan en equipos informáticos. Al final de la sesión, recibirá uno o varios archivos de audio, no una cinta. Estos archivos tendrán probablemente el formato WAV o AIFF en calidad de CD (16 bits a 44,1 kHz) o superior. 24 bits a 48 kHz es lo habitual y deseable. Por lo general, no se necesitan velocidades de muestreo más altas, como 96 kHz.

El portal Custom Voice exige que cada enunciado se proporcione en su propio archivo. Los archivos de audio entregados por el estudio contienen varias expresiones. Así que la principal tarea de posproducción es separar las grabaciones y prepararlas para su envío. Es posible que el ingeniero de grabación haya colocado marcadores en el archivo (o proporcionado una lista de pistas independiente) para indicar dónde comienza cada expresión.

Use sus notas para encontrar exactamente los cortes que quiera y, después, use una utilidad de edición de sonido, como [Avid Pro Tools](#), [Adobe Audition](#), o la aplicación gratuita [Audacity](#) para copiar las expresiones en un archivo nuevo.

Deje solamente unos 0,2 segundos de silencio al principio y al final de cada clip, excepto para el primero. Ese archivo debe empezar con cinco segundos completos de silencio. No use un editor de audio para "poner a cero" las partes de silencio del archivo. Incluir el "tono de la sala" ayudará a los algoritmos de Custom Voice a compensar los ruidos de fondo que hayan podido quedar.

Escuche atentamente cada archivo. En esta fase, puede editar pequeños sonidos no deseados que pasó por alto durante la grabación, como ligeros ruidos con los labios antes de una línea, pero tenga cuidado de no quitar la voz real. Si no puede corregir un archivo, elimínelo del conjunto de datos y tome nota de que lo ha hecho.

Convierta todos los archivos a 16 bits y una velocidad de muestreo de 16 kHz antes de guardar y, si graba la charla del estudio, quite el segundo canal. Guarde cada archivo en formato WAV y nombre los archivos con el número de enunciado del guion.

Por último, cree la *transcripción* que asocia cada archivo WAV con una versión de texto del enunciado correspondiente. En [Crear fuentes de voz personalizada](#) se incluye información detallada del formato requerido. Puede copiar el texto directamente del guion. Después, cree un archivo ZIP de los archivos WAV y de la transcripción de texto.

Archive las grabaciones originales en un lugar seguro en caso de que las necesite más adelante. También, conserve el guion y las notas.

## Pasos siguientes

Está listo para cargar las grabaciones y crear la voz personalizada.

[Creación de fuentes de voz personalizadas](#)

# Audio Content Creation

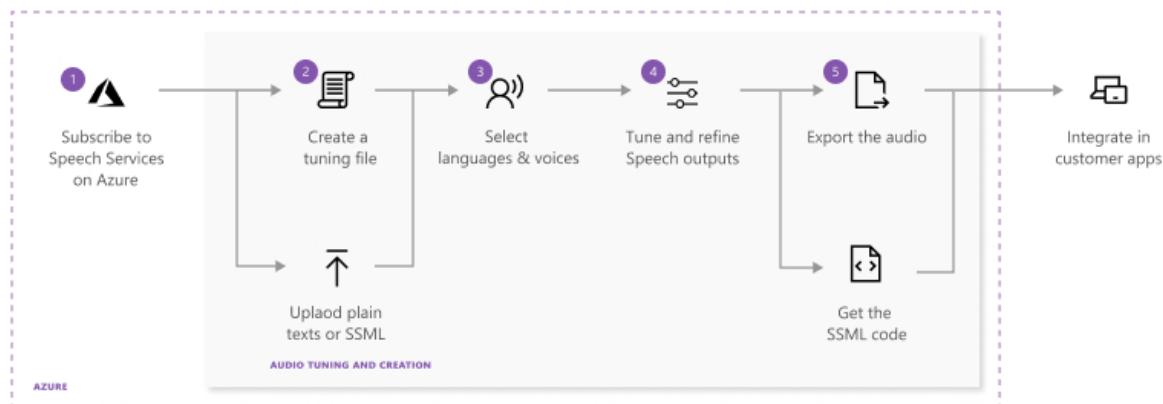
13/01/2020 • 7 minutes to read • [Edit Online](#)

**Audio Content Creation** es una herramienta en línea que le permite personalizar y ajustar la salida de texto a voz de Microsoft para sus aplicaciones y productos. Puede usar esta herramienta para ajustar las voces públicas y personalizadas a fin de lograr expresiones naturales más precisas y administrar el resultado en la nube.

La herramienta Audio Content Creation se basa en el [lenguaje de marcado de síntesis de voz \(SSML\)](#). Para simplificar la personalización y ajuste, Audio Content Creation le permite inspeccionar visualmente las salidas de texto a voz en tiempo real.

## ¿Cómo funciona?

En este diagrama se muestran los pasos necesarios para optimizar y exportar salidas de voz a texto personalizadas. Use los siguientes vínculos para obtener más información sobre cada paso.



1. El primer paso es [crear una cuenta de Azure, registrar un recurso de voz y obtener una clave de suscripción](#). Después de tener una clave de suscripción, puede utilizarla para llamar al servicio de voz y para obtener acceso a [Audio Content Creation](#).
2. [Cree un archivo de ajuste de audio con texto sin formato o SSML](#).
3. Elija la voz y el idioma que quiere optimizar. Audio Content Creation incluye todas las [voces de texto a voz de Microsoft](#). Puede usar voces estándar, neuronales o una propia personalizada.

### NOTE

El acceso controlado está disponible para las voces neuronales personalizadas, que le permiten crear voces de alta definición similares a una voz natural. Para obtener más información, consulte [Proceso de acceso controlado](#).

4. Revise el resultado predeterminado. A continuación, use la herramienta de optimización para ajustar la pronunciación, el tono, la velocidad, la entonación, el estilo de voz, etc. Para obtener una lista completa de opciones, consulte [Lenguaje de marcado de síntesis de voz](#).
5. Guarde y [exporte el audio optimizado](#). Cuando guarde la pista de optimización en el sistema, podrá seguir trabajando y recorrer en iteración la salida. Cuando esté satisfecho con el resultado, puede crear una tarea

de creación de audio con la característica de exportación. Puede observar el estado de la tarea de exportación y descargar la salida para usarla con sus aplicaciones y productos.

6. El último paso es usar la voz personalizada optimizada en sus aplicaciones y productos.

## Creación de un recurso de voz

Siga estos pasos para crear un recurso de voz y conectarlo con Speech Studio.

1. Siga estas instrucciones para [registrarse en una cuenta de Azure](#) y [crear un recurso de Voz](#). Asegúrese de que el plan de tarifa está establecido en **\$0**. Si usa una de las voces neuronales, asegúrese de crear el recurso en una [región compatible](#).
2. Inicie sesión en [Audio Content Creation](#).
3. Seleccione un proyecto existente o haga clic en **Crear nuevo**.
4. Puede modificar su suscripción en cualquier momento con la opción **Configuración**, que se encuentra en la navegación superior.

## Creación de un archivo de ajuste de audio

Hay dos maneras de obtener el contenido de la herramienta Audio Content Creation.

### Opción 1:

1. Despues de iniciar sesión en [Audio Content Creation](#), haga clic en **Audio Tuning** (ajuste de audio) para crear un nuevo archivo de ajuste de audio.
2. Cuando aparece la ventana de edición, puede introducir hasta 10 000 caracteres.
3. No olvide guardar.

### Opción 2:

1. Despues de iniciar sesión en [Audio Content Creation](#), haga clic en **Cargar** para importar uno o varios archivos de texto. Se admiten el texto sin formato y SSML.
2. Al cargar los archivos de texto, asegúrese de que el contenido cumple estos requisitos.

PROPIEDAD	VALOR/NOTAS
Formato de archivo	Texto sin formato (.txt) Texto en SSML (.txt) No se admiten los archivos ZIP.
Formato de codificación	UTF-8
Nombre de archivo	Cada archivo debe tener un nombre único. No se admiten los duplicados.
Longitud del texto	Los archivos de texto no deben tener más de 10 000 caracteres.
Restricciones de SSML	Cada archivo SSML solo puede contener un único fragmento de SSML.

### Ejemplo de texto sin formato

Welcome to use Audio Content Creation to customize audio output for your products.

## Ejemplo de texto en SSML

```
<speak xmlns="http://www.w3.org/2001/10/synthesis" xmlns:mstts="http://www.w3.org/2001/mstts" version="1.0"
xml:lang="en-US">
    <voice name="Microsoft Server Speech Text to Speech Voice (en-US, JessaNeural)">
        Welcome to use Audio Content Creation <break time="10ms" />to customize audio output for your products.
    </voice>
</speak>
```

## Exportación de audio ajustado

Una vez que haya revisado la salida de audio y esté satisfecho con la optimización y el ajuste, puede exportar el audio.

1. En la herramienta [Audio Content Creation](#), haga clic en **Exportar** para crear una tarea de creación de audio.
2. Elija el formato de salida para el audio ajustado. A continuación se ofrece una lista de formatos admitidos y frecuencias de muestreo.
3. Puede ver el estado de la tarea en la pestaña **Export task** (tarea de exportación). Si se produce un error en la tarea, consulte la página de información detallada para obtener un informe completo.
4. Una vez completada la tarea, el audio está disponible para su descarga en la pestaña **Audio Library** (biblioteca de audio).
5. Haga clic en **Descargar**. Ahora está listo para usar el audio ajustado personalizado en sus aplicaciones o productos.

### Formatos de audio compatibles

FORMATO	FRECUENCIA DE MUESTREO DE 16 KHZ	FRECUENCIA DE MUESTREO DE 24 KHZ
wav	riff-16khz-16bit-mono-pcm	riff-24khz-16bit-mono-pcm
mp3	audio-16khz-128kbitrate-mono-mp3	audio-24khz-160kbitrate-mono-mp3

## Otras referencias

- [Long Audio API](#)

## Pasos siguientes

[Speech Studio](#)

# Directrices para crear una palabra clave efectiva

15/01/2020 • 3 minutes to read • [Edit Online](#)

Crear una palabra clave efectiva es fundamental para garantizar que el dispositivo responda de forma coherente y precisa. La personalización de la palabra clave es una forma eficaz de diferenciar el dispositivo y de reforzar la marca. En este artículo, aprenderá algunos principios de orientación para crear una palabra clave efectiva.

## Elección de una palabra clave eficaz

Tenga en cuenta las siguientes directrices al elegir una palabra clave:

- La palabra clave debe ser una palabra o frase en inglés.
- No debería tardar más de dos segundos en decirse.
- Las palabras de 4 a 7 sílabas funcionan mejor. Por ejemplo, "Hey, Computer" es una buena palabra clave, mientras que "Hey" no es adecuada.
- Las palabras clave deben seguir las reglas comunes de pronunciación en inglés.
- Una palabra única o incluso inventada que siga las reglas comunes de pronunciación en inglés puede reducir los falsos positivos. Por ejemplo, "computerama" puede ser una buena palabra clave.
- No elija una palabra común. Por ejemplo, "eat" y "go" son palabras que la gente dice con frecuencia en una conversación ordinaria. Pueden ser desencadenadores falsos para su dispositivo.
- Evite usar una palabra clave que pueda tener pronunciacições alternativas. Los usuarios tendrían que saber la pronunciación "correcta" para que su dispositivo respondiera. Por ejemplo, "509" puede pronunciarse como "cinco cero nueve" o "quinientos nueve". "R.E.I." puede pronunciarse como "r-e-i" o "rey". "Live" puede pronunciarse como "/līv/" o "/liv/".
- No utilice caracteres especiales, símbolos o dígitos. Por ejemplo, "Go#" y "20 + cats" podrían ser palabras clave problemáticas. Sin embargo, "go sharp" o "twenty plus cats" pueden funcionar. Todavía puede utilizar los símbolos en su marca y utilizar el marketing y la documentación para reforzar la pronunciación adecuada.

### NOTE

Si elige una palabra registrada como marca comercial como palabra clave, asegúrese de que es el propietario de esa marca comercial, o bien de tener el permiso del propietario para usar la palabra. Microsoft no es responsable de ningún problema legal que pueda surgir de su elección de la palabra clave.

## Pasos siguientes

Aprenda a [crear una palabra clave personalizada con Speech Studio](#).

# Creación de una palabra clave personalizada mediante Speech Studio

15/01/2020 • 3 minutes to read • [Edit Online](#)

El dispositivo siempre espera una palabra clave (o frase). Por ejemplo, "Hola Cortana" es una palabra clave para el asistente Cortana. Cuando el usuario dice la palabra clave, el dispositivo envía todo el audio subsiguiente a la nube, hasta que el usuario deja de hablar. La personalización de la palabra clave es una forma eficaz de diferenciar el dispositivo y de reforzar la marca.

En este artículo, va a aprender a crear una palabra clave personalizada para el dispositivo.

## Creación de la palabra clave

Antes de poder usar una palabra clave personalizada, deberá crear una palabra clave con la página [Custom Keyword](#) (Palabra clave personalizada) en [Speech Studio](#). Después de proporcionar una palabra clave, se genera un archivo que se implementa en el dispositivo.

1. Vaya a [Speech Studio](#) e **inicie sesión** o, si todavía no tiene una suscripción a Voz, elija [Crear una suscripción](#).
2. En la página [Custom Keyword](#) (Palabra clave personalizada), cree un **Nuevo proyecto**.
3. Escriba un **Nombre**, una **Descripción** opcional y seleccione el idioma. Necesitará un proyecto por idioma, y la compatibilidad está limitada actualmente al idioma en-US.

The screenshot shows the 'New project' dialog box. It has fields for 'Name \*' (with placeholder 'Name your project'), 'Description' (with placeholder 'Describe your project'), and 'Language \*' (a dropdown menu with placeholder 'Select a language'). At the bottom, there is a note: 'Note: Your language should match the data you intend to use for training.' and two buttons: 'Cancel' and 'Create'.

New project

Name \*

Name your project

Description

Describe your project

Language \*

Select a language

Note: Your language should match the data you intend to use for training.

Cancel Create

4. Seleccione el proyecto de la lista.

## Speech Studio

### Custom Keyword

Create your own custom keyword to work with our [Speech Device SDK](#). Please read the guide to select an effective keyword. [Learn more](#)

[New project](#)  

<input type="checkbox"/> Name ↑↓	Description ↑↓	Language ↑↓	Datasets ↑↓	Models ↑↓	Created ↓
<input type="checkbox"/> MyKeyword	A test keyword	English (United States)	0	0	11/27/2019 9:16 AM

5. Para iniciar un nuevo modelo de palabra clave, haga clic en **Entrenar modelo**.
6. Escriba un **Nombre** para el modelo de palabra clave y una **Descripción** opcional, y escriba la **Palabra clave** de su elección y haga clic en **Siguiente**. Existen algunas [directrices](#) para ayudarlo a elegir una palabra clave efectiva.

### New model

X

Name \*

Description

Keyword \*

Cancel
Next

7. Ahora, el portal creará pronunciaciones candidatas para la palabra clave. Para escuchar a cada candidato, haga clic en los botones de reproducción y desactive las pronunciaciones que sean incorrectas. Una vez que solo las pronunciaciones correctas estén marcadas como activas, haga clic en **Entrenar** para empezar a generar la palabra clave.

### Choose pronunciations

X

Keyword

Hello Computer

Select pronunciations

---

Add training data to train keyword model.

Back
Train

8. El modelo puede tardar hasta diez minutos en generarse. La lista de palabras clave cambiará de **Procesando** a **Correcto** cuando el modelo esté completo. Luego puede descargar el archivo.

Name	Description	Keyword	Created	Status	Action
My New Keyword	A test keyword	Hello Computer	11/27/2019 9:19 AM	Succeeded	

9. Guarde el archivo .zip en el equipo. Necesitará este archivo para implementar la palabra clave personalizada en el dispositivo.

## Pasos siguientes

Pruebe la palabra clave personalizada con el [inicio rápido del SDK de dispositivos de Voz](#).

# Servicio Voz para datos de telefonía

13/01/2020 • 23 minutes to read • [Edit Online](#)

Los datos de telefonía que se generan a través de líneas fijas, teléfonos móviles y radios son típicamente de baja calidad, y de banda estrecha en el rango de 8 KHz, lo que crea desafíos al convertir la voz en texto. Los últimos modelos de reconocimiento de voz del servicio Voz destacan en la transcripción de estos datos de telefonía, incluso en los casos en que los datos son difíciles de entender para un humano. Estos modelos están entrenados con grandes volúmenes de datos de telefonía y tienen la mejor precisión de reconocimiento del mercado, incluso en entornos ruidosos.

Un escenario común para la conversión de voz a texto es la transcripción de grandes volúmenes de datos de telefonía que pueden provenir de varios sistemas, como la respuesta de voz interactiva (IVR). El audio que proporcionan estos sistemas puede ser estéreo o mono, y sin procesar con poco o nada de procesamiento posterior realizando en la señal. Con el servicio Voz y el modelo de voz unificado, una empresa puede obtener transcripciones de alta calidad, sea cual sea el sistema de captura de audio.

Los datos de telefonía se pueden utilizar para comprender mejor las necesidades de sus clientes, identificar nuevas oportunidades de marketing o evaluar el rendimiento de los agentes del centro de llamadas. Despues de que se transcriben los datos, la empresa puede utilizar la salida para mejorar la telemetría, identificar frases clave o analizar la opción del cliente.

Las tecnologías descritas en esta página son de Microsoft internamente para varios servicios de procesamiento de llamadas de soporte, tanto en tiempo real como por lotes.

Revisemos algunas de las tecnologías y características relacionadas que ofrece el servicio Voz.

## IMPORTANT

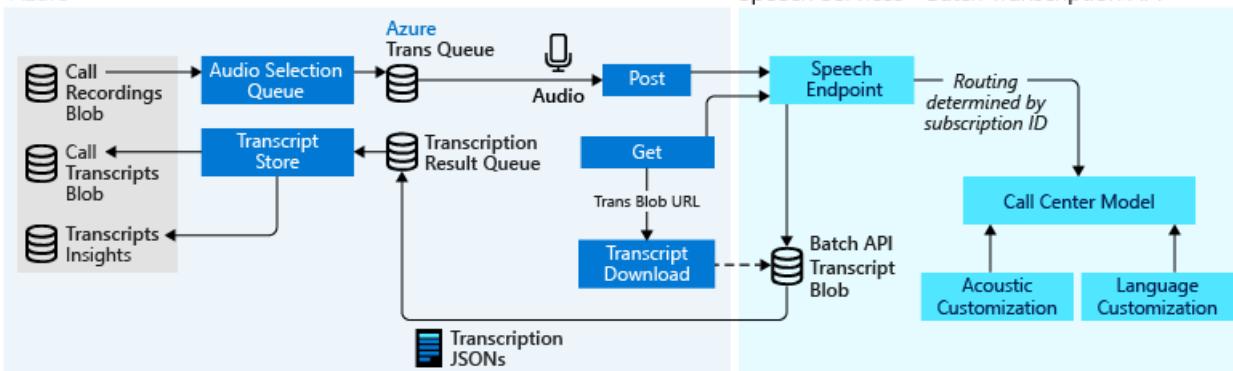
El modelo unificado del servicios Voz está entrenado con diversos datos y ofrece una solución de modelo único para una serie de escenarios que van desde el dictado hasta el análisis de la telefonía.

## Tecnología de Azure para centros de llamada

Más allá del aspecto funcional del servicio Voz, su objetivo principal, cuando se aplica al centro de llamadas, es mejorar la experiencia del cliente. A este respecto, existen tres ámbitos claros:

- Análisis posterior a la llamada, que esencialmente es el procesamiento por lotes de las grabaciones de llamadas después de la llamada.
- Análisis en tiempo real, que es el procesamiento de la señal de audio para extraer varias informaciones a medida que se realiza la llamada (siendo la opinión un caso de uso principal).
- Asistentes de voz (bots), ya sea conduciendo el diálogo entre el cliente y el bot en un intento de resolver el problema del cliente sin la participación del agente, o como aplicación de protocolos de inteligencia artificial (IA) para ayudar al agente.

Un diagrama de arquitectura típico de la implementación de un escenario por lotes se muestra en la siguiente ilustración



## Componentes de la tecnología de análisis de voz

Ya sea que el dominio sea posterior a la llamada o en tiempo real, Azure ofrece un conjunto de tecnologías maduras y emergentes para mejorar la experiencia del cliente.

### Conversión de voz en texto

La características de [conversión de voz en texto](#) es la más buscada en cualquier solución de centro de llamadas. Como muchos de los procesos analíticos posteriores se basan en texto transcritto, la tasa de errores por palabra (*WER*) es de suma importancia. Uno de los principales desafíos en la transcripción para los centros de llamadas es el ruido que prevalece en dichos centros (por ejemplo, otros agentes que hablan en segundo plano), la rica variedad de idiomas locales y dialectos, así como la baja calidad de la señal telefónica real. *WER* está muy correlacionado con el nivel de formación de los modelos acústicos y de lenguaje para una determinada configuración regional, por lo que es importante poder personalizar el modelo a la suya. Nuestros modelos de la versión 4.x unificada más recientes son la solución perfecta tanto para la precisión de transcripción como para la latencia. Entrenados con decenas de miles de horas de datos acústicos y miles de millones de información léxica, los modelos unificados son los más precisos del mercado para transcribir los datos del centro de llamadas.

### Opinión

Evaluar si el cliente tuvo una buena experiencia es una de las áreas más importantes del análisis de voz cuando se aplica al espacio del centro de llamadas. Nuestra [API Batch Transcription](#) ofrece análisis de las opiniones por expresión. Puede agregar el conjunto de valores obtenidos como parte de una transcripción de la llamada para determinar la opinión de la llamada tanto para sus agentes como para el cliente.

### Silencio (sin hablar)

No es raro que el 35 por ciento de una llamada de soporte técnico sea lo que llamamos tiempo de silencios. Algunos de los escenarios en los que no se habla son: agentes que consultan el historial de casos anteriores con un cliente, agentes que utilizan herramientas que les permiten acceder al escritorio del cliente y realizar funciones o clientes que están en espera de una transferencia, entre otros. Es extremadamente importante evaluar cuándo se está produciendo el silencio en una llamada, ya que hay una serie de importantes sensibilidades de los clientes que se producen en torno a este tipo de escenarios y dónde se producen en la llamada.

### Traducción

Algunas empresas experimentan ahora con ofrecer transcripciones traducidas de llamadas de soporte técnico en idiomas extranjeros para que los administradores de entrega puedan comprender la experiencia de sus clientes de todo el mundo. Nuestras funcionalidades de [traducción](#) son insuperables. Podemos traducir audio a audio o audio a texto para un gran número de configuraciones regionales.

### Texto a voz

[Texto a voz](#) es otra área importante en la implementación de bots que interactúan con los clientes. La vía típica es que el cliente habla, su voz se transcribe en texto, el texto se analiza en busca de intenciones, se sintetiza una respuesta basada en la intención reconocida y, a continuación, se muestra un recurso para el cliente o se genera una respuesta de voz sintetizada. Por supuesto, todo esto tiene que ocurrir rápidamente, por lo que la baja latencia

es un componente importante para el éxito de estos sistemas.

Nuestra latencia de un extremo a otro es considerablemente baja si se tienen en cuenta las diversas tecnologías implicadas, tales como [Conversión de voz en texto](#), [LUIS](#), [Bot Framework](#) y [Texto a voz](#).

Nuestras nuevas voces son también indistinguibles de las voces humanas. Puede usar nuestras voces para darle al bot su personalidad única.

## Search

Otro elemento básico del análisis es identificar las interacciones en las que ha ocurrido un evento o experiencia específica. Esto se hace típicamente con uno de dos enfoques, ya sea una búsqueda ad hoc donde el usuario simplemente escribe una frase y el sistema responde, o una consulta más estructurada, donde un analista puede crear un conjunto de sentencias lógicas que identifican un escenario en una llamada, y luego cada llamada puede ser indexada contra ese conjunto de consultas. Un buen ejemplo de búsqueda es la omnipresente declaración de conformidad "Esta llamada se grabará con fines de calidad...". Muchas empresas quieren asegurarse de que sus agentes están proporcionando esta declinación de responsabilidades a los clientes antes de que se grabe la llamada. La mayoría de los sistemas de análisis muestran la tendencia de los comportamientos encontrados por los algoritmos de consulta o búsqueda, ya que este informe de tendencias es en última instancia una de las funciones más importantes de un sistema de análisis. Mediante el [directorio de Cognitive Services](#), tu solución integral puede mejorar significativamente con funcionalidades de indexación y búsqueda.

## Extracción de frases clave

Esta área es una de las aplicaciones de análisis más desafiantes y una que se está beneficiando de la aplicación de la inteligencia artificial y del aprendizaje automático. El escenario principal en este caso es inferir la intención del cliente. ¿Por qué llama el cliente? ¿Cuál es el problema del cliente? ¿Por qué el cliente ha tenido una experiencia negativa? Nuestro [servicio de análisis de texto](#) proporciona un conjunto de análisis de fábrica para actualizar rápidamente su solución completa y extraer esas palabras clave o frases importantes.

Veamos ahora con más detalle el procesamiento por lotes y las canalizaciones en tiempo real para el reconocimiento de voz.

## Transcripción en lote de los datos del centro de llamadas

Para transcribir la mayor parte del audio, se ha desarrollado la [API Batch Transcription](#). La API Batch Transcription se desarrolló para transcribir grandes cantidades de datos de audio de forma asincrónica. En cuanto a la transcripción de datos del centro de llamadas, nuestra solución se basa en estos pilares:

- **Precisión:** con modelos unificados de cuarta generación, ofrecemos una calidad de transcripción insuperable.
- **Latencia:** entendemos que cuando se realizan transcripciones masivas, las transcripciones se necesitan rápidamente. Los trabajos de transcripción iniciados a través de la [API Batch Transcription](#) se pondrán en cola inmediatamente y, una vez que el trabajo empiece a ejecutarse, se realizará más rápidamente que la transcripción en tiempo real.
- **Seguridad:** entendemos que las llamadas pueden contener información confidencial. Puede estar tranquilo porque la seguridad es una de nuestras mayores prioridades. Nuestro servicio ha obtenido las certificaciones ISO, SOC, HIPAA y PCI.

Los centros de llamadas generan grandes volúmenes de datos de audio diariamente. Si su empresa almacena datos de telefonía en una ubicación central, como Azure Storage, puede utilizar la [API Batch Transcription](#) para solicitar y recibir transcripciones de forma asincrónica.

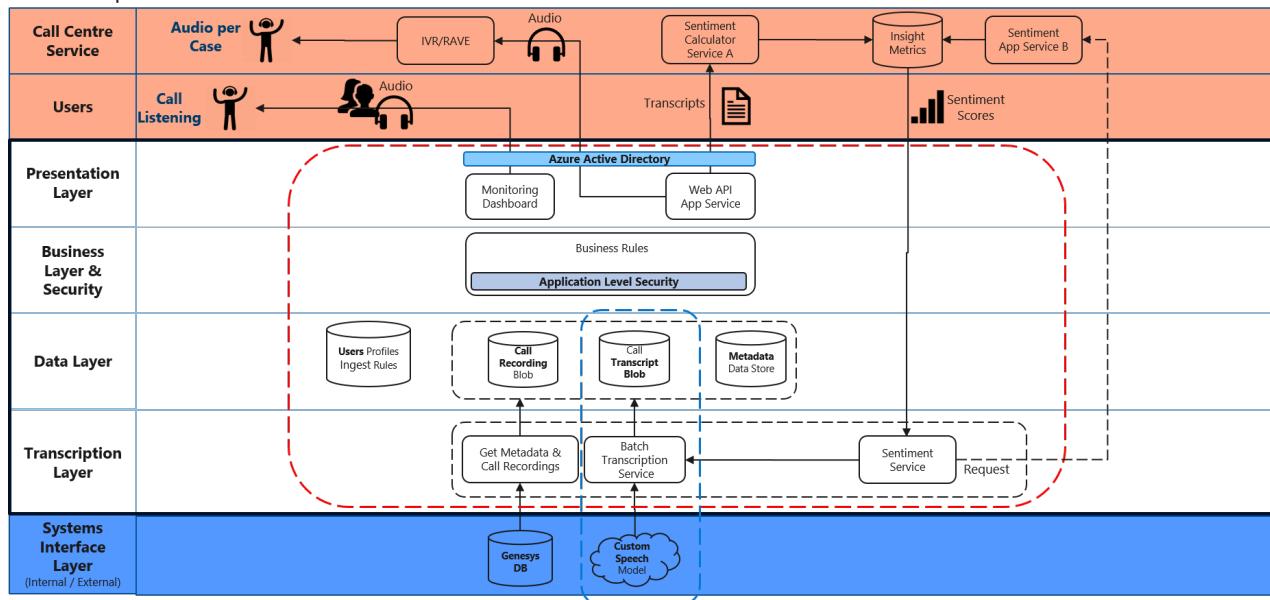
Una solución típica utiliza estos servicios:

- El servicio Voz se usan para convertir voz en texto. Se requiere una suscripción estándar (S0) al servicio Voz para usar la API Batch Transcription. Las suscripciones gratuitas (F0) no funcionarán.
- [Azure Storage](#) se utiliza para almacenar datos de telefonía y las transcripciones devueltas por la API Batch

Transcription. Esta cuenta de almacenamiento debe utilizar notificaciones, específicamente para cuando se agregan nuevos archivos. Estas notificaciones se utilizan para desencadenar el proceso de transcripción.

- La solución [Azure Functions](#) se utiliza para crear un identificador URI de firmas de acceso compartido (SAS) para cada grabación, y desencadenar la petición HTTP POST para iniciar una transcripción. Además, Azure Functions se utiliza para crear solicitudes de recuperación y eliminación de transcripciones mediante la API Batch Transcription.

Internamente estamos utilizando las tecnologías anteriores para admitir las llamadas de los clientes de Microsoft en modo por lotes.

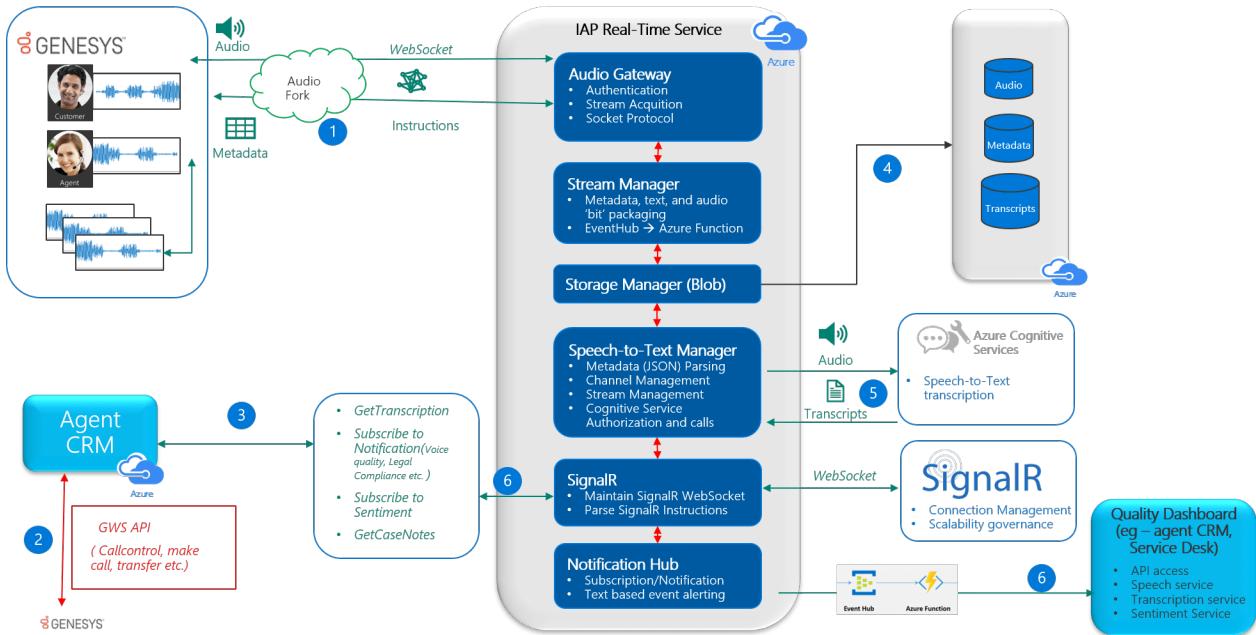


## Transcripción en tiempo real para los datos del centro de llamadas

Algunas empresas deben transcribir las conversaciones en tiempo real. Se puede utilizar la transcripción en tiempo real para identificar palabras clave y desencadenar búsquedas de contenido y recursos relevantes para la conversación, para supervisar la opinión, para mejorar la accesibilidad o para proporcionar traducciones a los clientes y agentes que no son hablantes nativos.

Para escenarios que requieren transcripción en tiempo real, recomendamos usar [Speech SDK](#). Actualmente, la conversión de voz a texto está disponible en [más de 20 idiomas](#) y el SDK está disponible en C++, C#, Java, Python, Node.js, Objective-C y JavaScript. En [GitHub](#) se pueden encontrar ejemplos en todos los idiomas. Para las últimas noticias y actualizaciones, consulte las [Notas de la versión](#).

Internamente estamos utilizando las tecnologías anteriores para analizar en tiempo real las llamadas de los clientes de Microsoft cuando se producen, como se muestra en el diagrama siguiente.



## Una palabra sobre las IVR

El servicio Voz se puede integrar fácilmente en cualquier solución mediante el uso del [SDK de Voz](#) o la [API REST](#). Sin embargo, la transcripción para el centro de llamadas puede requerir tecnologías adicionales. Normalmente, se requiere una conexión entre un sistema IVR y Azure. Aunque no ofrecemos tales componentes, esta es una descripción de lo que implica una conexión a un sistema IVR.

Varios productos de servicios de IVR o telefonía (como Genesys o AudioCodes) ofrecen funcionalidades de integración que pueden aprovecharse para permitir el paso de audio entrante y saliente a un servicio de Azure. Básicamente, un servicio personalizado de Azure puede proporcionar una interfaz específica para definir sesiones de llamadas telefónicas (tales como el inicio de llamada o el fin de llamada) y exponer una API de WebSocket para recibir el flujo de audio entrante que se utiliza con el servicio Voz. Las respuestas salientes, como la transcripción de conversaciones o las conexiones con Bot Framework, pueden sintetizarse con el servicio de texto a voz de Microsoft y devolverse al sistema IVR para su reproducción.

Otro escenario es la integración directa con el protocolo de inicio de sesión (SIP). Un servicio de Azure se conecta a un servidor SIP y obtiene así un flujo de entrada y un flujo de salida, que se utiliza para las fases de conversión de voz a texto y de texto a voz. Para conectarse a un servidor SIP, hay ofertas de software comercial, como Ozeki SDK, o [the Teams calling and meetings API](#) (actualmente en versión beta), que están diseñados para admitir este tipo de escenario para llamadas de audio.

## Personalización de la experiencias existentes

El servicio Voz funciona bien con los modelos integrados. Sin embargo, es posible que desee personalizar y optimizar más la experiencia para su producto o entorno. Las opciones de personalización abarcan desde la optimización de modelos acústicos a fuentes de voz únicas para su marca. Una vez que haya creado un modelo personalizado, podrá usarlo con cualquiera de las características del servicio Voz en tiempo real o en modo por lotes.

SERVICIO DE VOZ	MODELO	DESCRIPCIÓN
-----------------	--------	-------------

SERVICIO DE VOZ	MODELO	DESCRIPCIÓN
Voz a texto	Modelo acústico	Cree un modelo acústico personalizado para las aplicaciones, herramientas o dispositivos usados en entornos concretos como en un automóvil o en una planta de producción, cada uno con unas condiciones de grabación específicas. Los ejemplos incluyen el habla con acento, ruidos de fondo específicos o el uso de un micrófono específico para la grabación.
	Modelo de lenguaje	Cree un modelo de lenguaje personalizado para mejorar la transcripción de gramática y vocabulario específicos del sector, como terminología médica o jerga de TI.
	Modelo de pronunciación	Con un modelo de pronunciación personalizado, puede definir el formato fonético y mostrar una palabra o un término. Es útil para controlar términos personalizados, como nombres de producto o acrónimos. Basta con un archivo de pronunciación, que es un archivo <code>.txt</code> simple.
Texto a voz	Fuente de voz	Las fuentes de voz personalizadas le permiten crear una voz única y reconocible para su marca. Solo toma una pequeña cantidad de datos para empezar a trabajar. Cuantos más datos proporcione, más natural y similar a la humana sonará su fuente de voz.

## Código de ejemplo

El código de ejemplo está disponible en GitHub para cada una de las características del servicio Voz. En estos ejemplos se tratan escenarios comunes como la lectura de audio de un archivo o flujo, el reconocimiento continuo y de una sola emisión, y el trabajo con modelos personalizados. Use estos vínculos para ver ejemplos de SDK y REST:

- [Ejemplos de traducción de voz y voz a texto \(SDK\)](#)
- [Ejemplos de transcripción de Azure Batch \(REST\)](#)
- [Ejemplos de texto a voz \(REST\)](#)

## Documentos de referencia

- [Speech SDK](#)
- [Speech Devices SDK](#)
- [API REST: Speech-to-text \(API de REST: Voz a texto\)](#)
- [API REST: Text-to-speech \(API de REST: Texto a voz\)](#)
- [API REST: Batch transcription and customization \(API de REST: Transcripción y personalización de Azure Batch\)](#)

## Pasos siguientes

[Obtenga una clave de suscripción gratuita a los servicios de Voz](#)

# Prueba gratuita del servicio Voz

13/01/2020 • 9 minutes to read • [Edit Online](#)

Usar el servicio Voz es un proceso fácil y asequible. Hay dos opciones disponibles gratis para que pueda descubrir lo que puede hacer el servicio y decidir si es adecuado para sus necesidades:

- Obtener una evaluación gratuita sin proporcionar la información de la tarjeta de crédito (debe tener una cuenta de Azure existente)
- Crear una cuenta de Azure sin cargo alguno durante un período de prueba (se requiere la información de la tarjeta de crédito)

En este artículo, elegirá una de estas opciones que mejor se adapte a sus necesidades.

## NOTE

El servicio Voz tiene dos niveles de servicio: gratis y suscripción, que tienen diferentes limitaciones y ventajas. Si se registra para obtener una cuenta gratuita de Azure, obtendrá 200 USD en crédito de servicios que puede aplicar a una suscripción del servicio Voz de pago durante 30 días.

Si usa el nivel gratis del servicio Voz de bajo volumen, puede conservar esta suscripción gratuita incluso después de que expire la evaluación gratuita o el crédito del servicio.

Para más información, consulte [Precios de Cognitive Services: servicio de voz](#).

## Prueba del servicio Voz sin la información de la tarjeta de crédito

Aunque se recomienda probar el servicio Voz siguiendo las instrucciones de la sección siguiente, puede que prefiera las instrucciones de esta sección si se aplica lo siguiente:

- Ya tiene una cuenta activa de Azure.
- Quiere evaluar el servicio Voz sin crear una nueva cuenta de Azure.
- Prefiere que no se requiera ninguna tarjeta de crédito y que no se guarde ningún dato después del período de prueba.

## NOTE

El período de prueba se iniciará inmediatamente una vez que se hayan completado los pasos siguientes.

1. Vaya a [Pruebe Cognitive Services](#).

2. Seleccione la pestaña **Speech API**.

3. Elija **Obtener clave de API**.

Se le presentarán opciones de facturación. Elija la opción gratis y, después, lea y apruebe el acuerdo de usuario. Se le presentarán claves que puede usar para probar el servicio Voz durante un tiempo limitado.

## Prueba del servicio Voz con una nueva cuenta de Azure

Para suscribirse a una nueva cuenta de Azure, necesitará una cuenta de Microsoft. Si no tiene una cuenta de Microsoft, puede registrarse para obtener una gratuita en el [portal de la cuenta de Microsoft](#). Seleccione **Iniciar sesión con Microsoft** y, luego, cuando se le pida que inicie sesión, seleccione **Crear una cuenta de Microsoft**. Siga los pasos para crear y comprobar la nueva cuenta Microsoft.

Cuando tenga la cuenta de Microsoft, vaya a la [página de suscripción a Azure](#), seleccione **Comenzar gratis** y cree una cuenta de Azure con su cuenta de Microsoft.

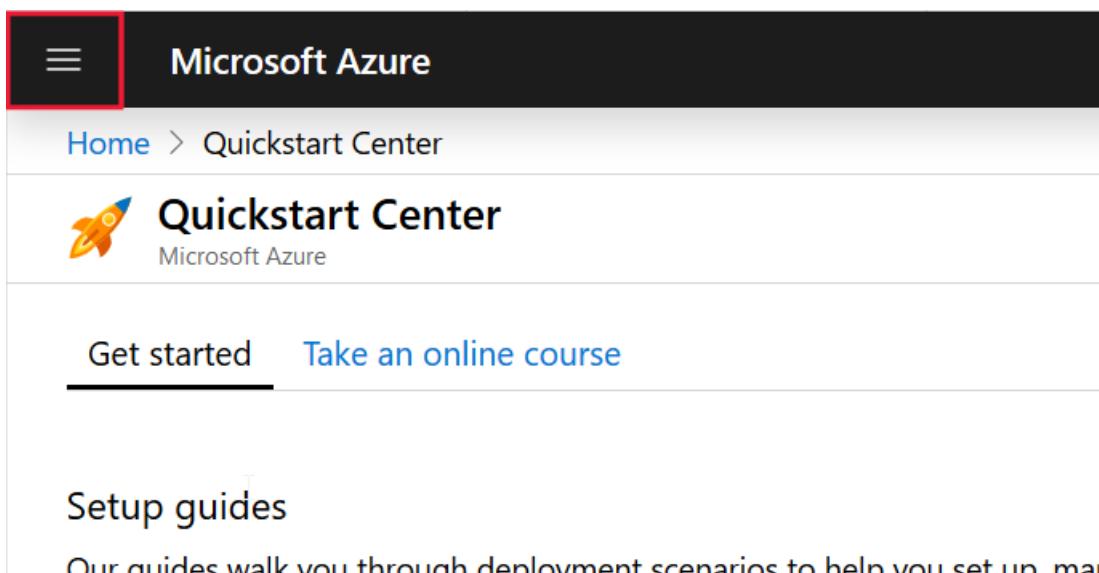
## Crear un recurso de voz en Azure

### NOTE

Puede crear un número ilimitado de suscripciones de plan estándar en una o varias regiones. Pero solo puede crear una suscripción de plan gratuito. Las implementaciones de modelo del plan gratuito que permanezcan inactivas durante siete días se retirarán automáticamente.

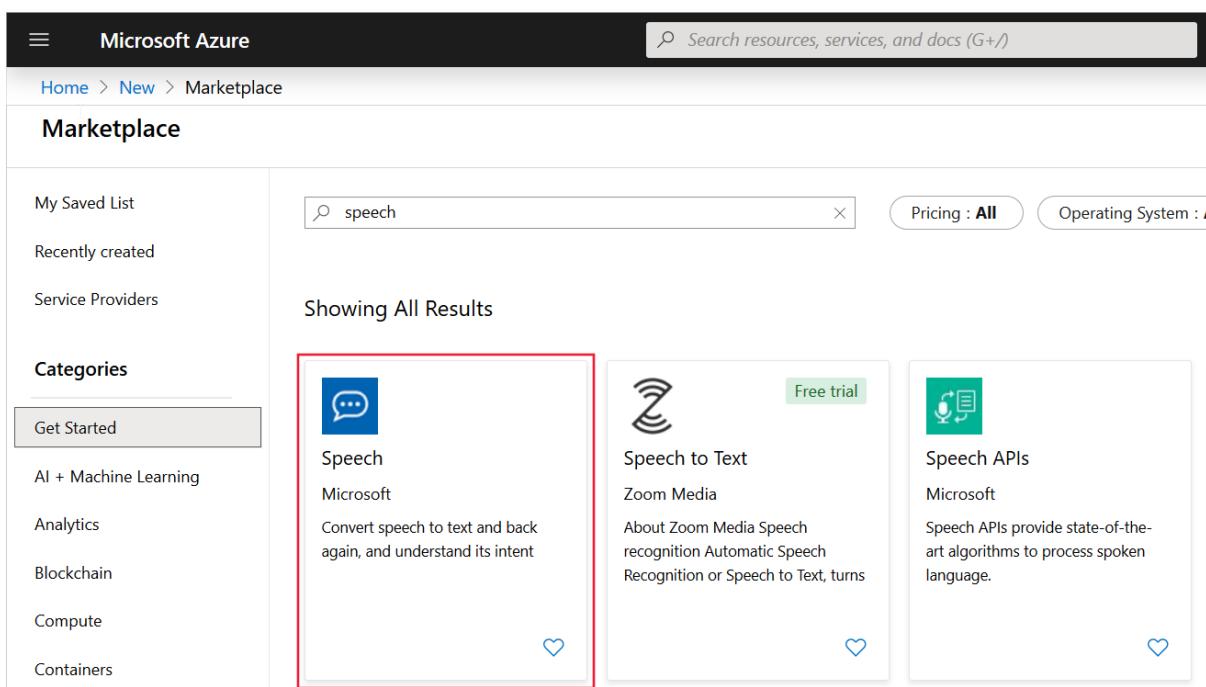
Para agregar un recurso de servicio de voz (plan gratuito o de pago) a la cuenta de Azure:

1. Inicie sesión en [Azure Portal](#) con la cuenta Microsoft.
2. Seleccione **Crear un recurso** en la parte superior izquierda del portal. Si no ve **Crear un recurso**, siempre puede encontrarlo mediante la selección del menú contraído en la parte superior izquierda:



The screenshot shows the Microsoft Azure Quickstart Center. At the top, there's a navigation bar with a menu icon and the text "Microsoft Azure". Below it, the breadcrumb navigation shows "Home > Quickstart Center". The main content area has a title "Quickstart Center" with a rocket ship icon and "Microsoft Azure" below it. There are two buttons at the top: "Get started" (which is underlined and has a red box around its left side) and "Take an online course". Below these buttons, the text "Setup guides" is visible, followed by a truncated sentence: "Our guides walk you through development scenarios to help you set up, manage, and scale your applications.".

3. En la ventana **Nuevo**, escriba "speech" en el cuadro de búsqueda y presione ENTRAR.
4. En los resultados de la búsqueda, seleccione **Voz**.



The screenshot shows the Microsoft Azure Marketplace search results for "speech". The search bar at the top contains "speech". On the left, there's a sidebar with "My Saved List", "Recently created", and "Service Providers". Below that is a "Categories" section with "Get Started" (which is highlighted with a red box), "AI + Machine Learning", "Analytics", "Blockchain", "Compute", and "Containers". The main area shows search results with the heading "Showing All Results". There are three cards displayed:

- Speech** (Microsoft): Converts speech to text and back again, and understands its intent. This card is also highlighted with a red box.
- Speech to Text** (Zoom Media): About Zoom Media Speech recognition. Automatic Speech Recognition or Speech to Text, turns
- Speech APIs** (Microsoft): Provides state-of-the-art algorithms to process spoken language.

## 5. Seleccione **Crear** y, después:

- Dé un nombre único al nuevo recurso. El nombre ayuda a distinguir entre varias suscripciones al mismo servicio.
- Elija la suscripción de Azure a la que esté asociado el recurso nuevo para determinar cómo se facturan las tarifas.
- Elija la [región](#) donde se va a usar el recurso.
- Elija un plan de tarifa de pago (S0) o gratis (F0). Para completar la información sobre los precios y las cuotas de uso de cada plan, seleccione **Ver todos los detalles de los precios**.
- Cree un nuevo grupo de recursos para esta suscripción de voz o asígnela a un grupo de recursos existente. Los grupos de recursos ayudan a mantener organizadas las distintas suscripciones de Azure.
- Seleccione **Crear**. Esto le llevará a la información general de la implementación y mostrará mensajes del progreso de la implementación.

La implementación del recurso de voz nuevo puede tardar unos instantes. Una vez completada la implementación, seleccione **Ir al recurso** y, en el panel de navegación izquierdo, seleccione **Claves** para mostrar las claves de suscripción del servicio Voz. Cada suscripción tiene dos claves; puede usar cualquiera de ellas en la aplicación. Para copiar y pegar rápidamente una clave en el editor de código o en otra ubicación, seleccione el botón **Copiar** que se encuentra junto a cada clave, cambie de ventana para pegar el contenido del portapapeles en la ubicación deseada.

### IMPORTANT

Estas claves de suscripción se usan para tener acceso a la API de Cognitive Services. No comparta las claves. Almacénelas de forma segura, por ejemplo, con Azure Key Vault. También se recomienda regenerar estas claves periódicamente. Solo se necesita una clave para realizar una llamada API. Al volver a generar la primera clave, puede usar la segunda clave para seguir teniendo acceso al servicio.

## Cambiar a una nueva suscripción

Para cambiar de una suscripción a otra, por ejemplo, cuando la evaluación gratuita expire o al publicar la aplicación, sustituya la clave de región y suscripción del código por la clave de región y suscripción del nuevo recurso de Azure.

## Acerca de las regiones

- Si la aplicación usa [Speech SDK](#), proporcione el código de región, por ejemplo, `westus`, al crear una configuración de voz.
- Si la aplicación usa una de las [API de REST](#) del servicio de voz, la región forma parte del URI del punto de conexión que se emplea al realizar solicitudes.
- Las claves creadas para una región son válidas únicamente en esa región. Si intenta usarlas con otras regiones se producen errores de autenticación.

## Pasos siguientes

Realice alguno de los inicios rápidos de 10 minutos o visite los ejemplos de SDK:

[Inicio rápido: Reconocimiento de voz en C# Ejemplos del SDK de Voz](#)

# Prueba gratuita del servicio Voz

13/01/2020 • 9 minutes to read • [Edit Online](#)

Usar el servicio Voz es un proceso fácil y asequible. Hay dos opciones disponibles gratis para que pueda descubrir lo que puede hacer el servicio y decidir si es adecuado para sus necesidades:

- Obtener una evaluación gratuita sin proporcionar la información de la tarjeta de crédito (debe tener una cuenta de Azure existente)
- Crear una cuenta de Azure sin cargo alguno durante un período de prueba (se requiere la información de la tarjeta de crédito)

En este artículo, elegirá una de estas opciones que mejor se adapte a sus necesidades.

## NOTE

El servicio Voz tiene dos niveles de servicio: gratis y suscripción, que tienen diferentes limitaciones y ventajas. Si se registra para obtener una cuenta gratuita de Azure, obtendrá 200 USD en crédito de servicios que puede aplicar a una suscripción del servicio Voz de pago durante 30 días.

Si usa el nivel gratis del servicio Voz de bajo volumen, puede conservar esta suscripción gratuita incluso después de que expire la evaluación gratuita o el crédito del servicio.

Para más información, consulte [Precios de Cognitive Services: servicio de voz](#).

## Prueba del servicio Voz sin la información de la tarjeta de crédito

Aunque se recomienda probar el servicio Voz siguiendo las instrucciones de la sección siguiente, puede que prefiera las instrucciones de esta sección si se aplica lo siguiente:

- Ya tiene una cuenta activa de Azure.
- Quiere evaluar el servicio Voz sin crear una nueva cuenta de Azure.
- Prefiere que no se requiera ninguna tarjeta de crédito y que no se guarde ningún dato después del período de prueba.

## NOTE

El período de prueba se iniciará inmediatamente una vez que se hayan completado los pasos siguientes.

1. Vaya a [Pruebe Cognitive Services](#).
2. Seleccione la pestaña **Speech API**.
3. Elija **Obtener clave de API**.

Se le presentarán opciones de facturación. Elija la opción gratis y, después, lea y apruebe el acuerdo de usuario. Se le presentarán claves que puede usar para probar el servicio Voz durante un tiempo limitado.

## Prueba del servicio Voz con una nueva cuenta de Azure

Para suscribirse a una nueva cuenta de Azure, necesitará una cuenta de Microsoft. Si no tiene una cuenta de Microsoft, puede registrarse para obtener una gratuita en el [portal de la cuenta de Microsoft](#). Seleccione **Iniciar sesión con Microsoft** y, luego, cuando se le pida que inicie sesión, seleccione **Crear una cuenta de Microsoft**. Siga los pasos para crear y comprobar la nueva cuenta Microsoft.

Cuando tenga la cuenta de Microsoft, vaya a la [página de suscripción a Azure](#), seleccione **Comenzar gratis** y cree una cuenta de Azure con su cuenta de Microsoft.

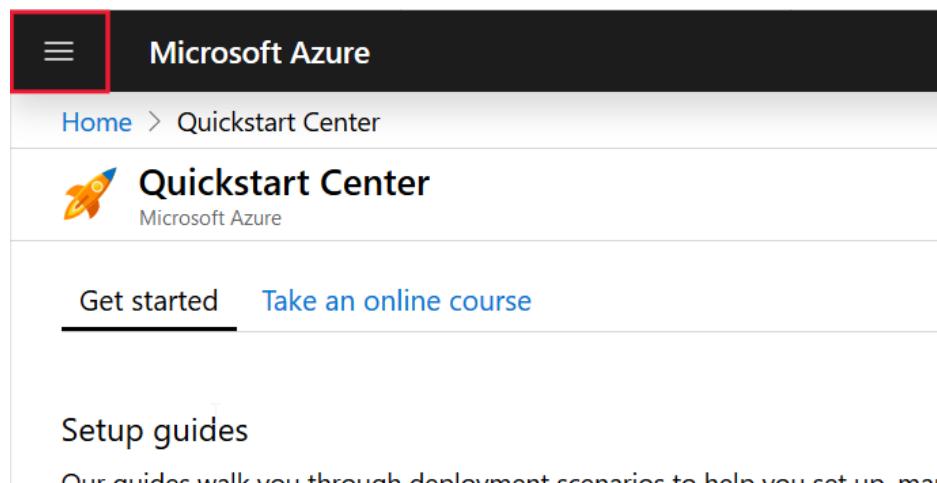
### Crear un recurso de voz en Azure

#### NOTE

Puede crear un número ilimitado de suscripciones de plan estándar en una o varias regiones. Pero solo puede crear una suscripción de plan gratuito. Las implementaciones de modelo del plan gratuito que permanezcan inactivas durante siete días se retirarán automáticamente.

Para agregar un recurso de servicio de voz (plan gratuito o de pago) a la cuenta de Azure:

1. Inicie sesión en [Azure Portal](#) con la cuenta Microsoft.
2. Seleccione **Crear un recurso** en la parte superior izquierda del portal. Si no ve **Crear un recurso**, siempre puede encontrarlo mediante la selección del menú contraído en la parte superior izquierda:



3. En la ventana **Nuevo**, escriba "speech" en el cuadro de búsqueda y presione ENTRAR.
4. En los resultados de la búsqueda, seleccione **Voz**.

The screenshot shows the Microsoft Azure Marketplace interface. At the top, there's a search bar with the placeholder 'Search resources, services, and docs (G+/-)' and a search icon. Below the search bar, the breadcrumb navigation shows 'Home > New > Marketplace'. On the left, there's a sidebar with 'My Saved List', 'Recently created', and 'Service Providers'. The main area is titled 'Marketplace' and shows 'Showing All Results'. A search bar at the top of this area contains the text 'speech'. Below it, there are three service cards. The first card, 'Speech' by Microsoft, is highlighted with a red border. It has a blue speech bubble icon, the service name, the provider 'Microsoft', and a brief description: 'Convert speech to text and back again, and understand its intent'. The second card, 'Speech to Text', is associated with 'Zoom Media' and has a green 'Free trial' button. The third card, 'Speech APIs', also by Microsoft, describes how Speech APIs provide state-of-the-art algorithms to process spoken language. Each service card includes a small heart icon for favoriting.

##### 5. Seleccione **Crear** y, después:

- Dé un nombre único al nuevo recurso. El nombre ayuda a distinguir entre varias suscripciones al mismo servicio.
- Elija la suscripción de Azure a la que esté asociado el recurso nuevo para determinar cómo se facturan las tarifas.
- Elija la **región** donde se va a usar el recurso.
- Elija un plan de tarifa de pago (S0) o gratis (F0). Para completar la información sobre los precios y las cuotas de uso de cada plan, seleccione **Ver todos los detalles de los precios**.
- Cree un nuevo grupo de recursos para esta suscripción de voz o asígnela a un grupo de recursos existente. Los grupos de recursos ayudan a mantener organizadas las distintas suscripciones de Azure.
- Seleccione **Crear**. Esto le llevará a la información general de la implementación y mostrará mensajes del progreso de la implementación.

La implementación del recurso de voz nuevo puede tardar unos instantes. Una vez completada la implementación, seleccione **Ir al recurso** y, en el panel de navegación izquierdo, seleccione **Claves** para mostrar las claves de suscripción del servicio Voz. Cada suscripción tiene dos claves; puede usar cualquiera de ellas en la aplicación. Para copiar y pegar rápidamente una clave en el editor de código o en otra ubicación, seleccione el botón **Copiar** que se encuentra junto a cada clave, cambie de ventana para pegar el contenido del portapapeles en la ubicación deseada.

#### **IMPORTANT**

Estas claves de suscripción se usan para tener acceso a la API de Cognitive Services. No comparta las claves. Almacénelas de forma segura, por ejemplo, con Azure Key Vault. También se recomienda regenerar estas claves periódicamente. Solo se necesita una clave para realizar una llamada API. Al volver a generar la primera clave, puede usar la segunda clave para seguir teniendo acceso al servicio.

## Cambiar a una nueva suscripción

Para cambiar de una suscripción a otra, por ejemplo, cuando la evaluación gratuita expire o al publicar la aplicación, sustituya la clave de región y suscripción del código por la clave de región y suscripción del nuevo recurso de Azure.

## Acerca de las regiones

- Si la aplicación usa [Speech SDK](#), proporcione el código de región, por ejemplo, `westus`, al crear una configuración de voz.
- Si la aplicación usa una de las [API de REST](#) del servicio de voz, la región forma parte del URI del punto de conexión que se emplea al realizar solicitudes.
- Las claves creadas para una región son válidas únicamente en esa región. Si intenta usarlas con otras regiones se producen errores de autenticación.

## Pasos siguientes

Realice alguno de los inicios rápidos de 10 minutos o visite los ejemplos de SDK:

[Inicio rápido: Reconocimiento de voz en C# Ejemplos del SDK de Voz](#)

# Instalación y ejecución de contenedores del servicio de voz (versión preliminar)

14/01/2020 • 38 minutes to read • [Edit Online](#)

Los contenedores le permiten ejecutar algunas de las API del servicio de voz en su propio entorno. Los contenedores son excelentes para requisitos específicos de control de datos y seguridad. En este artículo, aprenderá a descargar, instalar y ejecutar un contenedor de Voz.

Los contenedores de Voz permiten a los clientes compilar una arquitectura de aplicación de voz optimizada para las sólidas funcionalidades de la nube y la localidad del perímetro. Hay cuatro contenedores distintos disponibles. Los dos contenedores estándar son **conversión de voz a texto** y **conversión de texto a voz**. Los dos contenedores personalizados son **conversión de voz a texto personalizada** y **conversión de texto a voz personalizada**.

## IMPORTANT

Actualmente, todos los contenedores de voz se ofrecen como parte de una [versión preliminar pública "validada"](#). Se hará un anuncio cuando los contenedores de voz pasen a la disponibilidad general (GA).

FUNCIÓN	CARACTERÍSTICAS	MÁS RECIENTE
Voz a texto	Transcribe registros continuos de voz en tiempo real o de audio por lotes a texto con resultados intermedios.	2.0.0
Conversión de voz a texto personalizada	Con un modelo personalizado del <a href="#">portal de Voz personalizada</a> , transcribe las grabaciones continuas de voz en tiempo real o de audio por lotes a texto con resultados inmediatos.	2.0.0
Texto a voz	Convierte texto a voz de sonido natural con entrada de texto sin formato o Lenguaje de marcado de síntesis de voz (SSML).	1.3.0
Conversión de texto a voz personalizada	Con un modelo personalizado del <a href="#">portal de Voz personalizada</a> , convierte texto a voz de sonido natural con entrada de texto sin formato o Lenguaje de marcado de síntesis de voz (SSML).	1.3.0

Si no tiene una suscripción a Azure, cree una [cuenta gratuita](#) antes de empezar.

## Prerequisites

Requisitos previos para poder usar los contenedores de Voz:

OBLIGATORIO	PROPÓSITO
Motor de Docker	<p>Necesita que el motor de Docker esté instalado en un <a href="#">equipo host</a>. Docker dispone de paquetes que configuran el entorno de Docker en <a href="#">macOS</a>, <a href="#">Windows</a> y <a href="#">Linux</a>. Para conocer los principios básicos de Docker y de los contenedores, consulte <a href="#">Introducción a Docker</a>.</p> <p>Docker debe configurarse para permitir que los contenedores se conecten con Azure y envíen datos de facturación a dicho servicio.</p> <p><b>En Windows</b>, Docker también debe estar configurado de forma que admita los contenedores de Linux.</p>
Conocimientos sobre Docker	<p>Debe tener conocimientos básicos sobre los conceptos de Docker, como los registros, los repositorios, los contenedores y las imágenes de contenedor, así como conocer los comandos <code>docker</code> básicos.</p>
Recurso de Voz	<p>Para usar estos contenedores, debe tener:</p> <p>Recurso de Voz de Azure para obtener la clave de API y el URI de punto de conexión asociados. Ambos valores están disponibles en las páginas Introducción y Claves de <b>Voz</b> de Azure Portal. Los dos son necesarios para iniciar el contenedor.</p> <p><b>{API_KEY}</b> : una de las dos claves de recurso disponibles en la página <b>Claves</b></p> <p><b>{ENDPOINT_URI}</b> : el punto de conexión tal como se proporciona en la página de <b>Información general</b>.</p>

## Solicitud de acceso al registro de contenedor

Rellene y envíe el [formulario de solicitud de contenedores de Voz de Cognitive Services](#) para solicitar acceso al contenedor.

El formulario solicita información acerca del usuario y de su empresa, así como del escenario de usuario para el que se va a usar el contenedor. Después de haber enviado el formulario, el equipo de Azure Cognitive Services lo revisa para asegurarse de que cumple los criterios de acceso al registro de contenedor privado.

### IMPORTANT

Debe usar una dirección de correo electrónico que esté asociada con una cuenta de Microsoft (MSA) o de Azure Active Directory (Azure AD) en el formulario.

Si se aprueba la solicitud, recibirá un correo electrónico con instrucciones que describen cómo obtener las credenciales y tener acceso al registro de contenedor privado.

## Uso de la CLI de Docker para autenticar un registro de contenedor privado

Puede autenticarse con el registro de contenedor privado de contenedores de Cognitive Services, pero el método recomendado de la línea de comandos es mediante el uso de la [CLI de Docker](#).

Use el comando `docker login`, como se muestra en el ejemplo siguiente, para iniciar sesión en `containerpreview.azurecr.io`, el registro de contenedor privado de los contenedores de Cognitive Services.

Reemplace `<username>` por el nombre de usuario y `<password>` por la contraseña proporcionada en las credenciales que recibió del equipo de Azure Cognitive Services.

```
docker login containerpreview.azurecr.io -u <username> -p <password>
```

Si ha protegido las credenciales en un archivo de texto, puede concatenar el contenido de dicho archivo de texto, mediante el comando `cat`, al comando `docker login`, tal como se muestra en el ejemplo siguiente. Reemplace `<passwordFile>` por la ruta de acceso y el nombre del archivo de texto que contiene la contraseña, y `<username>` por el nombre de usuario proporcionado en las credenciales.

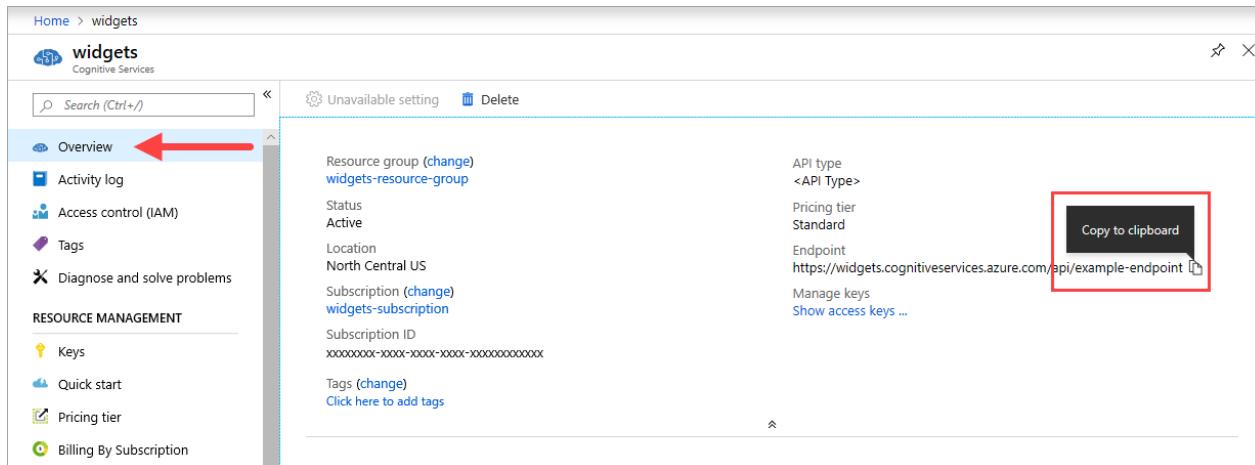
```
cat <passwordFile> | docker login containerpreview.azurecr.io -u <username> --password-stdin
```

## Recopilación de los parámetros obligatorios

Hay tres parámetros principales para todos los contenedores de Cognitive Services que son necesarios. El contrato de licencia para el usuario final (CLUF) debe estar presente con un valor de `accept`. Además, se necesitan una dirección URL de punto de conexión y una clave de API.

### URI de punto de conexión {ENDPOINT\_URI}

El valor del URI del **punto de conexión** está disponible en la página *Información general* de Azure Portal del recurso de Cognitive Services correspondiente. Vaya a la página *Información general*, mantenga el cursor sobre el punto de conexión y aparecerá un icono `Copy to clipboard`. Cópielo y utilícelo cuando sea necesario.



### Claves {API\_KEY}

Esta clave se usa para iniciar el contenedor y está disponible en la página de claves de Azure Portal del recurso de Cognitive Services correspondiente. Vaya a la página *Claves* y haga clic en el icono `Copy to clipboard`.

The screenshot shows the Azure Cognitive Services Keys page for the 'widgets' resource. On the left, there's a sidebar with options like Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, RESOURCE MANAGEMENT, and Keys. The 'Keys' option is highlighted with a red arrow. The main area shows a 'NAME' input field with 'widgets'. Below it is an information box about subscription keys. Under 'KEY 1', there's an input field containing '<key 1 value>' with a red box around it. There's also a 'KEY 2' input field below it.

## IMPORTANT

Estas claves de suscripción se usan para tener acceso a la API de Cognitive Services. No comparta las claves. Almacénelas de forma segura, por ejemplo, con Azure Key Vault. También se recomienda regenerar estas claves periódicamente. Solo se necesita una clave para realizar una llamada API. Al volver a generar la primera clave, puede usar la segunda clave para seguir teniendo acceso al servicio.

## El equipo host

El host es un equipo basado en x64 que ejecuta el contenedor de Docker. Puede ser un equipo del entorno local o un servicio de hospedaje de Docker incluido en Azure, como:

- [Azure Kubernetes Service](#).
- [Azure Container Instances](#).
- Un clúster de [Kubernetes](#) implementado en [Azure Stack](#). Para obtener más información, consulte [Implementación de Kubernetes en Azure Stack](#).

### Compatibilidad con la extensión avanzada de vector

El **host** es el equipo que ejecuta el contenedor de Docker. El host *debe ser compatible* con las [extensiones avanzadas de vector](#) (AVX2). Puede comprobar la compatibilidad con AVX2 en los hosts de Linux con el comando siguiente:

```
grep -q avx2 /proc/cpuinfo && echo AVX2 supported || echo No AVX2 support detected
```

## WARNING

El equipo host es *necesario* para admitir AVX2. El contenedor *no* funcionará correctamente sin compatibilidad con AVX2.

## Recomendaciones y requisitos del contenedor

En la tabla siguiente se describe la asignación mínima y recomendada de recursos para cada contenedor de Voz.

- [Voz a texto](#)
- [Conversión de voz a texto personalizada](#)
- [Texto a voz](#)
- [Conversión de texto a voz personalizada](#)

CONTENEDOR	MÍNIMA	RECOMENDADO
Voz a texto	2 núcleos, 2 GB de memoria	4 núcleos, 4 GB de memoria

- Cada núcleo debe ser de 2,6 gigahercios (GHz) como mínimo.

El núcleo y la memoria se corresponden con los valores de `--cpus` y `--memory` que se usan como parte del comando `docker run`.

#### NOTE

Los valores mínimos y recomendados se basan en los límites de Docker y *no* en los recursos de la máquina host. Por ejemplo, partes de la asignación de memoria de contenedores de voz a texto de un modelo grande de lenguaje, y se *recomienda* que todo el archivo se ajuste en memoria, que es de 4 a 6 GB adicionales. Además, la primera ejecución de cualquier contenedor puede tardar más, dado que los modelos se van a paginar en la memoria.

## Obtención de la imagen del contenedor con `docker pull`

Las imágenes de contenedor para Voz están disponibles en la instancia de Container Registry siguiente.

- [Voz a texto](#)
- [Conversión de voz a texto personalizada](#)
- [Texto a voz](#)
- [Conversión de texto a voz personalizada](#)

CONTENEDOR	REPOSITORIO
Voz a texto	<code>containerpreview.azurecr.io/microsoft/cognitive-services-speech-to-text:latest</code>

#### TIP

Puede usar el comando `docker images` para enumerar las imágenes de contenedor descargadas. Por ejemplo, el comando siguiente muestra el id., el repositorio y la etiqueta de cada imagen de contenedor descargada, con formato de tabla:

```
docker images --format "table {{.ID}}\t{{.Repository}}\t{{.Tag}}"
IMAGE ID      REPOSITORY          TAG
<image-id>   <repository-path/name> <tag-name>
```

## Docker pull para los contenedores de Voz

- [Voz a texto](#)
- [Conversión de voz a texto personalizada](#)
- [Texto a voz](#)
- [Conversión de texto a voz personalizada](#)

### Docker pull para el contenedor de conversión de voz a texto

Use el comando `docker pull` para descargar una imagen de contenedor desde la versión preliminar del registro de contenedor.

```
docker pull containerpreview.azurecr.io/microsoft/cognitive-services-speech-to-text:latest
```

## IMPORTANT

La etiqueta `latest` extrae la configuración regional `en-US`. Para otras configuraciones regionales, consulte [Configuración regional de conversión de voz a texto](#).

### Configuraciones regionales de voz a texto

Todas las etiquetas, a excepción de `latest` tienen el formato siguiente y distinguen mayúsculas de minúsculas:

```
<major>.<minor>.<patch>-<platform>-<locale>-<prerelease>
```

La etiqueta siguiente es un ejemplo del formato:

```
2.0.0-amd64-en-us-preview
```

Para ver todas las configuraciones regionales admitidas del contenedor de **conversión de voz a texto**, consulte las [etiquetas de imágenes de la conversión de voz a texto](#).

## Uso del contenedor

Una vez que el contenedor esté en el [equipo host](#), utilice el siguiente proceso para trabajar con el contenedor.

1. Ejecute el contenedor con la configuración de facturación requerida. Hay más [ejemplos](#) del comando `docker run` disponibles.
2. Consulta del punto de conexión de predicción del contenedor.

### Ejecute el contenedor con `docker run`.

Utilice el comando `docker run` para ejecutar el contenedor. Consulte [Recopilación de los parámetros obligatorios](#) para más información sobre cómo obtener los valores de `{Endpoint_URI}` y `{API_Key}`. También hay disponibles otros [ejemplos](#) del comando `docker run`.

- [Voz a texto](#)
- [Conversión de voz a texto personalizada](#)
- [Texto a voz](#)
- [Conversión de texto a voz personalizada](#)

Para ejecutar el contenedor *Conversión de voz a texto*, ejecute el comando `docker run` siguiente.

```
docker run --rm -it -p 5000:5000 --memory 4g --cpus 4 \
containerpreview.azurecr.io/microsoft/cognitive-services-speech-to-text \
Eula=accept \
Billing={ENDPOINT_URI} \
ApiKey={API_KEY}
```

Este comando:

- Ejecuta un contenedor *Conversión de voz a texto* desde la imagen de contenedor.
- Asigna 4 núcleos de CPU y 4 gigabytes (GB) de memoria.
- Expone el puerto TCP 5000 y asigna un seudo-TTY para el contenedor.
- Una vez que se produce la salida, quita automáticamente el contenedor. La imagen del contenedor sigue estando disponible en el equipo host.

#### IMPORTANT

Para poder ejecutar el contenedor, las opciones `Eula`, `Billing` y `ApiKey` deben estar especificadas; de lo contrario, el contenedor no se iniciará. Para obtener más información, vea [Facturación](#).

## Consulta del punto de conexión de predicción del contenedor

CONTENEDORES	DIRECCIÓN URL DEL HOST DEL SDK	PROTOCOLO
Conversión de voz en texto y conversión de voz en texto personalizada	<code>ws://localhost:5000</code>	WS
Conversión de texto a voz y conversión de texto a voz personalizada	<code>http://localhost:5000</code>	HTTP

Para más información sobre cómo usar los protocolos WSS y HTTPS, consulte la [seguridad del contenedor](#).

### Conversión de voz a texto o Conversión de voz a texto personalizada

El contenedor proporciona las API de punto de conexión de consulta basadas en WebSocket, a las que se accede mediante el [SDK de Voz](#). De forma predeterminada, el SDK de Voz usa servicios de voz en línea. Para usar el contenedor, deberá cambiar el método de inicialización.

#### TIP

Al usar el SDK de Voz con los contenedores, no es necesario proporcionar la [clave de suscripción del recurso de Voz de Azure](#) o un [token de portador de autenticación](#).

Consulte los ejemplos siguientes.

- [C#](#)
- [Python](#)

Cambie de usar esta llamada de inicialización en la nube de Azure:

```
var config = SpeechConfig.FromSubscription("YourSubscriptionKey", "YourServiceRegion");
```

por esta llamada mediante el `host` de contenedor:

```
var config = SpeechConfig.FromHost(  
    new Uri("ws://localhost:5000"));
```

### Conversión de texto a voz o conversión de texto a voz personalizada

El contenedor proporciona [API de punto de conexión basadas en REST](#). Hay muchos [proyectos de código fuente de ejemplo](#) disponibles para las variaciones de lenguaje, marco y plataforma.

Con el contenedor *Conversión de texto a voz estándar*, debe basarse en la configuración regional y en la voz de la etiqueta de imagen que descargó. Por ejemplo, si descargó la etiqueta `latest`, la configuración regional predeterminada es `en-US` y la voz `JessaRUS`. El argumento `{VOICE_NAME}` sería `en-US-JessaRUS`. Vea el SSML de ejemplo siguiente:

```

<speak version="1.0" xml:lang="en-US">
  <voice name="en-US-JessaRUS">
    This text will get converted into synthesized speech.
  </voice>
</speak>

```

Sin embargo, para *Conversión de texto a voz personalizada*, tendrá que obtener el valor de **Voice / model** desde el [portal de Voz personalizada](#). El nombre del modelo personalizado es sinónimo del nombre de la voz. Vaya a la página de **entrenamiento** y copie el valor de **Voice / model** que se va a usar como el argumento `{VOICE_NAME}`.

Vea el SSML de ejemplo siguiente:

```

<speak version="1.0" xml:lang="en-US">
  <voice name="custom-voice-model">
    This text will get converted into synthesized speech.
  </voice>
</speak>

```

Vamos a crear una solicitud HTTP POST, proporcionando algunos encabezados y una carga de datos. Reemplace el marcador de posición `{VOICE_NAME}` por un valor propio.

```

curl -s -v -X POST http://localhost:5000/speech/synthesize/cognitiveservices/v1 \
-H 'Accept: audio/*' \
-H 'Content-Type: application/ssml+xml' \
-H 'X-Microsoft-OutputFormat: riff-16khz-16bit-mono-pcm' \
-d '<speak version="1.0" xml:lang="en-US"><voice name="{VOICE_NAME}">This is a test, only a test.</voice>
</speak>''

```

Este comando:

- Construye una solicitud HTTP POST para el punto de conexión `speech/synthesize/cognitiveservices/v1`.
- Especifica un encabezado `Accept` de `audio/*`
- Especifica un encabezado `Content-Type` de `application/ssml+xml`. Para más información, consulte el [cuerpo de la solicitud](#).
- Especifica un encabezado `X-Microsoft-OutputFormat` de `riff-16khz-16bit-mono-pcm`. Para más opciones, consulte la [salida de audio](#).
- Envía la solicitud [Lenguaje de marcado de síntesis de voz \(SSML\)](#) dado el `{VOICE_NAME}` al punto de conexión.

## Ejecución de varios contenedores en el mismo host

Si tiene pensado ejecutar varios contenedores con puertos expuestos, asegúrese de que ejecuta cada contenedor con un puerto expuesto diferente. Por ejemplo, ejecute el primer contenedor en el puerto 5000 y el segundo en el

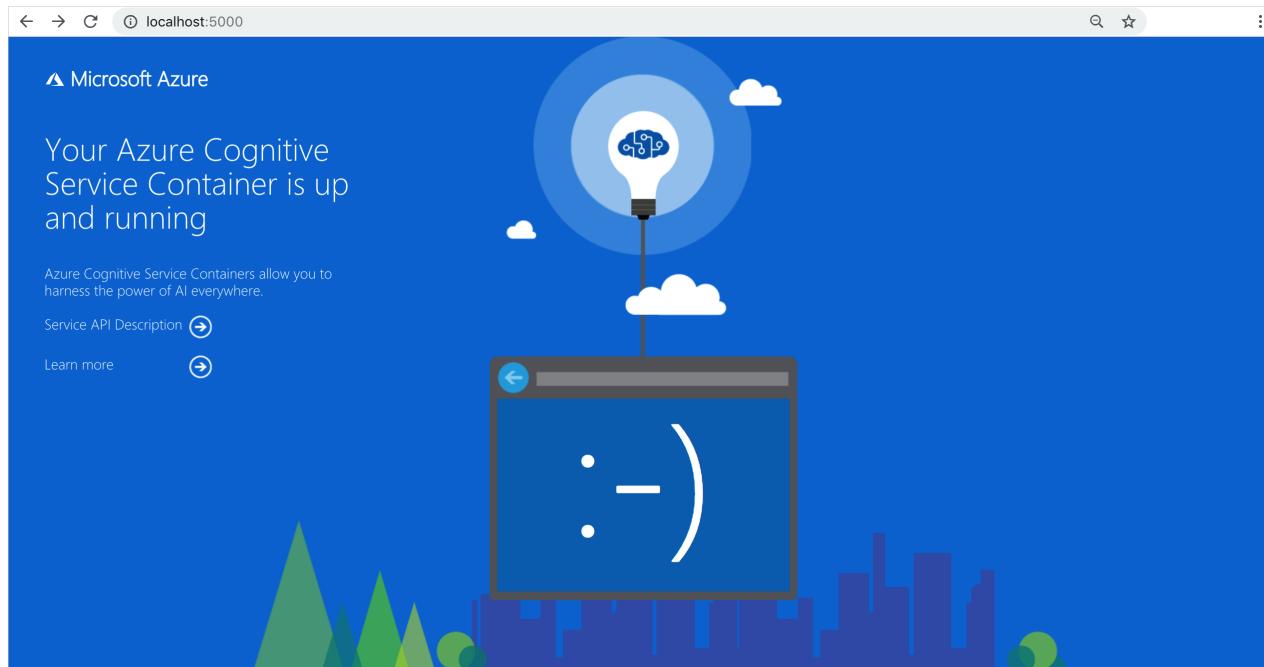
puerto 5001.

Puede tener este contenedor y un contenedor de Azure Cognitive Services diferente en ejecución simultáneamente en el HOST. También puede tener varios contenedores del mismo contenedor de Cognitive Services en ejecución.

## Comprobación de que un contenedor está en ejecución

Hay varias maneras de comprobar que el contenedor está en ejecución. Busque la dirección *IP externa* y el puerto expuesto del contenedor en cuestión y abra el explorador web que prefiera. Use las distintas direcciones URL de solicitud para validar que el contenedor se está ejecutando. Las direcciones URL de solicitud de ejemplo que se enumeran a continuación son `http://localhost:5000`, pero el contenedor específico puede variar. Tenga en cuenta que va a confiar en la dirección *IP externa* y el puerto expuesto del contenedor.

URL DE LA SOLICITUD	PROPOSITO
<code>http://localhost:5000/</code>	El contenedor ofrece una página principal.
<code>http://localhost:5000/status</code>	Se solicitó con HTTP GET, para comprobar que el contenedor está en ejecución sin causar una consulta al punto de conexión. Esta solicitud se puede usar con los <a href="#">sondeos de ejecución y preparación</a> de Kubernetes.
<code>http://localhost:5000/swagger</code>	El contenedor cuenta con un completo conjunto de documentación sobre los puntos de conexión y una característica de <b>prueba</b> . Esta característica le permite especificar la configuración en un formulario HTML basado en web y realizar la consulta sin necesidad de escribir código. Una vez que la consulta devuelve resultados, se proporciona un ejemplo del comando CURL para mostrar los encabezados HTTP y el formato de cuerpo requeridos.



## Detención del contenedor

Para apagar el contenedor, en el entorno de la línea de comandos donde se ejecuta el contenedor, seleccione `Ctrl + C`.

# Solución de problemas

Puede encontrar algunos problemas al iniciar o ejecutar el contenedor. Use un [montaje](#) de salida y habilite el registro. Si lo hace, permitirá que el contenedor genere archivos de registro que son útiles para solucionar problemas.

## TIP

Para más información e instrucciones para solución de problemas, vea [Preguntas frecuentes de los contenedores de Cognitive Services](#).

## Facturación

Los contenedores de Voz envían información de facturación a Azure mediante un recurso de Voz en la cuenta de Azure.

Las consultas en el contenedor se facturan con el plan de tarifa del recurso de Azure que se usa para `<ApiKey>`.

Los contenedores de Azure Cognitive Services no tienen licencia para ejecutarse si no están conectados al punto de conexión de facturación para las mediciones. Debe habilitar los contenedores para que comuniquen la información de facturación al punto de conexión de facturación en todo momento. Los contenedores de Cognitive Services no envían datos de los clientes (por ejemplo, la imagen o el texto que se está analizando) a Microsoft.

### Conexión con Azure

El contenedor necesita que se ejecuten los valores del argumento de facturación. Estos valores permiten al contenedor conectarse al punto de conexión de facturación. El contenedor informa sobre el uso cada 10 a 15 minutos. Si el contenedor no se conecta a Azure en la ventana de tiempo permitida, continuará ejecutándose pero no atenderá las consultas hasta que se restaure el punto de conexión de facturación. Se intenta 10 veces la conexión en el mismo intervalo de tiempo de 10 a 15 minutos. Si no se puede conectar con el punto de conexión de facturación en esos 10 intentos, el contenedor deja de ejecutarse.

### Argumentos de facturación

Para que el comando `docker run` inicie el contenedor, se deben especificar las tres opciones siguientes se deben especificar con valores válidos:

OPCIÓN	DESCRIPCIÓN
<code>ApiKey</code>	La clave de API del recurso de Cognitive Services que se usa para realizar un seguimiento de la información de facturación. El valor de esta opción se debe establecer en una clave de API para el recurso aprovisionado que se especifica en <code>Billing</code> .
<code>Billing</code>	El punto de conexión del recurso de Cognitive Services que se usa para realizar el seguimiento de la información de facturación. El valor de esta opción debe establecerse en el URI del punto de conexión de un recurso aprovisionado de Azure.
<code>Eula</code>	Indica que ha aceptado la licencia del contenedor. El valor de esta opción debe establecerse en <code>accept</code> .

Para obtener más información acerca de estas opciones, consulte [Configure containers \(Configuración de contenedores\)](#).

## Publicaciones de blog

- Ejecución de contenedores de Cognitive Services
- Azure Cognitive Services

## Ejemplos para desarrolladores

Hay ejemplos para desarrolladores disponibles en nuestro [repositorio de GitHub](#).

## Ver seminario web

Únase al [seminario web](#) para más información sobre:

- Implementación de Cognitive Services en cualquier máquina con Docker
- Implementación de Cognitive Services en AKS

## Resumen

En este artículo, ha aprendido los conceptos y el flujo de trabajo para la descarga, instalación y ejecución de contenedores de Voz. En resumen:

- Voz proporciona cuatro contenedores Linux para Docker, que encapsulan varias funcionalidades:
  - *Voz a texto*
  - *Conversión de voz a texto personalizada*
  - *Texto a voz*
  - *Conversión de texto a voz personalizada*
- Las imágenes de contenedor se descargan desde el registro de contenedor de Azure.
- Las imágenes de contenedor se ejecutan en Docker.
- Puede usar la API REST o el SDK para llamar a operaciones en contenedores de Voz mediante la especificación del URI del host del contenedor.
- Debe proporcionar la información de facturación al crear una instancia de un contenedor.

### IMPORTANT

Los contenedores de Cognitive Services no tienen licencia para ejecutarse sin estar conectados a Azure para realizar mediciones. Los clientes tienen que habilitar los contenedores para comunicar la información de facturación con el servicio de medición en todo momento. Los contenedores de Cognitive Services no envían datos de los clientes (por ejemplo, la imagen o el texto que se está analizando) a Microsoft.

## Pasos siguientes

- Revise [Configuración de contenedores](#) para ver las opciones de configuración.
- Aprenda a [usar contenedores del servicio de voz con Kubernetes y Helm](#).
- Uso de otros [contenedores de Cognitive Services](#)

# Configuración de contenedores del servicio de voz

13/01/2020 • 20 minutes to read • [Edit Online](#)

Los contenedores de Voz permiten a los clientes compilar una arquitectura de aplicación de voz optimizada para aprovechar las sólidas capacidades de la nube y la localidad del perímetro. Los cuatro contenedores de voz compatibles ahora son **conversión de voz a texto**, **conversión de voz a texto personalizada**, **conversión de texto a voz** y **conversión de texto a voz personalizada**.

El entorno en tiempo de ejecución del contenedor de **Speech** se configura mediante los argumentos del comando `docker run`. Este contenedor tiene varias opciones de configuración necesarias, así como otras opcionales. Hay disponibles varios [ejemplos](#) del comando. La configuración específica del contenedor es la configuración de facturación.

## Valores de configuración

Este contenedor tiene las siguientes opciones de configuración:

OBLIGATORIO	CONFIGURACIÓN	PROPÓSITO
Sí	<a href="#">ApiKey</a>	Realiza el seguimiento de la información de facturación.
Sí	<a href="#">Application Insights</a>	Permite agregar compatibilidad con los datos de telemetría de <a href="#">Azure Application Insights</a> al contenedor.
Sí	<a href="#">Facturación</a>	Especifica el URI del punto de conexión del recurso de servicio en Azure.
Sí	<a href="#">Eula</a>	Indica que ha aceptado la licencia del contenedor.
Sí	<a href="#">Fluentd</a>	Escribe el registro y, opcionalmente, los datos de métricas en un servidor de Fluentd.
Sí	<a href="#">Proxy HTTP</a>	Configura un proxy HTTP para realizar solicitudes de salida.
Sí	<a href="#">Logging</a>	Proporciona compatibilidad con el registro de ASP.NET Core al contenedor.
Sí	<a href="#">Mounts</a>	Lee y escribe los datos desde el equipo host al contenedor y del contenedor de nuevo al equipo host.

### IMPORTANT

Las opciones [ApiKey](#), [Billing](#) y [Eula](#) se usan en conjunto y debe proporcionar valores válidos para las tres; en caso contrario, no se inicia el contenedor. Para obtener más información sobre el uso de estas opciones de configuración para crear instancias de un contenedor, consulte [Facturación](#).

## Opción de configuración ApiKey

La opción de configuración `ApiKey` especifica la clave de recurso de Azure usada para realizar un seguimiento de la información de facturación del contenedor. Debe especificar un valor para `ApiKey` que debe ser una clave válida para el recurso de `Speech` especificado para la opción de configuración `Billing`.

Este valor se puede encontrar en el siguiente lugar:

- Azure Portal: Administración de recursos de **Speech** en **Claves**

## Opción de configuración ApplicationInsights

La opción de configuración `ApplicationInsights` le permite agregar compatibilidad con los datos de telemetría de [Azure Application Insights](#) al contenedor. Application Insights proporciona una supervisión detallada del contenedor. Puede supervisar fácilmente la disponibilidad, el rendimiento y el uso del contenedor. También puede identificar y diagnosticar errores en el contenedor rápidamente.

En la tabla siguiente se describen las opciones de configuración compatibles en la sección `ApplicationInsights`.

OBLIGATORIO	NOMBRE	TIPO DE DATOS	DESCRIPCIÓN
Sin	<code>InstrumentationKey</code>	Cadena	<p>Clave de instrumentación de la instancia de Application Insights para la que se envían los datos de telemetría del contenedor. Para más información, consulte <a href="#">Application Insights para ASP.NET Core</a>.</p> <p>Ejemplo: <code>InstrumentationKey=123456789</code></p>

## Opción de configuración Billing

La configuración `Billing` especifica el URI de punto de conexión del recurso de `Speech` en Azure que se usa para medir la información de facturación del contenedor. Debe especificar un valor para esta opción de configuración y este debe ser un URI de punto de conexión válido para un recurso de `Speech` en Azure. El contenedor informa sobre el uso cada 10 a 15 minutos.

Este valor se puede encontrar en el siguiente lugar:

- Azure Portal: Información general de **Speech**, con la etiqueta `Endpoint`

OBLIGATORIO	NOMBRE	TIPO DE DATOS	DESCRIPCIÓN
Sí	<code>Billing</code>	Cadena	<p>Identificador URI del punto de conexión de facturación. Para más información sobre cómo obtener el URI de facturación, consulte la <a href="#">recopilación de los parámetros necesarios</a>. Para más información y para obtener una lista completa de los puntos de conexión regionales, consulte <a href="#">Nombres de subdominios personalizados para Cognitive Services</a>.</p>

## Opción de configuración Eula

La opción de configuración `Eula` indica que ha aceptado la licencia del contenedor. Debe especificar un valor para esta

opción de configuración y el valor debe establecerse en `accept`.

OBLIGATORIO	NOMBRE	TIPO DE DATOS	DESCRIPCIÓN
Sí	<code>Eula</code>	Cadena	Aceptación de la licencia Ejemplo: <code>Eula=accept</code>

Los contenedores de Cognitive Services tienen una licencia sujeta al [contrato](#) que rige el uso de Azure. Si no tiene ningún contrato que rija el uso de Azure, acepta que el contrato que rige el uso de Azure es el [Contrato Microsoft Online Subscription](#), que incorpora los [Términos de Online Services](#). En el caso de las versiones preliminares, acepta también los [Términos de uso complementarios para las versiones preliminares de Microsoft Azure](#). Al usar el contenedor, acepta estos términos.

## Opción de configuración Fluentd

Fluentd es un recopilador de datos de código abierto para el registro unificado. La opción de configuración `Fluentd` administra la conexión del contenedor a un servidor de [Fluentd](#). En el contenedor, se incluye un proveedor de registros de Fluentd que permite que el contenedor escriba los registros y, opcionalmente, los datos de métricas en un servidor de Fluentd.

En la tabla siguiente se describen las opciones de configuración compatibles en la sección `Fluentd`.

NOMBRE	TIPO DE DATOS	DESCRIPCIÓN
<code>Host</code>	Cadena	Dirección IP o nombre de host DNS del servidor de Fluentd.
<code>Port</code>	Entero	Puerto del servidor de Fluentd. El valor predeterminado es 24 224.
<code>HeartbeatMs</code>	Entero	Intervalo de latidos (en milisegundos). Si no se envía ningún tráfico de evento antes de que este intervalo expire, se envía un latido al servidor de Fluentd. El valor predeterminado es 60 000 milisegundos (1 minuto).
<code>SendBufferSize</code>	Entero	Espacio en búfer de red (en bytes) asignado para las operaciones de envío. El valor predeterminado es 32 768 bytes (32 kilobytes).
<code>TlsConnectionEstablishmentTimeoutMs</code>	Entero	Tiempo de expiración (en milisegundos) para establecer una conexión SSL/TLS con el servidor de Fluentd. El valor predeterminado es 10 000 milisegundos (10 segundos). Si <code>useTLS</code> se establece en <code>false</code> , este valor se ignora.
<code>UseTLS</code>	Boolean	Indica si el contenedor debe utilizar SSL/TLS para comunicarse con el servidor de Fluentd. El valor predeterminado es <code>false</code> .

## Configuración de las credenciales del proxy HTTP

Si necesita configurar un proxy HTTP para realizar solicitudes de salida, use estos dos argumentos:

NOMBRE	TIPO DE DATOS	DESCRIPCIÓN
HTTP_PROXY	string	El proxy que se va a utilizar, por ejemplo, <code>&lt;proxy-url&gt;</code>
HTTP_PROXY_CREDS	string	Las credenciales necesarias para autenticarse en el proxy, por ejemplo, nombre de usuario:contraseña.
<code>&lt;proxy-user&gt;</code>	string	El usuario del proxy.
<code>&lt;proxy-password&gt;</code>	string	La contraseña asociada con <code>&lt;proxy-user&gt;</code> para el proxy.

```
docker run --rm -it -p 5000:5000 \
--memory 2g --cpus 1 \
--mount type=bind,src=/home/azureuser/output,target=/output \
<registry-location>/<image-name> \
Eula=accept \
Billing=<endpoint> \
ApiKey=<api-key> \
HTTP_PROXY=<proxy-url> \
HTTP_PROXY_CREDS=<proxy-user>:<proxy-password> \
```

## Opción de configuración Logging

La opción de configuración `Logging` administra la compatibilidad con el registro de ASP.NET Core del contenedor. Puede usar los mismos valores y opciones de configuración para el contenedor que los que usa para una aplicación ASP.NET Core.

Los siguientes proveedores de registro son compatibles con el contenedor:

PROVEEDOR	PROPÓSITO
Console	Proveedor de registro de <code>Console</code> de ASP.NET Core. Se admiten todos los valores predeterminados y las opciones de configuración de ASP.NET Core para este proveedor de registro.
Depuración	Proveedor de registro de <code>Debug</code> de ASP.NET Core. Se admiten todos los valores predeterminados y las opciones de configuración de ASP.NET Core para este proveedor de registro.
Disco	Proveedor de registro JSON. Este proveedor de registro escribe datos de registro para el montaje de salida.

Este comando de contenedor almacena información de registro en formato JSON en el montaje de salida:

```
docker run --rm -it -p 5000:5000 \
--memory 2g --cpus 1 \
--mount type=bind,src=/home/azureuser/output,target=/output \
<registry-location>/<image-name> \
Eula=accept \
Billing=<endpoint> \
ApiKey=<api-key> \
Logging:Disk:Format=json
```

Este comando de contenedor muestra información de depuración, con el prefijo `dbug` mientras el contenedor se

ejecuta:

```
docker run --rm -it -p 5000:5000 \
--memory 2g --cpus 1 \
<registry-location>/<image-name> \
Eula=accept \
Billing=<endpoint> \
ApiKey=<api-key> \
Logging:Console:LogLevel:Default=Debug
```

## Registro del disco

El proveedor de registro `Disk` admite la configuración siguiente:

NOMBRE	TIPO DE DATOS	DESCRIPCIÓN
<code>Format</code>	Cadena	Formato de salida de los archivos de registro. <b>Nota:</b> Este valor debe establecerse en <code>json</code> para habilitar el proveedor de registro. Si se especifica este valor sin especificar también un montaje de salida al crear una instancia de un contenedor, se produce un error.
<code>MaxFileSize</code>	Entero	Tamaño máximo en megabytes (MB) de un archivo de registro. Cuando el tamaño del archivo de registro actual cumple este valor o lo supera, el proveedor de registro inicia un nuevo archivo de registro. Si se especifica <code>-1</code> , el tamaño del archivo de registro solo está limitado por el tamaño máximo de archivo, si existe, para el montaje de salida. El valor predeterminado es <code>1</code> .

Para obtener más información acerca de cómo configurar la compatibilidad con el registro de ASP.NET Core, consulte [Configuración del archivo de configuración](#).

## Configuración de montaje

Utilice montajes de enlace para leer y escribir datos hacia y desde el contenedor. Puede especificar un montaje de entrada o un montaje de salida mediante la opción `--mount` del comando `docker run`.

Los contenedores de Voz estándar no usan montajes de entrada o salida para almacenar datos de entrenamiento o servicio. Sin embargo, los contenedores de Voz personalizados dependen de los montajes de volumen.

La sintaxis exacta de la ubicación de montaje del host varía según el sistema operativo del host. Además, la ubicación de montaje del [equipo host](#) puede no estar accesible debido a un conflicto entre los permisos que utiliza la cuenta de servicio de Docker y los permisos de la ubicación de montaje del host.

OPCIONAL	NOMBRE	TIPO DE DATOS	DESCRIPCIÓN
No permitida	<code>Input</code>	Cadena	Los contenedores de Voz estándar no usan esto. Los contenedores de Voz personalizados usan los <a href="#">montajes de volumen</a> .

OPCIONAL	NOMBRE	TIPO DE DATOS	DESCRIPCIÓN
Opcional	<code>output</code>	Cadena	<p>Destino del montaje de salida. El valor predeterminado es <code>/output</code>. Esta es la ubicación de los registros. Esto incluye los registros de contenedor.</p> <p>Ejemplo:  <code>--mount type=bind,src=c:\output,target=/o</code></p>

## Configuración de montaje de volumen

Los contenedores de Voz personalizados usan [montajes de volumen](#) para conservar los modelos personalizados. Para especificar un montaje de volumen, agregue la opción `-v` (o `--volume`) al comando [docker run](#).

Los modelos personalizados se descargan la primera vez que se ingesta un nuevo modelo como parte del comando docker run del contenedor de Voz personalizado. Las ejecuciones secuenciales del mismo `ModelId` para un contenedor de Voz personalizado usarán el modelo descargado anteriormente. Si no se proporciona el montaje de volumen, no se pueden conservar los modelos personalizados.

La configuración de montaje de volumen consta de tres campos `:` de colores separados:

1. El primer campo es el nombre del volumen en la máquina host, por ejemplo, `C:\input`.
2. El segundo campo es el directorio del contenedor, por ejemplo `/usr/local/models`.
3. El tercer campo (opcional) es una lista de opciones separada por comas. Para más información, consulte el [uso de los volúmenes](#).

### Ejemplo de montaje de volumen

```
-v C:\input:/usr/local/models
```

Este comando monta el directorio `C:\input` de la máquina host en el directorio `/usr/local/models` de los contenedores.

#### IMPORTANT

La configuración del montaje de volumen solo se aplica a los contenedores **Conversión de voz a texto personalizada** y **Conversión de texto a voz personalizada**. Los contenedores **Conversión de voz a texto** y **Conversión de texto a voz** estándar no usan los montajes de volumen.

## Comandos de ejemplo de docker run

Los ejemplos siguientes usan las opciones de configuración para ilustrar cómo escribir y usar comandos `docker run`. Una vez que se está ejecutando, el contenedor continúa ejecutándose hasta que lo [detenga](#).

- **Carácter de continuación de línea:** Los comandos de Docker de las secciones siguientes usan la barra diagonal inversa (`\`) como un carácter de continuación de línea. Puede quitarla o reemplazarla en función de los requisitos del sistema operativo del host.
- **Orden de los argumentos:** No cambie el orden de los argumentos a menos que esté familiarizado con los contenedores de Docker.

Reemplace `{argument_name}` por sus propios valores:

MARCADOR DE POSICIÓN	VALOR	FORMATO O EJEMPLO
----------------------	-------	-------------------

MARCADOR DE POSICIÓN	VALOR	FORMATO O EJEMPLO
{CLAVE_API}	La clave del punto de conexión del recurso <b>Speech</b> en la página Claves de <b>Speech</b> de Azure.	xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
{URI_PUNTODECONEXIÓN}	El valor del punto de conexión de facturación está disponible en la página Información general de Azure <b>Speech</b> .	Consulte el apartado de <a href="#">recopilación de los parámetros necesarios</a> para ejemplos explícitos.

#### NOTE

Los nuevos recursos creados después del 1 de julio de 2019 usarán nombres de subdominio personalizados. Para más información y para obtener una lista completa de los puntos de conexión regionales, consulte [Nombres de subdominios personalizados para Cognitive Services](#).

#### IMPORTANT

Para poder ejecutar el contenedor, las opciones **Eula**, **Billing** y **ApiKey** deben estar especificadas; de lo contrario, el contenedor no se iniciará. Para obtener más información, vea [Facturación](#). El valor de ApiKey es la **clave** de la página de claves de recursos de Azure Speech.

## Ejemplos de Docker del contenedor de Speech

En los siguientes ejemplos de Docker encontrará el contenedor de Speech.

- [Voz a texto](#)
- [Conversión de voz a texto personalizada](#)
- [Texto a voz](#)
- [Conversión de texto a voz personalizada](#)

### Ejemplo básico de Conversión de voz a texto

```
docker run --rm -it -p 5000:5000 --memory 4g --cpus 4 \
containerpreview.azurecr.io/microsoft/cognitive-services-speech-to-text \
Eula=accept \
Billing={ENDPOINT_URI} \
ApiKey={API_KEY}
```

### Ejemplo de registro de Conversión de voz a texto

```
docker run --rm -it -p 5000:5000 --memory 4g --cpus 4 \
containerpreview.azurecr.io/microsoft/cognitive-services-custom-speech-to-text \
Eula=accept \
Billing={ENDPOINT_URI} \
ApiKey={API_KEY} \
Logging:Console:LogLevel:Default=Information
```

## Pasos siguientes

- Consulte [Instalación y ejecución de contenedores](#)

# Uso de contenedores del servicio de voz con Kubernetes y Helm

16/01/2020 • 22 minutes to read • [Edit Online](#)

Una opción para administrar los contenedores de Speech locales es usar Kubernetes y Helm. Mediante el empleo de Kubernetes y Helm para definir las imágenes de contenedor de conversión de voz en texto y de texto en voz, se va a crear un paquete de Kubernetes. Este paquete se va a implementar en un clúster de Kubernetes en el entorno local. Por último, se va a explicar cómo probar los servicios implementados y diversas opciones de configuración. Para más información sobre cómo ejecutar contenedores de Docker sin orquestación de Kubernetes, consulte [Instalación y ejecución de contenedores de servicio de voz](#).

## Prerequisites

Requisitos previos para poder usar los contenedores de Voz en el entorno local:

OBLIGATORIO	PROPOSITO
Cuenta de Azure	Si no tiene una suscripción a Azure, cree una <a href="#">cuenta gratuita</a> antes de empezar.
Acceso a Container Registry	Para incorporar las imágenes de Docker al clúster, Kubernetes necesita acceso al registro de contenedor.
CLI de Kubernetes	La <a href="#">CLI de Kubernetes</a> es necesaria para administrar las credenciales compartidas desde el registro de contenedor. Kubernetes también se necesita antes que Helm, que es el administrador de paquetes de Kubernetes.
CLI de Helm	Como parte de la instalación de la <a href="#">CLI de Helm</a> , también tendrá que inicializar Helm, que instalará <a href="#">Tiller</a> .
Recurso de Voz	Para usar estos contenedores, debe tener:  Un recurso de Voz de Azure para obtener la clave de facturación asociada y el URI del punto de conexión de facturación. Ambos valores están disponibles en las páginas de claves y de introducción de <a href="#">Voz</a> de Azure Portal y son necesarios para iniciar el contenedor.  <b>{API_KEY}</b> : clave de recurso  <b>{ENDPOINT_URI}</b> : el ejemplo de URI de punto de conexión es: <code>https://westus.api.cognitive.microsoft.com/sts/v1.0</code>

## Configuración de equipo host recomendada

Vea los detalles del [equipo host del contenedor del servicio de voz](#) como referencia. En este *gráfico de Helm* se calculan automáticamente los requisitos de CPU y memoria en función de cuántas descodificaciones (solicitudes simultáneas) especifica el usuario. Además, se ajusta en función de si las optimizaciones de entrada de audio o texto están configuradas como `enabled`. Los valores predeterminados del gráfico de Helm son dos solicitudes

simultáneas y la deshabilitación de la optimización.

SERVICIO	CPU O CONTENEDOR	MEMORIA O CONTENEDOR
Voz a texto	un descodificador requiere un mínimo de 1150 milésimas de núcleo. Si <code>optimizedForAudioFile</code> está habilitado, se requieren 1950 milésimas de núcleo. (valor predeterminado: dos descodificadores)	Requerido: 2 GB Limitado: 4 GB
Texto a voz	una solicitud simultánea requiere un mínimo de 500 milésimas de núcleo. Si <code>optimizeForTurboMode</code> está habilitado, se requieren 1000 milésimas de núcleo. (valor predeterminado: dos solicitudes simultáneas)	Requerido: 1 GB Limitado: 2 GB

## Conectarse al clúster de Kubernetes

Se espera que el equipo host tenga un clúster de Kubernetes disponible. Vea este tutorial sobre la [implementación de un clúster de Kubernetes](#) para lograr un reconocimiento conceptual de cómo implementar un clúster de Kubernetes en un equipo host.

### Uso compartido de credenciales de Docker con el clúster de Kubernetes

Para permitir que el clúster de Kubernetes `docker pull` las imágenes configuradas del registro de contenedor `containerpreview.azurecr.io`, debe transferir las credenciales de Docker al clúster. Ejecute el comando `kubectl create` siguiente para crear un *secreto de registro de Docker* basado en las credenciales proporcionadas en el requisito previo de acceso al registro de contenedor.

Ejecute el siguiente comando desde la interfaz de la línea de comandos que prefiera. Asegúrese de reemplazar `<username>`, `<password>` y `<email-address>` por las credenciales del registro de contenedor.

```
kubectl create secret docker-registry mcr \
--docker-server=containerpreview.azurecr.io \
--docker-username=<username> \
--docker-password=<password> \
--docker-email=<email-address>
```

#### NOTA

Si ya tiene acceso al registro de contenedor `containerpreview.azurecr.io`, puede crear un secreto de Kubernetes con la marca genérica en su lugar. Tenga en cuenta el siguiente comando que se ejecuta en el JSON de configuración de Docker.

```
kubectl create secret generic mcr \
--from-file=.dockerconfigjson=~/.docker/config.json \
--type=kubernetes.io/dockerconfigjson
```

El siguiente resultado se imprime en la consola una vez que el secreto se ha creado correctamente.

```
secret "mcr" created
```

Para comprobar que el secreto se ha creado, ejecute `kubectl get` con la marca `secrets`.

```
kubectl get secrets
```

Al ejecutar `kubectl get secrets`, se imprimen todos los secretos configurados.

NAME	TYPE	DATA	AGE
mcr	kubernetes.io/dockerconfigjson	1	30s

## Configuración de los valores del gráfico de Helm para la implementación

Visite [Microsoft Helm Hub](#) para ver todos los gráficos de Helm disponibles públicamente que ofrece Microsoft.

En Microsoft Helm Hub se encuentra el **gráfico Cognitive Services Speech On-Premises. Cognitive Services Speech On-Premises** es el gráfico que se va a instalar, aunque primero se debe crear un archivo `config-values.yaml` con configuraciones explícitas. Vamos a comenzar por agregar el repositorio de Microsoft a la instancia de Helm.

```
helm repo add microsoft https://microsoft.github.io/charts/repo
```

Luego vamos a configurar los valores del gráfico de Helm. Copie y pegue el siguiente YAML en un archivo denominado `config-values.yaml`. Para obtener más información sobre la personalización del **gráfico de Helm Cognitive Services Speech On-Premises**, vea [Personalizar gráficos de Helm](#). Reemplace los comentarios `# {ENDPOINT_URI}` y `# {API_KEY}` por sus propios valores.

```
# These settings are deployment specific and users can provide customizations

# speech-to-text configurations
speechToText:
  enabled: true
  numberOfConcurrentRequest: 3
  optimizeForAudioFile: true
  image:
    registry: containerpreview.azurecr.io
    repository: microsoft/cognitive-services-speech-to-text
    tag: latest
    pullSecrets:
      - mcr # Or an existing secret
  args:
    eula: accept
    billing: # {ENDPOINT_URI}
    apikey: # {API_KEY}

# text-to-speech configurations
textToSpeech:
  enabled: true
  numberOfConcurrentRequest: 3
  optimizeForTurboMode: true
  image:
    registry: containerpreview.azurecr.io
    repository: microsoft/cognitive-services-text-to-speech
    tag: latest
    pullSecrets:
      - mcr # Or an existing secret
  args:
    eula: accept
    billing: # {ENDPOINT_URI}
    apikey: # {API_KEY}
```

#### **IMPORTANT**

Si no se proporcionan los valores `billing` y `apikey`, los servicios exíran pasados 15 minutos. Del mismo modo, se produce un error de comprobación, ya que los servicios no están disponibles.

### **Paquete de Kubernetes (gráfico de Helm)**

El *gráfico de Helm* contiene la configuración de las imágenes de Docker que se van a extraer del registro de contenedor `containerpreview.azurecr.io`.

Un [gráfico de Helm](#) es una colección de archivos que describen un conjunto relacionado de recursos de Kubernetes. Un solo gráfico se podría usar para implementar algo sencillo, como un pod almacenado en memoria, o complejo, como una pila de aplicación web completa con servidores HTTP, bases de datos, memorias caché, etc.

Los *gráficos de Helm* proporcionados extraen las imágenes de Docker del servicio de voz, los servicios de conversión de texto en voz y de voz en texto desde el registro de contenedor `containerpreview.azurecr.io`.

### **Instalación del gráfico de Helm en el clúster de Kubernetes**

Para instalar el *gráfico de Helm*, se debe ejecutar el comando `helm install`, reemplazando `<config-values.yaml>` por el argumento de ruta de acceso y nombre de archivo adecuado. El gráfico de Helm `microsoft/cognitive-services-speech-onpremise` al que se hace referencia a continuación está disponible en [Microsoft Helm Hub aquí](#).

```
helm install onprem-speech microsoft/cognitive-services-speech-onpremise \
--version 0.1.1 \
--values <config-values.yaml>
```

Este es el resultado de ejemplo que se puede ver tras una ejecución de instalación correcta:

```

NAME: onprem-speech
LAST DEPLOYED: Tue Jul  2 12:51:42 2019
NAMESPACE: default
STATUS: DEPLOYED

RESOURCES:
==> v1/Pod(related)
NAME                      READY  STATUS           RESTARTS  AGE
speech-to-text-7664f5f465-87w2d  0/1   Pending          0         0s
speech-to-text-7664f5f465-klbr8  0/1   ContainerCreating 0         0s
text-to-speech-56f8fb685b-4jtzh  0/1   ContainerCreating 0         0s
text-to-speech-56f8fb685b-frwxf  0/1   Pending          0         0s

==> v1/Service
NAME            TYPE      CLUSTER-IP     EXTERNAL-IP   PORT(S)      AGE
speech-to-text  LoadBalancer  10.0.252.106 <pending>    80:31811/TCP  1s
text-to-speech  LoadBalancer  10.0.125.187 <pending>    80:31247/TCP  0s

==> v1beta1/PodDisruptionBudget
NAME                  MIN AVAILABLE  MAX UNAVAILABLE ALLOWED DISRUPTIONS  AGE
speech-to-text-poddisruptionbudget  N/A          20%          0             1s
text-to-speech-poddisruptionbudget  N/A          20%          0             1s

==> v1beta2/Deployment
NAME        READY  UP-TO-DATE  AVAILABLE  AGE
speech-to-text  0/2    2          0          0s
text-to-speech  0/2    2          0          0s

==> v2beta2/HorizontalPodAutoscaler
NAME          REFERENCE      TARGETS  MINPODS  MAXPODS  REPLICAS  AGE
speech-to-text-autoscaler  Deployment/speech-to-text  <unknown>/50%  2        10        0         0s
text-to-speech-autoscaler  Deployment/text-to-speech <unknown>/50%  2        10        0         0s

NOTES:
cognitive-services-speech-onpremise has been installed!
Release is named onprem-speech

```

La implementación de Kubernetes puede tardar varios minutos en completarse. Para confirmar que los pods y los servicios se han implementado correctamente y están disponibles, ejecute el siguiente comando:

```
kubectl get all
```

Debería ver algo parecido al resultado siguiente:

NAME	READY	STATUS	RESTARTS	AGE
pod/speech-to-text-7664f5f465-87w2d	1/1	Running	0	34m
pod/speech-to-text-7664f5f465-k1br8	1/1	Running	0	34m
pod/text-to-speech-56f8fb685b-4jtzh	1/1	Running	0	34m
pod/text-to-speech-56f8fb685b-frwxrf	1/1	Running	0	34m

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE
service/kubernetes	ClusterIP	10.0.0.1	<none>	443/TCP	3h
service/speech-to-text	LoadBalancer	10.0.252.106	52.162.123.151	80:31811/TCP	34m
service/text-to-speech	LoadBalancer	10.0.125.187	65.52.233.162	80:31247/TCP	34m

NAME	DESIRED	CURRENT	UP-TO-DATE	AVAILABLE	AGE
deployment.apps/speech-to-text	2	2	2	2	34m
deployment.apps/text-to-speech	2	2	2	2	34m

NAME	DESIRED	CURRENT	READY	AGE
replicaset.apps/speech-to-text-7664f5f465	2	2	2	34m
replicaset.apps/text-to-speech-56f8fb685b	2	2	2	34m

NAME	REFERENCE	TARGETS	MINPODS
MAXPODS			
horizontalpodautoscaler.autoscaling/speech-to-text-autoscaler	Deployment/speech-to-text	1%/50%	2
10	2	34m	
horizontalpodautoscaler.autoscaling/text-to-speech-autoscaler	Deployment/text-to-speech	0%/50%	2
10	2	34m	

## Comprobación de la implementación de Helm con pruebas de Helm

Los gráficos de Helm instalados definen *pruebas de Helm*, que resultan cómodas para la comprobación. Estas pruebas validan la disponibilidad del servicio. Para comprobar los servicios de conversión de **voz en texto** y de **texto en voz**, se ejecuta el comando [Helm test](#).

```
helm test onprem-speech
```

### IMPORTANT

Se produce un error en estas pruebas si el estado de POD no es `Running` o si la implementación no aparece en la columna `AVAILABLE`. Tenga paciencia, ya que esto puede tardar más de diez minutos en completarse.

Estas pruebas generan distintos resultados de estado:

```
RUNNING: speech-to-text-readiness-test
PASSED: speech-to-text-readiness-test
RUNNING: text-to-speech-readiness-test
PASSED: text-to-speech-readiness-test
```

Como alternativa a la ejecución de *pruebas de Helm*, puede recopilar las direcciones *IP externas* y los puertos correspondientes desde el comando `kubectl get all`. Con la dirección IP y el puerto, abra un explorador web y vaya a `http://<external-ip>:<port>/swagger/index.html` para ver las páginas de Swagger de API.

## Personalizar gráficos de Helm

Los gráficos de Helm son jerárquicos. La jerarquía permite la herencia de gráficos y, además, proporciona el concepto de especificidad, que consiste en que los valores más específicos invalidan las reglas heredadas.

### Speech (gráfico de nivel superior)

Los valores del gráfico de nivel superior reemplazan a los valores de los gráficos secundarios correspondientes.

Por lo tanto, todos los valores locales personalizados se deben agregar aquí.

PARÁMETRO	DESCRIPCIÓN	VALOR PREDeterminado
<code>speechToText.enabled</code>	Si el servicio <b>speech-to-text</b> está habilitado.	<code>true</code>
<code>speechToText.verification.enabled</code>	Si la funcionalidad <code>helm test</code> del servicio <b>speech-to-text</b> está habilitada.	<code>true</code>
<code>speechToText.verification.image.registry</code>	El repositorio de imágenes de Docker que <code>helm test</code> usa para probar el servicio <b>speech-to-text</b> . Helm crea un pod independiente dentro del clúster para realizar pruebas y extrae la imagen <i>test-use</i> de este Registro.	<code>docker.io</code>
<code>speechToText.verification.image.repository</code>	El repositorio de imágenes de Docker que <code>helm test</code> usa para probar el servicio <b>speech-to-text</b> . El pod de prueba de Helm usa este repositorio para extraer la imagen <i>test-use</i> .	<code>antsu/on-prem-client</code>
<code>speechToText.verification.image.tag</code>	La etiqueta de la imagen de Docker usada con <code>helm test</code> para el servicio <b>speech-to-text</b> . El pod de prueba de Helm usa esta etiqueta para extraer la imagen <i>test-use</i> .	<code>latest</code>
<code>speechToText.verification.image.pullByHash</code>	Si la imagen de Docker <i>test-use</i> se extrae mediante hash. Si <code>true</code> , debería agregarse <code>speechToText.verification.image.hash</code> con un valor de hash de imagen válido.	<code>false</code>
<code>speechToText.verification.image.arguments</code>	Los argumentos usados para ejecutar la imagen de Docker <i>test-use</i> . El pod de prueba de Helm pasa estos argumentos al contenedor cuando se ejecuta <code>helm test</code> .	<code>"/./speech-to-text-client" "/./audio/whatstheweatherlike.wav" "--expect=What's the weather like" "--host=\$(SPEECH_TO_TEXT_HOST)" "--port=\$(SPEECH_TO_TEXT_PORT)"</code>
<code>textToSpeech.enabled</code>	Si el servicio <b>text-to-speech</b> está habilitado.	<code>true</code>
<code>textToSpeech.verification.enabled</code>	Si la funcionalidad <code>helm test</code> del servicio <b>speech-to-text</b> está habilitada.	<code>true</code>
<code>textToSpeech.verification.image.registry</code>	El repositorio de imágenes de Docker que <code>helm test</code> usa para probar el servicio <b>speech-to-text</b> . Helm crea un pod independiente dentro del clúster para realizar pruebas y extrae la imagen <i>test-use</i> de este Registro.	<code>docker.io</code>

PARÁMETRO	DESCRIPCIÓN	VALOR PREDETERMINADO
<code>textToSpeech.verification.image.repository</code>	El repositorio de imágenes de Docker que <code>helm test</code> usa para probar el servicio <b>speech-to-text</b> . El pod de prueba de Helm usa este repositorio para extraer la imagen <code>test-use</code> .	<code>antsu/on-prem-client</code>
<code>textToSpeech.verification.image.tag</code>	La etiqueta de la imagen de Docker usada con <code>helm test</code> para el servicio <b>speech-to-text</b> . El pod de prueba de Helm usa esta etiqueta para extraer la imagen <code>test-use</code> .	<code>latest</code>
<code>textToSpeech.verification.image.pullByHash</code>	Si la imagen de Docker <code>test-use</code> se extrae mediante hash. Si <code>true</code> , debería agregarse <code>textToSpeech.verification.image.hash</code> con un valor de Hash de imagen válido.	<code>false</code>
<code>textToSpeech.verification.image.arguments</code>	Los argumentos que se ejecutarán con la imagen de Docker <code>test-use</code> . El pod de prueba de Helm pasa estos argumentos al contenedor cuando se ejecuta <code>helm test</code> .	<code>"/text-to-speech-client" "--input='What's the weather like'" "--host=\$(TEXT_TO_SPEECH_HOST)" "--port=\$(TEXT_TO_SPEECH_PORT)"</code>

### Speech-to-Text (gráfico secundario: charts/speechToText)

Para reemplazar al gráfico de nivel superior, agregue el prefijo `speechToText.` a todos los parámetros para que sean más específicos. Se reemplazará el parámetro correspondiente, por ejemplo `speechToText.numberOfConcurrentRequest` reemplaza a `numberOfConcurrentRequest`.

PARÁMETRO	DESCRIPCIÓN	VALOR PREDETERMINADO
<code>enabled</code>	Si el servicio <b>speech-to-text</b> está habilitado.	<code>false</code>
<code>numberOfConcurrentRequest</code>	Número de solicitudes simultáneas del servicio <b>speech-to-text</b> . Este gráfico calcula automáticamente los recursos de CPU y memoria en función de este valor.	<code>2</code>
<code>optimizeForAudioFile</code>	Si el servicio necesita optimizar la entrada de audio a través de archivos de audio. Si <code>true</code> , este gráfico asignará más recursos de CPU al servicio.	<code>false</code>
<code>image.registry</code>	Registro de la imagen de Docker de <b>speech-to-text</b> .	<code>containerpreview.azurecr.io</code>
<code>image.repository</code>	Repositorio de la imagen de Docker de <b>speech-to-text</b> .	<code>microsoft/cognitive-services-speech-to-text</code>
<code>image.tag</code>	Etiqueta de la imagen de Docker de <b>speech-to-text</b> .	<code>latest</code>

PARÁMETRO	DESCRIPCIÓN	VALOR PREDETERMINADO
<code>image.pullSecrets</code>	Secretos de la imagen para extraer la imagen de Docker de <b>speech-to-text</b> .	
<code>image.pullByHash</code>	Si la imagen de Docker se extrae mediante hash. Si <code>true</code> , <code>image.hash</code> es obligatorio.	<code>false</code>
<code>image.hash</code>	Hash de la imagen de Docker de <b>speech-to-text</b> . Solo se usa con <code>image.pullByHash: true</code> .	
<code>image.args.eula</code> (obligatorio)	Indica que ha aceptado la licencia. El único valor válido es <code>accept</code> .	
<code>image.args.billing</code> (obligatorio)	El valor de URI del punto de conexión de facturación está disponible en Azure Portal, en la página de Información general de Información general del servicio de voz.	
<code>image.args.apikey</code> (obligatorio)	Se usa para realizar un seguimiento de la información de facturación.	
<code>service.type</code>	Tipo de servicio de Kubernetes del servicio <b>speech-to-text</b> . Consulte las <a href="#">instrucciones de los tipos de servicio de Kubernetes</a> para obtener más información y comprobar la compatibilidad con los proveedores de nube.	<code>LoadBalancer</code>
<code>service.port</code>	Puerto del servicio <b>speech-to-text</b> .	<code>80</code>
<code>service.annotations</code>	Anotaciones de <b>conversión de voz a texto</b> para los metadatos del servicio. Las anotaciones son pares de clave-valor. <code>annotations:</code> <code>some/annotation1: value1</code> <code>some/annotation2: value2</code>	
<code>service.autoScaler.enabled</code>	Si <a href="#">Horizontal Pod Autoscaler</a> está habilitado. Si <code>true</code> , <code>speech-to-text-autoscaler</code> se implementará en el clúster de Kubernetes.	<code>true</code>
<code>service.podDisruption.enabled</code>	Si <a href="#">Pod Disruption Budget</a> está habilitado. Si <code>true</code> , <code>speech-to-text-poddisruptionbudget</code> se implementará en el clúster de Kubernetes.	<code>true</code>

**Text-to-Speech (gráfico secundario: charts/textToSpeech)**

Para reemplazar al gráfico de nivel superior, agregue el prefijo `textToSpeech.` a todos los parámetros para que sean más específicos. Se reemplazará el parámetro correspondiente, por ejemplo

`textToSpeech.numberOfConcurrentRequest` reemplaza a `numberOfConcurrentRequest`.

PARÁMETRO	DESCRIPCIÓN	VALOR PREDETERMINADO
<code>enabled</code>	Si el servicio <b>text-to-speech</b> está habilitado.	<code>false</code>
<code>numberOfConcurrentRequest</code>	Número de solicitudes simultáneas del servicio <b>text-to-speech</b> . Este gráfico calcula automáticamente los recursos de CPU y memoria en función de este valor.	<code>2</code>
<code>optimizeForTurboMode</code>	Si el servicio necesita optimizar la entrada de texto a través de archivos de texto. Si <code>true</code> , este gráfico asignará más recursos de CPU al servicio.	<code>false</code>
<code>image.registry</code>	Registro de la imagen de Docker de <b>text-to-speech</b> .	<code>containerpreview.azurecr.io</code>
<code>image.repository</code>	Repositorio de la imagen de Docker de <b>text-to-speech</b> .	<code>microsoft/cognitive-services-text-to-speech</code>
<code>image.tag</code>	Etiqueta de la imagen de Docker de <b>text-to-speech</b> .	<code>latest</code>
<code>image.pullSecrets</code>	Secretos de la imagen para extraer la imagen de Docker de <b>text-to-speech</b> .	
<code>image.pullByHash</code>	Si la imagen de Docker se extrae mediante hash. Si <code>true</code> , <code>image.hash</code> es obligatorio.	<code>false</code>
<code>image.hash</code>	Hash de la imagen de Docker de <b>text-to-speech</b> . Solo se usa con <code>image.pullByHash: true</code> .	
<code>image.args.eula</code> (obligatorio)	Indica que ha aceptado la licencia. El único valor válido es <code>accept</code> .	
<code>image.args.billing</code> (obligatorio)	El valor de URI del punto de conexión de facturación está disponible en Azure Portal, en la página de Información general de Información general del servicio de voz.	
<code>image.args.apikey</code> (obligatorio)	Se usa para realizar un seguimiento de la información de facturación.	

PARÁMETRO	DESCRIPCIÓN	VALOR PREDETERMINADO
<code>service.type</code>	Tipo de servicio de Kubernetes del servicio <b>text-to-speech</b> . Consulte las <a href="#">instrucciones de los tipos de servicio de Kubernetes</a> para obtener más información y comprobar la compatibilidad con los proveedores de nube.	<code>LoadBalancer</code>
<code>service.port</code>	El puerto del servicio <b>text-to-speech</b> .	<code>80</code>
<code>service.annotations</code>	Las anotaciones de <b>text-to-speech</b> para los metadatos del servicio. Las anotaciones son pares clave-valor. <code>annotations:</code> <code>some/annotation1: value1</code> <code>some/annotation2: value2</code>	
<code>service.autoScaler.enabled</code>	Si <a href="#">Horizontal Pod Autoscaler</a> está habilitado. Si <code>true</code> , <code>text-to-speech-autoscaler</code> se implementará en el clúster de Kubernetes.	<code>true</code>
<code>service.podDisruption.enabled</code>	Si <a href="#">Pod Disruption Budget</a> está habilitado. Si <code>true</code> , <code>text-to-speech-poddisruptionbudget</code> se implementará en el clúster de Kubernetes.	<code>true</code>

## Pasos siguientes

Para obtener más información sobre la instalación de aplicaciones con Helm en Azure Kubernetes Service (AKS), [visite esta página](#).

[Contenedores de Cognitive Services](#)

# Implementación del contenedor del servicio de voz en Azure Container Instances

16/01/2020 • 9 minutes to read • [Edit Online](#)

Aprenda a implementar el contenedor del [servicio de voz](#) de Cognitive Services en una instancia de [Azure Container Instances](#). Este procedimiento muestra la creación de un recurso del servicio de voz de Azure. Luego se trata la extracción de la imagen de contenedor asociada. Por último, se resalta la posibilidad de aprovechar la orquestación de los dos desde un explorador. El uso de contenedores puede desviar la atención de los desarrolladores de la administración de la infraestructura y centrarla en el desarrollo de aplicaciones.

## Requisitos previos

- Use una suscripción de Azure. Si no tiene una suscripción a Azure, cree una [cuenta gratuita](#) antes de empezar.
- Instale la [CLI de Azure \(az\)](#).
- [Motor de docker](#) y se asegura de que la CLI de Docker funciona en una ventana de consola.

## Solicitud de acceso al registro de contenedor

Primero debe completar y enviar el [formulario de solicitud de contenedores del servicio Voz de Cognitive Services](#) para solicitar acceso al contenedor.

El formulario solicita información acerca del usuario y de su empresa, así como del escenario de usuario para el que se va a usar el contenedor. Después de haber enviado el formulario, el equipo de Azure Cognitive Services lo revisa para asegurarse de que cumple los criterios de acceso al registro de contenedor privado.

### IMPORTANT

Debe usar una dirección de correo electrónico que esté asociada con una cuenta de Microsoft (MSA) o de Azure Active Directory (Azure AD) en el formulario.

Si se aprueba la solicitud, recibirá un correo electrónico con instrucciones que describen cómo obtener las credenciales y tener acceso al registro de contenedor privado.

## Creación de un recurso de voz

1. Inicie sesión en [Azure Portal](#).
2. Haga clic en [Crear un recurso de voz](#).
3. Establezca todas las opciones de configuración necesarias:

CONFIGURACIÓN	VALOR
NOMBRE	Nombre que quiera (2-64 caracteres).
Subscription	Seleccione una suscripción adecuada.
Location	Seleccione cualquier ubicación disponible cercana.
Nivel de precios	<input type="text"/> F0 : el plan de tarifa mínimo.
Grupo de recursos	Seleccione un grupo de recursos disponible.

4. Haga clic en **Crear** y espere a que el recurso se cree. Una vez creado, vaya a la página de recursos.
5. Recopile el elemento `endpoint` configurado y una clave de API:

PESTAÑA DE RECURSOS EN EL PORTAL	CONFIGURACIÓN	VALOR
Información general	Punto de conexión	Copie el punto de conexión. Tiene un aspecto similar a <div style="border: 1px solid black; padding: 2px;"><a href="https://speech.cognitiveservices.azure.com/sts/v1">https://speech.cognitiveservices.azure.com/sts/v1</a></div>
Claves	Clave de API	Copie una de las dos claves. Es una cadena de 32 caracteres alfanuméricos sin espacios ni guiones, <div style="border: 1px solid black; padding: 2px;">xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx</div>

## Creación de un recurso de instancia de contenedor de Azure con la CLI de Azure

El siguiente código YAML define el recurso de instancia de contenedor de Azure. Copie y pegue el contenido en un nuevo archivo llamado `my-aci.yaml` y reemplace los valores comentados por los suyos propios. Consulte el [formato de plantilla](#) para ver el código YAML válido. Consulte las [imágenes y los repositorios de contenedor](#) para ver los nombres de imagen disponibles y su repositorio correspondiente. Para más información sobre la referencia de YAML para las instancias de contenedor, vea [referencia de YAML: Azure Container Instances](#).

```

apiVersion: 2018-10-01
location: # < Valid location >
name: # < Container Group name >
imageRegistryCredentials: # This is required when pulling a non-public image
  - server: containerpreview.azurecr.io
    username: # < The username for the preview container registry >
    password: # < The password for the preview container registry >
properties:
  containers:
    - name: # < Container name >
      properties:
        image: # < Repository/Image name >
        environmentVariables: # These env vars are required
          - name: eula
            value: accept
          - name: billing
            value: # < Service specific Endpoint URL >
          - name: apikey
            value: # < Service specific API key >
      resources:
        requests:
          cpu: 4 # Always refer to recommended minimal resources
          memoryInGb: 8 # Always refer to recommended minimal resources
      ports:
        - port: 5000
  osType: Linux
  volumes: # This node, is only required for container instances that pull their model in at runtime, such as LUIS.
  - name: aci-file-share
    azureFile:
      shareName: # < File share name >
      storageAccountName: # < Storage account name>
      storageAccountKey: # < Storage account key >
  restartPolicy: OnFailure
  ipAddress:
    type: Public
    ports:
      - protocol: tcp
        port: 5000
  tags: null
type: Microsoft.ContainerInstance/containerGroups

```

### NOTE

No todas las ubicaciones tienen la misma disponibilidad de CPU y memoria. Consulte la tabla [ubicación y recursos](#) para obtener una lista de los recursos disponibles para los contenedores por ubicación y sistema operativo.

Nos basaremos en el archivo YAML que hemos creado para el comando `az container create`. En la CLI de Azure, ejecute el comando `az container create` reemplazando `<resource-group>` por el suyo propio. Además, para proteger los valores dentro de una implementación de YAML, consulte cómo [proteger los valores](#).

```
az container create -g <resource-group> -f my-aci.yaml
```

La salida del comando es `Running...` si es válido. Después de unos minutos, la salida cambia a una cadena JSON que representa el recurso ACI recién creado. Es muy probable que la imagen de contenedor no esté disponible durante un tiempo, pero el recurso ya está implementado.

#### TIP

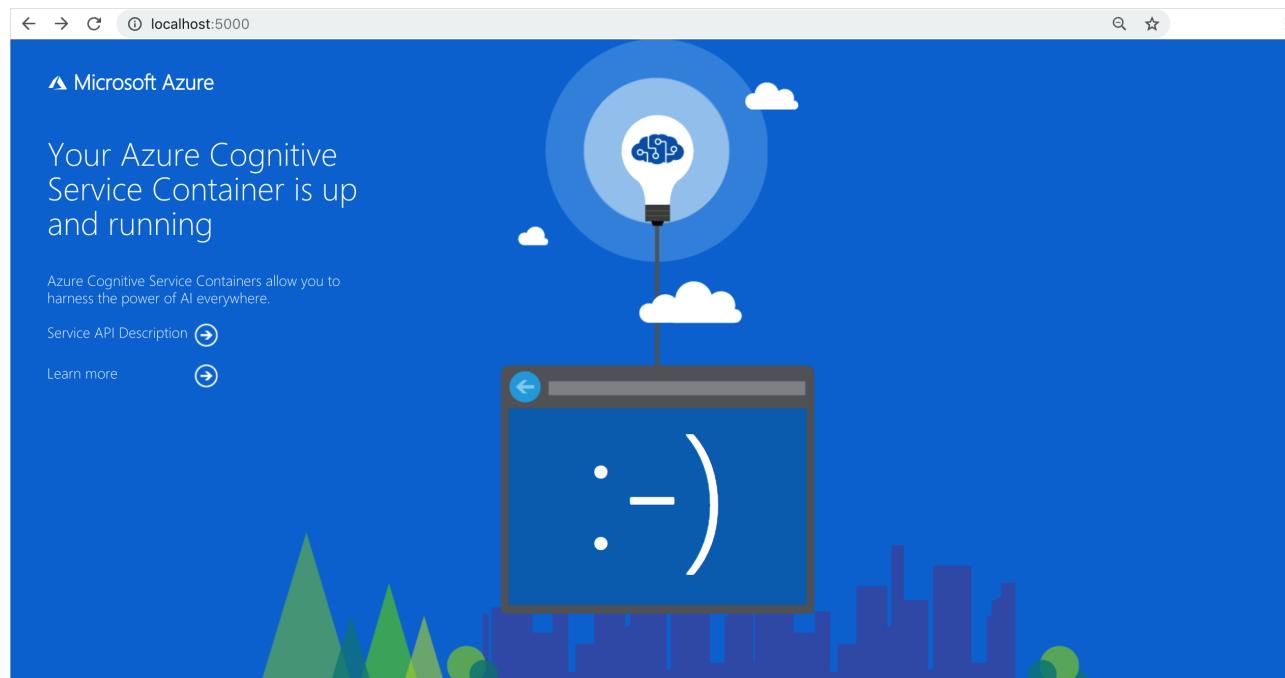
Preste mucha atención a las ubicaciones de las ofertas de la versión preliminar pública de Azure Cognitive Services, ya que el código YAML tendrá que ajustarse según la ubicación.

## Comprobación de que un contenedor está en ejecución

Hay varias maneras de comprobar que el contenedor está en ejecución. Busque la dirección *IP externa* y el puerto expuesto del contenedor en cuestión y abra el explorador web que prefiera. Use las distintas direcciones URL de solicitud para validar que el contenedor se está ejecutando. Las direcciones URL de solicitud de ejemplo que se enumeran a continuación son

`http://localhost:5000`, pero el contenedor específico puede variar. Tenga en cuenta que va a confiar en la *dirección IP externa* y el puerto expuesto del contenedor.

URL DE LA SOLICITUD	PROPOSITO
<code>http://localhost:5000/</code>	El contenedor ofrece una página principal.
<code>http://localhost:5000/status</code>	Se solicitó con HTTP GET, para comprobar que el contenedor está en ejecución sin causar una consulta al punto de conexión. Esta solicitud se puede usar con los <a href="#">sondeos de ejecución y preparación</a> de Kubernetes.
<code>http://localhost:5000/swagger</code>	El contenedor cuenta con un completo conjunto de documentación sobre los puntos de conexión y una característica de <b>prueba</b> . Esta característica le permite especificar la configuración en un formulario HTML basado en web y realizar la consulta sin necesidad de escribir código. Una vez que la consulta devuelve resultados, se proporciona un ejemplo del comando CURL para mostrar los encabezados HTTP y el formato de cuerpo requeridos.



## Pasos siguientes

Vamos a seguir trabajando con los contenedores de Azure Cognitive Services.

[Usar otros contenedores de Cognitive Services](#)

# Speech-to-text REST API

13/01/2020 • 19 minutes to read • [Edit Online](#)

Como alternativa al [SDK de Voz](#), el servicio de voz le permite convertir voz a texto mediante una API REST. Cada punto de conexión accesible se asocia con una región. La aplicación requiere una clave de suscripción para el punto de conexión que se va a usar.

Antes de usar speech-to-text API REST debe comprender que:

- Las solicitudes que usan la API REST y transmiten audio directamente solo pueden contener hasta 60 segundos de audio.
- La API REST de voz a texto solo devuelve resultados finales. No se proporcionan resultados parciales.

Si el envío de un audio más grande es necesario para la aplicación, considere la posibilidad de usar el [SDK de Voz](#) o una API REST basada en archivos, como la [transcripción por lotes](#).

## Authentication

Cada solicitud requiere un encabezado de autorización. Esta tabla muestra qué encabezados son compatibles con cada servicio:

ENCABEZADOS DE AUTORIZACIÓN COMPATIBLES	VOZ A TEXTO	TEXTO A VOZ
Ocp-Apim-Subscription-Key	Sí	Sin
Autorización: Portador	Sí	Sí

Cuando se usa el encabezado `Ocp-Apim-Subscription-Key`, solo se le pide que proporcione la clave de suscripción. Por ejemplo:

```
'Ocp-Apim-Subscription-Key': 'YOUR_SUBSCRIPTION_KEY'
```

Cuando se usa el encabezado `Authorization: Bearer`, se le pide que haga una solicitud al punto de conexión `issueToken`. En esta solicitud, va a intercambiar la clave de suscripción de un token de acceso que es válido durante 10 minutos. En las secciones siguientes obtendrá información sobre cómo obtener un token y usar un token.

### Obtención de un token de acceso

Para obtener un token de acceso, tiene que realizar una solicitud al punto de conexión `issueToken` mediante `Ocp-Apim-Subscription-Key` y su clave de suscripción.

Se admiten estas regiones y puntos de conexión:

REGION	PUNTO DE CONEXIÓN DE SERVICIO DE TOKEN
Este de Australia	<a href="https://australiaeast.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://australiaeast.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
Centro de Canadá	<a href="https://canadacentral.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://canadacentral.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
Centro de EE. UU.	<a href="https://centralus.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://centralus.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
Asia oriental	<a href="https://eastasia.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://eastasia.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
Este de EE. UU.	<a href="https://eastus.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://eastus.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
Este de EE. UU. 2	<a href="https://eastus2.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://eastus2.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
Centro de Francia	<a href="https://francecentral.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://francecentral.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
India central	<a href="https://centralindia.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://centralindia.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
Este de Japón	<a href="https://japaneast.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://japaneast.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
Corea Central	<a href="https://koreacentral.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://koreacentral.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
Centro-Norte de EE. UU	<a href="https://northcentralus.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://northcentralus.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
Europa del Norte	<a href="https://northeurope.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://northeurope.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
Centro-Sur de EE. UU	<a href="https://southcentralus.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://southcentralus.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>

REGION	PUNTO DE CONEXIÓN DE SERVICIO DE TOKEN
Sudeste asiático	<a href="https://southeastasia.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://southeastasia.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
Sur del Reino Unido 2	<a href="https://uksouth.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://uksouth.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
Europa occidental	<a href="https://westeurope.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://westeurope.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
Oeste de EE. UU.	<a href="https://westus.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://westus.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
Oeste de EE. UU. 2	<a href="https://westus2.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://westus2.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>

Use estos ejemplos para crear la solicitud de token de acceso.

#### Ejemplo de HTTP

Este ejemplo es una solicitud HTTP para obtener un token. Reemplace `YOUR_SUBSCRIPTION_KEY` por la clave de suscripción del servicio Voz. Si la suscripción no está en la región Oeste de EE. UU., reemplace el encabezado `Host` por el nombre de host de la región.

```
POST /sts/v1.0/issueToken HTTP/1.1
Ocp-Apim-Subscription-Key: YOUR_SUBSCRIPTION_KEY
Host: westus.api.cognitive.microsoft.com
Content-type: application/x-www-form-urlencoded
Content-Length: 0
```

El cuerpo de la respuesta contiene el token de acceso en formato JSON Web Token (JWT).

#### Ejemplo de PowerShell

En este ejemplo se muestra un script sencillo de PowerShell para obtener un token de acceso. Reemplace `YOUR_SUBSCRIPTION_KEY` por la clave de suscripción del servicio Voz. Asegúrese de usar el punto de conexión correcto para la región que coincida con su suscripción. En este ejemplo la región es Oeste de EE. UU.

```
$FetchTokenHeader = @{
    'Content-type'='application/x-www-form-urlencoded';
    'Content-Length'= '0';
    'Ocp-Apim-Subscription-Key' = 'YOUR_SUBSCRIPTION_KEY'
}

$OAuthToken = Invoke-RestMethod -Method POST -Uri https://westus.api.cognitive.microsoft.com/sts/v1.0/issueToken
-Headers $FetchTokenHeader

# show the token received
$OAuthToken
```

#### Ejemplo de cURL

cURL es una herramienta de la línea de comandos disponible en Linux (y en el subsistema Windows para Linux). Este comando cURL muestra cómo obtener un token de acceso. Reemplace `YOUR_SUBSCRIPTION_KEY` por la clave de suscripción del servicio Voz. Asegúrese de usar el punto de conexión correcto para la región que coincide con su suscripción. En este ejemplo la región es Oeste de EE. UU.

```
curl -v -X POST
"https://westus.api.cognitive.microsoft.com/sts/v1.0/issueToken" \
-H "Content-type: application/x-www-form-urlencoded" \
-H "Content-Length: 0" \
-H "Ocp-Apim-Subscription-Key: YOUR_SUBSCRIPTION_KEY"
```

#### Ejemplo de C#

Esta clase de C# muestra cómo obtener un token de acceso. Pase la clave de suscripción del servicio Voz al crear una instancia de la clase. Si su suscripción no está en la región Oeste de EE. UU., cambie el valor de `FetchTokenUri` para que coincida con la región de su suscripción.

```

public class Authentication
{
    public static readonly string FetchTokenUri =
        "https://westus.api.cognitive.microsoft.com/sts/v1.0/issueToken";
    private string subscriptionKey;
    private string token;

    public Authentication(string subscriptionKey)
    {
        this.subscriptionKey = subscriptionKey;
        this.token = FetchTokenAsync(FetchTokenUri, subscriptionKey).Result;
    }

    public string GetAccessToken()
    {
        return this.token;
    }

    private async Task<string> FetchTokenAsync(string fetchUri, string subscriptionKey)
    {
        using (var client = new HttpClient())
        {
            client.DefaultRequestHeaders.Add("Ocp-Apim-Subscription-Key", subscriptionKey);
            UriBuilder uriBuilder = new UriBuilder(fetchUri);

            var result = await client.PostAsync(uriBuilder.Uri.AbsoluteUri, null);
            Console.WriteLine("Token Uri: {0}", uriBuilder.Uri.AbsoluteUri);
            return await result.Content.ReadAsStringAsync();
        }
    }
}

```

#### Ejemplo de Python

```

# Request module must be installed.
# Run pip install requests if necessary.
import requests

subscription_key = 'REPLACE_WITH_YOUR_KEY'

def get_token(subscription_key):
    fetch_token_url = 'https://westus.api.cognitive.microsoft.com/sts/v1.0/issueToken'
    headers = {
        'Ocp-Apim-Subscription-Key': subscription_key
    }
    response = requests.post(fetch_token_url, headers=headers)
    access_token = str(response.text)
    print(access_token)

```

#### Uso de un token de acceso

Se debe enviar el token de acceso al servicio como encabezado `Authorization: Bearer <TOKEN>`. Cada token de acceso tiene una validez de 10 minutos. Puede obtener un nuevo token en cualquier momento. No obstante, para reducir el tráfico de red y la latencia, se recomienda usar el mismo token durante nueve minutos.

Este es un ejemplo de solicitud HTTP a la API REST de texto a voz:

```

POST /cognitiveservices/v1 HTTP/1.1
Authorization: Bearer YOUR_ACCESS_TOKEN
Host: westus.stt.speech.microsoft.com
Content-type: application/ssml+xml
Content-Length: 199
Connection: Keep-Alive

// Message body here...

```

## Regiones y puntos de conexión

Estas regiones son compatibles con la transcripción de voz a texto mediante la API REST. Asegúrese de que selecciona el punto de conexión que coincide con la región de su suscripción.

REGION	PUNTO DE CONEXIÓN
Este de Australia	<a href="https://australiaeast.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1">https://australiaeast.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1</a>
Centro de Canadá	<a href="https://canadacentral.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1">https://canadacentral.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1</a>
Centro de EE. UU.	<a href="https://centralus.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1">https://centralus.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1</a>

REGION	PUNTO DE CONEXIÓN
Asia oriental	<a href="https://eastasia.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1?language=en-US">https://eastasia.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1?language=en-US</a>
Este de EE. UU	<a href="https://eastus.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1?language=en-US">https://eastus.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1?language=en-US</a>
Este de EE. UU. 2	<a href="https://eastus2.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1?language=en-US">https://eastus2.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1?language=en-US</a>
Centro de Francia	<a href="https://francecentral.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1?language=en-US">https://francecentral.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1?language=en-US</a>
India central	<a href="https://centralindia.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1?language=en-US">https://centralindia.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1?language=en-US</a>
Este de Japón	<a href="https://japaneast.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1?language=en-US">https://japaneast.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1?language=en-US</a>
Corea Central	<a href="https://koreacentral.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1?language=en-US">https://koreacentral.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1?language=en-US</a>
Centro-Norte de EE. UU	<a href="https://northcentralus.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1?language=en-US">https://northcentralus.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1?language=en-US</a>
Europa del Norte	<a href="https://northeurope.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1?language=en-US">https://northeurope.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1?language=en-US</a>
Centro-Sur de EE. UU	<a href="https://southcentralus.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1?language=en-US">https://southcentralus.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1?language=en-US</a>
Sudeste asiático	<a href="https://southeastasia.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1?language=en-US">https://southeastasia.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1?language=en-US</a>
Sur del Reino Unido 2	<a href="https://uksouth.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1?language=en-US">https://uksouth.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1?language=en-US</a>
Europa occidental	<a href="https://westeurope.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1?language=en-US">https://westeurope.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1?language=en-US</a>
Oeste de EE. UU.	<a href="https://westus.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1?language=en-US">https://westus.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1?language=en-US</a>
Oeste de EE. UU. 2	<a href="https://westus2.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1?language=en-US">https://westus2.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1?language=en-US</a>

#### NOTE

El parámetro de idioma debe anexarse a la dirección URL para evitar la recepción de errores HTTP 4xx. Por ejemplo, el idioma definido a inglés de Estados Unidos con el punto de conexión del Oeste de EE. UU. es:

<https://westus.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1?language=en-US>.

## Parámetros de consulta

Estos parámetros podrían incluirse en la cadena de consulta de la solicitud de REST.

PARÁMETRO	DESCRIPCIÓN	OBLIGATORIO U OPCIONAL
<code>language</code>	Identifica el idioma hablado que se está reconociendo. Vea <a href="#">Idiomas admitidos</a> .	Obligatorio
<code>format</code>	Especifica el formato del resultado. Los valores aceptados son: <code>simple</code> y <code>detailed</code> . Los resultados simples incluyen <code>RecognitionStatus</code> , <code>DisplayText</code> , <code>Offset</code> y <code>Duration</code> . Las respuestas detalladas incluyen varios resultados con valores de confianza y cuatro representaciones diferentes. La configuración predeterminada es <code>simple</code> .	Opcional
<code>profanity</code>	Especifica cómo controlar las palabras soeces en los resultados del reconocimiento. Los valores aceptados son <code>masked</code> , que reemplaza las palabras soeces con asteriscos, <code>removed</code> , que quita todas las palabras soeces del resultado o <code>raw</code> que incluye la palabra soez en el resultado. La configuración predeterminada es <code>masked</code> .	Opcional

## Encabezados de solicitud

Esta tabla enumera los encabezados obligatorios y opcionales para las solicitudes de voz a texto.

ENCABEZADO	DESCRIPCIÓN	OBLIGATORIO U OPCIONAL
<code>Ocp-Apim-Subscription-Key</code>	Clave de suscripción del servicio de voz.	Se necesita este encabezado, o bien <code>Authorization</code> .
<code>Authorization</code>	Un token de autorización precedido por la palabra <code>Bearer</code> . Para más información, consulte <a href="#">Autenticación</a> .	Se necesita este encabezado, o bien <code>Ocp-Apim-Subscription-Key</code> .
<code>Content-type</code>	Describe el formato y el códec de los datos de audio proporcionados. Los valores aceptados son: <code>audio/wav; codecs=audio/pcm; samplerate=16000</code> y <code>audio/ogg; codecs=opus</code> .	Obligatorio
<code>Transfer-Encoding</code>	Especifica que se están enviando datos de audio fragmentados en lugar de un único archivo. Use este encabezado solo si hay fragmentación de los datos de audio.	Opcional
<code>Expect</code>	Si usa la transferencia fragmentada, envíe <code>Expect: 100-continue</code> . El servicio de voz confirma la solicitud inicial y espera datos adicionales.	Obligatorio si se envían datos de audio fragmentados.
<code>Accept</code>	Si se proporciona, debe ser <code>application/json</code> . El servicio de voz proporciona resultados en JSON. Algunos marcos de solicitud proporcionan un valor predeterminado no compatible. Se recomienda incluir siempre <code>Accept</code> .	Opcional pero recomendable.

## Formatos de audio

El audio se envía en el cuerpo de la solicitud HTTP `POST`. Debe estar en uno de los formatos de esta tabla:

FORMATO	CÓDEC	BITRATE	VELOCIDAD DE MUESTREO
WAV	PCM	16 bits	16 kHz, mono
OGG	OPUS	16 bits	16 kHz, mono

### NOTE

Se admiten los formatos anteriores a través de la API REST y WebSocket en el servicio de voz. El [SDK de Voz](#) actualmente solo admite el formato WAV con el códec PCM.

## Solicitud de ejemplo

El ejemplo siguiente incluye el nombre de host y los encabezados necesarios. Es importante tener en cuenta que el servicio también espera datos de audio, que no están incluidos en este ejemplo. Como se ha mencionado anteriormente, la fragmentación es recomendable pero no obligatoria.

```
POST speech/recognition/conversation/cognitiveservices/v1?language=en-US&format=detailed HTTP/1.1
Accept: application/json;text/xml
Content-Type: audio/wav; codecs=audio/pcm; samplerate=16000
Ocp-Apim-Subscription-Key: YOUR_SUBSCRIPTION_KEY
Host: westus.stt.speech.microsoft.com
Transfer-Encoding: chunked
Expect: 100-continue
```

## Códigos de estado HTTP

El estado HTTP de cada respuesta indica estados de corrección o error comunes.

CÓDIGO DE ESTADO HTTP	DESCRIPCIÓN	POSIBLE MOTIVO
100	Continuar	Se ha aceptado la solicitud inicial. Continúe con el envío del resto de los datos. (Se usa con la transferencia fragmentada).
200	OK	La solicitud es correcta; el cuerpo de la respuesta es un objeto JSON.
400	Solicitud incorrecta	Código de idioma no proporcionado, idioma no compatible; archivo de audio no válido, etc.

CÓDIGO DE ESTADO HTTP	DESCRIPCIÓN	POSSIBLE MOTIVO
401	No autorizado	Clave de suscripción o token de autorización no válido en la región especificada, o punto de conexión no válido.
403	Prohibido	Falta la clave de suscripción o el token de autorización.

## Transferencia fragmentada

La transferencia fragmentada (`Transfer-Encoding: chunked`) puede ayudar a reducir la latencia de reconocimiento. Permite al servicio de voz empezar a procesar el archivo de audio mientras se transmite. La API de REST no proporciona resultados parciales ni provisionales.

Este ejemplo de código muestra cómo enviar audio en fragmentos. Solo el primer fragmento debe contener el encabezado del archivo de audio. `request` es un objeto `HttpWebRequest` conectado al punto de conexión de REST adecuado. `audioFile` es la ruta de acceso a un archivo de audio en disco.

```
HttpWebRequest request = null;
request = (HttpWebRequest)HttpWebRequest.Create(requestUri);
request.SendChunked = true;
request.Accept = @"application/json;text/xml";
request.Method = "POST";
request.ProtocolVersion = HttpVersion.Version11;
request.Host = host;
request.ContentType = @"audio/wav; codecs=audio/pcm; samplerate=16000";
request.Headers["Ocp-Apim-Subscription-Key"] = args[1];
request.AllowWriteStreamBuffering = false;

using (fs = new FileStream(audioFile, FileMode.Open, FileAccess.Read))
{
    /*
     * Open a request stream and write 1024 byte chunks in the stream one at a time.
     */
    byte[] buffer = null;
    int bytesRead = 0;
    using (Stream requestStream = request.GetRequestStream())
    {
        /*
         * Read 1024 raw bytes from the input audio file.
         */
        buffer = new Byte[checked((uint)Math.Min(1024, (int)fs.Length))];
        while ((bytesRead = fs.Read(buffer, 0, buffer.Length)) != 0)
        {
            requestStream.Write(buffer, 0, bytesRead);
        }
        // Flush
        requestStream.Flush();
    }
}
```

## Parámetros de respuesta

Los resultados se proporcionan como JSON. El formato `simple` incluye los siguientes campos de nivel superior.

PARÁMETRO	DESCRIPCIÓN
<code>RecognitionStatus</code>	Estado, como <code>Success</code> , para un reconocimiento correcto. Vea la tabla siguiente.
<code>DisplayText</code>	Texto reconocido tras mayúsculas, puntuación, normalización inversa de texto (conversión de texto hablado en formularios más cortos, como 200 para "doscientos" o "Dr. Smith" para "doctor smith") y enmascaramiento de palabras soeces. Solo se presenta en caso de corrección.
<code>Offset</code>	El tiempo (en unidades de 100 nanosegundos) en el que comienza la voz reconocida en la secuencia de audio.
<code>Duration</code>	La duración (en unidades de 100 nanosegundos) de la voz reconocida en la secuencia de audio.

El campo `RecognitionStatus` puede contener estos valores:

STATUS	DESCRIPCIÓN
<code>Success</code>	El reconocimiento es correcto y el campo <code>DisplayText</code> está presente.

STATUS	DESCRIPCIÓN
NoMatch	Se detectó voz en la secuencia de audio, pero no se encontraron coincidencias de palabras en el idioma de destino. Normalmente significa que el idioma de reconocimiento es un idioma distinto al que habla el usuario.
InitialSilenceTimeout	El inicio de la secuencia de audio contiene solo silencio y el servicio ha agotado el tiempo de espera de la voz.
BabbleTimeout	El inicio de la secuencia de audio contiene solo ruido y el servicio ha agotado el tiempo de espera de la voz.
Error	El servicio de reconocimiento ha detectado un error interno y no ha podido continuar. Vuelva a intentarlo si es posible.

#### NOTE

Si el audio consta solo de palabras soeces y el parámetro de consulta `profanity` está establecido en `remove`, el servicio no devuelve ningún resultado de voz.

El formato `detailed` incluye los mismos datos que el formato `simple`, junto con `NBest`, una lista de interpretaciones alternativas del mismo resultado de reconocimiento. Estos resultados se clasifican de más probable a menos probable. La primera entrada es la misma que el resultado de reconocimiento principal. Cuando se usa el formato `detailed`, se proporciona `DisplayText` como `Display` para cada resultado de la lista `NBest`.

Cada objeto de la lista `NBest` incluye:

PARÁMETRO	DESCRIPCIÓN
<code>Confidence</code>	La puntuación de confianza de la entrada de 0,0 (ninguna confianza) a 1,0 (plena confianza)
<code>Lexical</code>	La forma léxica del texto reconocido: palabras reales reconocidas.
<code>ITN</code>	El formato de normalización inversa de texto ("canónica") del texto reconocido, con números de teléfono, números, abreviaturas ("doctor smith" a "dr smith") y otras transformaciones aplicadas.
<code>MaskedITN</code>	Formato ITN con enmascaramiento de palabras soeces aplicado, si se solicita.
<code>Display</code>	Formato de presentación del texto reconocido, con adición de signos de puntuación y mayúsculas. Este parámetro es el mismo que <code>DisplayText</code> que se proporcionó cuando el formato se estableció en <code>simple</code> .

## Respuestas de ejemplo

La siguiente es una respuesta típica de reconocimiento de `simple`:

```
{
  "RecognitionStatus": "Success",
  "DisplayText": "Remind me to buy 5 pencils.",
  "Offset": "1236645672289",
  "Duration": "1236645672289"
}
```

La siguiente es una respuesta típica de reconocimiento de `detailed`:

```
{  
    "RecognitionStatus": "Success",  
    "Offset": "1236645672289",  
    "Duration": "1236645672289",  
    "NBest": [  
        {  
            "Confidence" : "0.87",  
            "Lexical" : "remind me to buy five pencils",  
            "ITN" : "remind me to buy 5 pencils",  
            "MaskedITN" : "remind me to buy 5 pencils",  
            "Display" : "Remind me to buy 5 pencils.",  
        },  
        {  
            "Confidence" : "0.54",  
            "Lexical" : "rewind me to buy five pencils",  
            "ITN" : "rewind me to buy 5 pencils",  
            "MaskedITN" : "rewind me to buy 5 pencils",  
            "Display" : "Rewind me to buy 5 pencils.",  
        }  
    ]  
}
```

## Pasos siguientes

- [Obtenga su suscripción de prueba a Voz](#)
- [Personalización de modelos acústicos](#)
- [Personalización de modelos de lenguaje](#)

# Text-to-speech REST API

13/01/2020 • 18 minutes to read • [Edit Online](#)

El servicio de voz le permite [convertir texto en voz sintetizada](#) y [obtener una lista de voces admitidas](#) para una región con un conjunto de API REST. Cada punto de conexión disponible se asocia con una región. Se requiere una clave de suscripción para el punto de conexión o región que se va a usar.

Text to Speech REST API admite voces neuronales y de texto a voz estándar, y cada una de ellas admite un idioma y un dialecto específicos, que se identifican mediante la configuración regional.

- Para ver una lista completa de voces, consulte [compatibilidad con idiomas](#).
- Para obtener información acerca de la disponibilidad regional, consulte las [regiones](#).

## IMPORTANT

Los costes varían para voces estándar, personalizadas y neuronales. Para obtener más información, consulte el apartado [Precios](#).

Antes de utilizar esta API, comprenda:

- La API REST de texto a voz requiere un encabezado de autorización. Esto significa que tiene que completar un intercambio de tokens para acceder al servicio. Para más información, consulte [Autenticación](#).

## Authentication

Cada solicitud requiere un encabezado de autorización. Esta tabla muestra qué encabezados son compatibles con cada servicio:

ENCABEZADOS DE AUTORIZACIÓN COMPATIBLES	VOZ A TEXTO	TEXTO A VOZ
Ocp-Apim-Subscription-Key	Sí	Sí
Autorización: Portador	Sí	Sí

Cuando se usa el encabezado `Ocp-Apim-Subscription-Key`, solo se le pide que proporcione la clave de suscripción. Por ejemplo:

```
'Ocp-Apim-Subscription-Key': 'YOUR_SUBSCRIPTION_KEY'
```

Cuando se usa el encabezado `Authorization: Bearer`, se le pide que haga una solicitud al punto de conexión `issueToken`. En esta solicitud, va a intercambiar la clave de suscripción de un token de acceso que es válido durante 10 minutos. En las secciones siguientes obtendrá información sobre cómo obtener un token y usar un token.

### Obtención de un token de acceso

Para obtener un token de acceso, tiene que realizar una solicitud al punto de conexión `issueToken` mediante `Ocp-Apim-Subscription-Key` y su clave de suscripción.

Se admiten estas regiones y puntos de conexión:

REGION	PUNTO DE CONEXIÓN DE SERVICIO DE TOKEN
Este de Australia	<code>https://australiaeast.api.cognitive.microsoft.com/sts/v1.0/issueToken</code>
Centro de Canadá	<code>https://canadacentral.api.cognitive.microsoft.com/sts/v1.0/issueToken</code>
Centro de EE. UU.	<code>https://centralus.api.cognitive.microsoft.com/sts/v1.0/issueToken</code>
Asia oriental	<code>https://eastasia.api.cognitive.microsoft.com/sts/v1.0/issueToken</code>
Este de EE. UU	<code>https://eastus.api.cognitive.microsoft.com/sts/v1.0/issueToken</code>

REGION	PUNTO DE CONEXIÓN DE SERVICIO DE TOKEN
Este de EE. UU. 2	<a href="https://eastus2.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://eastus2.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
Centro de Francia	<a href="https://francecentral.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://francecentral.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
India central	<a href="https://centralindia.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://centralindia.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
Este de Japón	<a href="https://japaneast.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://japaneast.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
Corea Central	<a href="https://koreacentral.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://koreacentral.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
Centro-Norte de EE. UU	<a href="https://northcentralus.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://northcentralus.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
Europa del Norte	<a href="https://northeurope.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://northeurope.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
Centro-Sur de EE. UU	<a href="https://southcentralus.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://southcentralus.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
Sudeste asiático	<a href="https://southeastasia.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://southeastasia.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
Sur del Reino Unido 2	<a href="https://uksouth.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://uksouth.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
Europa occidental	<a href="https://westeurope.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://westeurope.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
Oeste de EE. UU.	<a href="https://westus.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://westus.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>
Oeste de EE. UU. 2	<a href="https://westus2.api.cognitive.microsoft.com/sts/v1.0/issueToken">https://westus2.api.cognitive.microsoft.com/sts/v1.0/issueToken</a>

Use estos ejemplos para crear la solicitud de token de acceso.

#### Ejemplo de HTTP

Este ejemplo es una solicitud HTTP para obtener un token. Reemplace `YOUR_SUBSCRIPTION_KEY` por la clave de suscripción del servicio Voz. Si la suscripción no está en la región Oeste de EE. UU., reemplace el encabezado `Host` por el nombre de host de la región.

```
POST /sts/v1.0/issueToken HTTP/1.1
Ocp-Apim-Subscription-Key: YOUR_SUBSCRIPTION_KEY
Host: westus.api.cognitive.microsoft.com
Content-type: application/x-www-form-urlencoded
Content-Length: 0
```

El cuerpo de la respuesta contiene el token de acceso en formato JSON Web Token (JWT).

#### Ejemplo de PowerShell

En este ejemplo se muestra un script sencillo de PowerShell para obtener un token de acceso. Reemplace `YOUR_SUBSCRIPTION_KEY` por la clave de suscripción del servicio Voz. Asegúrese de usar el punto de conexión correcto para la región que coincide con su suscripción. En este ejemplo la región es Oeste de EE. UU.

```
$FetchTokenHeader = @{
    'Content-type'='application/x-www-form-urlencoded';
    'Content-Length'='0';
    'Ocp-Apim-Subscription-Key' = 'YOUR_SUBSCRIPTION_KEY'
}

$OAuthToken = Invoke-RestMethod -Method POST -Uri https://westus.api.cognitive.microsoft.com/sts/v1.0/issueToken
-Headers $FetchTokenHeader

# show the token received
$OAuthToken
```

#### Ejemplo de cURL

cURL es una herramienta de la línea de comandos disponible en Linux (y en el subsistema Windows para Linux). Este comando

cURL muestra cómo obtener un token de acceso. Reemplace `YOUR_SUBSCRIPTION_KEY` por la clave de suscripción del servicio Voz. Asegúrese de usar el punto de conexión correcto para la región que coincide con su suscripción. En este ejemplo la región es Oeste de EE. UU.

```
curl -v -X POST
"https://westus.api.cognitive.microsoft.com/sts/v1.0/issueToken" \
-H "Content-type: application/x-www-form-urlencoded" \
-H "Content-Length: 0" \
-H "Ocp-Apim-Subscription-Key: YOUR_SUBSCRIPTION_KEY"
```

#### Ejemplo de C#

Esta clase de C# muestra cómo obtener un token de acceso. Pase la clave de suscripción del servicio Voz al crear una instancia de la clase. Si su suscripción no está en la región Oeste de EE. UU., cambie el valor de `FetchTokenUri` para que coincida con la región de su suscripción.

```
public class Authentication
{
    public static readonly string FetchTokenUri =
        "https://westus.api.cognitive.microsoft.com/sts/v1.0/issueToken";
    private string subscriptionKey;
    private string token;

    public Authentication(string subscriptionKey)
    {
        this.subscriptionKey = subscriptionKey;
        this.token = FetchTokenAsync(FetchTokenUri, subscriptionKey).Result;
    }

    public string GetAccessToken()
    {
        return this.token;
    }

    private async Task<string> FetchTokenAsync(string fetchUri, string subscriptionKey)
    {
        using (var client = new HttpClient())
        {
            client.DefaultRequestHeaders.Add("Ocp-Apim-Subscription-Key", subscriptionKey);
            UriBuilder uriBuilder = new UriBuilder(fetchUri);

            var result = await client.PostAsync(uriBuilder.Uri.AbsoluteUri, null);
            Console.WriteLine("Token Uri: {0}", uriBuilder.Uri.AbsoluteUri);
            return await result.Content.ReadAsStringAsync();
        }
    }
}
```

#### Ejemplo de Python

```
# Request module must be installed.
# Run pip install requests if necessary.
import requests

subscription_key = 'REPLACE_WITH_YOUR_KEY'

def get_token(subscription_key):
    fetch_token_url = 'https://westus.api.cognitive.microsoft.com/sts/v1.0/issueToken'
    headers = {
        'Ocp-Apim-Subscription-Key': subscription_key
    }
    response = requests.post(fetch_token_url, headers=headers)
    access_token = str(response.text)
    print(access_token)
```

#### Uso de un token de acceso

Se debe enviar el token de acceso al servicio como encabezado `Authorization: Bearer <TOKEN>`. Cada token de acceso tiene una validez de 10 minutos. Puede obtener un nuevo token en cualquier momento. No obstante, para reducir el tráfico de red y la latencia, se recomienda usar el mismo token durante nueve minutos.

Este es un ejemplo de solicitud HTTP a la API REST de texto a voz:

```

POST /cognitiveservices/v1 HTTP/1.1
Authorization: Bearer YOUR_ACCESS_TOKEN
Host: westus.stt.speech.microsoft.com
Content-type: application/ssml+xml
Content-Length: 199
Connection: Keep-Alive

// Message body here...

```

## Obtener una lista de voces

El punto de conexión `voices/list` le permite obtener una lista completa de las voces de una región o punto de conexión en concreto.

### Regiones y puntos de conexión

REGION	PUNTO DE CONEXIÓN
Este de Australia	<a href="https://australiaeast.tts.speech.microsoft.com/cognitiveservices/voices/list">https://australiaeast.tts.speech.microsoft.com/cognitiveservices/voices/list</a>
Sur de Brasil	<a href="https://brazilsouth.tts.speech.microsoft.com/cognitiveservices/voices/list">https://brazilsouth.tts.speech.microsoft.com/cognitiveservices/voices/list</a>
Centro de Canadá	<a href="https://canadacentral.tts.speech.microsoft.com/cognitiveservices/voices/list">https://canadacentral.tts.speech.microsoft.com/cognitiveservices/voices/list</a>
Centro de EE. UU.	<a href="https://centralus.tts.speech.microsoft.com/cognitiveservices/voices/list">https://centralus.tts.speech.microsoft.com/cognitiveservices/voices/list</a>
Asia oriental	<a href="https://eastasia.tts.speech.microsoft.com/cognitiveservices/voices/list">https://eastasia.tts.speech.microsoft.com/cognitiveservices/voices/list</a>
East US	<a href="https://eastus.tts.speech.microsoft.com/cognitiveservices/voices/list">https://eastus.tts.speech.microsoft.com/cognitiveservices/voices/list</a>
Este de EE. UU. 2	<a href="https://eastus2.tts.speech.microsoft.com/cognitiveservices/voices/list">https://eastus2.tts.speech.microsoft.com/cognitiveservices/voices/list</a>
Centro de Francia	<a href="https://francecentral.tts.speech.microsoft.com/cognitiveservices/voices/list">https://francecentral.tts.speech.microsoft.com/cognitiveservices/voices/list</a>
India central	<a href="https://centralindia.tts.speech.microsoft.com/cognitiveservices/voices/list">https://centralindia.tts.speech.microsoft.com/cognitiveservices/voices/list</a>
Este de Japón	<a href="https://japaneast.tts.speech.microsoft.com/cognitiveservices/voices/list">https://japaneast.tts.speech.microsoft.com/cognitiveservices/voices/list</a>
Corea Central	<a href="https://koreacentral.tts.speech.microsoft.com/cognitiveservices/voices/list">https://koreacentral.tts.speech.microsoft.com/cognitiveservices/voices/list</a>
Centro-Norte de EE. UU	<a href="https://northcentralus.tts.speech.microsoft.com/cognitiveservices/voices/list">https://northcentralus.tts.speech.microsoft.com/cognitiveservices/voices/list</a>
Europa del Norte	<a href="https://northeurope.tts.speech.microsoft.com/cognitiveservices/voices/list">https://northeurope.tts.speech.microsoft.com/cognitiveservices/voices/list</a>
Centro-Sur de EE. UU	<a href="https://southcentralus.tts.speech.microsoft.com/cognitiveservices/voices/list">https://southcentralus.tts.speech.microsoft.com/cognitiveservices/voices/list</a>
Sudeste asiático	<a href="https://southeastasia.tts.speech.microsoft.com/cognitiveservices/voices/list">https://southeastasia.tts.speech.microsoft.com/cognitiveservices/voices/list</a>
Sur del Reino Unido 2	<a href="https://uksouth.tts.speech.microsoft.com/cognitiveservices/voices/list">https://uksouth.tts.speech.microsoft.com/cognitiveservices/voices/list</a>
Europa occidental	<a href="https://westeurope.tts.speech.microsoft.com/cognitiveservices/voices/list">https://westeurope.tts.speech.microsoft.com/cognitiveservices/voices/list</a>
Oeste de EE. UU.	<a href="https://westus.tts.speech.microsoft.com/cognitiveservices/voices/list">https://westus.tts.speech.microsoft.com/cognitiveservices/voices/list</a>
Oeste de EE. UU. 2	<a href="https://westus2.tts.speech.microsoft.com/cognitiveservices/voices/list">https://westus2.tts.speech.microsoft.com/cognitiveservices/voices/list</a>

### Encabezados de solicitud

En esta tabla se enumeran los encabezados obligatorios y opcionales para las solicitudes de texto a voz.

ENCABEZADO	DESCRIPCIÓN	OBLIGATORIO U OPCIONAL
<code>Authorization</code>	Un token de autorización precedido por la palabra <code>Bearer</code> . Para más información, consulte <a href="#">Autenticación</a> .	Obligatorio

## Cuerpo de la solicitud

No es necesario un cuerpo para las solicitudes `GET` a este punto de conexión.

## Solicitud de ejemplo

Esta solicitud solo requiere un encabezado de autorización.

```
GET /cognitiveservices/voices/list HTTP/1.1
Host: westus.tts.speech.microsoft.com
Authorization: Bearer [Base64 access_token]
```

## Respuesta de muestra

Esta respuesta se ha truncado para ilustrar la estructura de una respuesta.

### NOTE

La disponibilidad de voces varía según la región o el punto de conexión.

```
[
  {
    "Name": "Microsoft Server Speech Text to Speech Voice (ar-EG, Hoda)",
    "ShortName": "ar-EG-Hoda",
    "Gender": "Female",
    "Locale": "ar-EG"
  },
  {
    "Name": "Microsoft Server Speech Text to Speech Voice (ar-SA, Naayf)",
    "ShortName": "ar-SA-Naayf",
    "Gender": "Male",
    "Locale": "ar-SA"
  },
  {
    "Name": "Microsoft Server Speech Text to Speech Voice (bg-BG, Ivan)",
    "ShortName": "bg-BG-Ivan",
    "Gender": "Male",
    "Locale": "bg-BG"
  },
  {
    "Name": "Microsoft Server Speech Text to Speech Voice (ca-ES, HerenaRUS)",
    "ShortName": "ca-ES-HerenaRUS",
    "Gender": "Female",
    "Locale": "ca-ES"
  },
  {
    "Name": "Microsoft Server Speech Text to Speech Voice (cs-CZ, Jakub)",
    "ShortName": "cs-CZ-Jakub",
    "Gender": "Male",
    "Locale": "cs-CZ"
  },
  ...
]
```

## Códigos de estado HTTP

El estado HTTP de cada respuesta indica estados de corrección o error comunes.

CÓDIGO DE ESTADO HTTP	DESCRIPCIÓN	POSIBLE MOTIVO
200	OK	La solicitud fue correcta.

CÓDIGO DE ESTADO HTTP	DESCRIPCIÓN	POSSIBLE MOTIVO
400	Bad Request	Falta un parámetro requerido, está vacío o es nulo. O bien, el valor pasado a un parámetro obligatorio u opcional no es válido. Un problema común es que el encabezado sea demasiado largo.
401	No autorizado	La solicitud no está autenticada. Asegúrese de que la clave de suscripción o el token sean válidos y de la región correcta.
429	Demasiadas solicitudes	Ha superado la cuota o la tasa de solicitudes permitidas para su suscripción.
502	Puerta de enlace incorrecta	Problema de red o de servidor. Podría indicar también encabezados no válidos.

## Conversión de texto a voz

El punto de conexión `v1` le permite convertir texto a voz usando [lenguaje de marcado de síntesis de voz \(SSML\)](#).

### Regiones y puntos de conexión

Estas regiones son compatibles con la conversión de texto a voz mediante la API REST. Asegúrese de que selecciona el punto de conexión que coincida con la región de su suscripción.

### Voces estándares y neuronales

Utilice esta tabla para determinar la disponibilidad de las voces estándar y neuronales por región o punto de conexión:

REGION	PUNTO DE CONEXIÓN	VOCES ESTÁNDAR	VOCES NEURONALES
Este de Australia	<a href="https://australiaeast.tts.speech.microsoft.com/cognitiveservices/v1">https://australiaeast.tts.speech.microsoft.com/cognitiveservices/v1</a>	Sí	
Centro de Canadá	<a href="https://canadacentral.tts.speech.microsoft.com/cognitiveservices/v1">https://canadacentral.tts.speech.microsoft.com/cognitiveservices/v1</a>	Sí	
Centro de EE. UU.	<a href="https://centralus.tts.speech.microsoft.com/cognitiveservices/v1">https://centralus.tts.speech.microsoft.com/cognitiveservices/v1</a>	Sin	
Asia oriental	<a href="https://eastasia.tts.speech.microsoft.com/cognitiveservices/v1">https://eastasia.tts.speech.microsoft.com/cognitiveservices/v1</a>	Sin	
Este de EE. UU	<a href="https://eastus.tts.speech.microsoft.com/cognitiveservices/v1">https://eastus.tts.speech.microsoft.com/cognitiveservices/v1</a>	Sí	
Este de EE. UU. 2	<a href="https://eastus2.tts.speech.microsoft.com/cognitiveservices/v1">https://eastus2.tts.speech.microsoft.com/cognitiveservices/v1</a>	Sin	
Centro de Francia	<a href="https://francecentral.tts.speech.microsoft.com/cognitiveservices/v1">https://francecentral.tts.speech.microsoft.com/cognitiveservices/v1</a>	Sí	
India central	<a href="https://centralindia.tts.speech.microsoft.com/cognitiveservices/v1">https://centralindia.tts.speech.microsoft.com/cognitiveservices/v1</a>	Sí	
Este de Japón	<a href="https://japaneast.tts.speech.microsoft.com/cognitiveservices/v1">https://japaneast.tts.speech.microsoft.com/cognitiveservices/v1</a>	Sin	
Corea Central	<a href="https://koreacentral.tts.speech.microsoft.com/cognitiveservices/v1">https://koreacentral.tts.speech.microsoft.com/cognitiveservices/v1</a>	Sí	
Centro-Norte de EE. UU	<a href="https://northcentralus.tts.speech.microsoft.com/cognitiveservices/v1">https://northcentralus.tts.speech.microsoft.com/cognitiveservices/v1</a>	Sí	
Europa del Norte	<a href="https://northeurope.tts.speech.microsoft.com/cognitiveservices/v1">https://northeurope.tts.speech.microsoft.com/cognitiveservices/v1</a>	Sin	
Centro-Sur de EE. UU	<a href="https://southcentralus.tts.speech.microsoft.com/cognitiveservices/v1">https://southcentralus.tts.speech.microsoft.com/cognitiveservices/v1</a>	Sí	
Sudeste asiático	<a href="https://southeastasia.tts.speech.microsoft.com/cognitiveservices/v1">https://southeastasia.tts.speech.microsoft.com/cognitiveservices/v1</a>	Sí	
Sur de Reino Unido 2	<a href="https://uksouth.tts.speech.microsoft.com/cognitiveservices/v1">https://uksouth.tts.speech.microsoft.com/cognitiveservices/v1</a>	Sí	

REGION	PUNTO DE CONEXIÓN	VOCES ESTÁNDAR	VOCES NEURONALES
Europa occidental	<a href="https://westeurope.tts.speech.microsoft.com/cognitiveservices/v1">https://westeurope.tts.speech.microsoft.com/cognitiveservices/v1</a>	Sí	
Oeste de EE. UU.	<a href="https://westus.tts.speech.microsoft.com/cognitiveservices/v1">https://westus.tts.speech.microsoft.com/cognitiveservices/v1</a>		Sin
Oeste de EE. UU. 2	<a href="https://westus2.tts.speech.microsoft.com/cognitiveservices/v1">https://westus2.tts.speech.microsoft.com/cognitiveservices/v1</a>		Sí

### Voces personalizadas

Si ha creado una fuente de voz personalizada, use el punto de conexión que ha creado. También puede utilizar los puntos de conexión que se indican a continuación; para ello, reemplace `{deploymentId}` por el identificador de implementación para el modelo de voz.

REGION	PUNTO DE CONEXIÓN
Este de Australia	<a href="https://australiaeast.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}">https://australiaeast.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}</a>
Centro de Canadá	<a href="https://canadacentral.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}">https://canadacentral.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}</a>
Centro de EE. UU.	<a href="https://centralus.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}">https://centralus.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}</a>
Asia oriental	<a href="https://eastasia.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}">https://eastasia.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}</a>
Este de EE. UU.	<a href="https://eastus.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}">https://eastus.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}</a>
Este de EE. UU. 2	<a href="https://eastus2.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}">https://eastus2.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}</a>
Centro de Francia	<a href="https://francecentral.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}">https://francecentral.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}</a>
India central	<a href="https://centralindia.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}">https://centralindia.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}</a>
Este de Japón	<a href="https://japaneast.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}">https://japaneast.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}</a>
Corea Central	<a href="https://koreacentral.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}">https://koreacentral.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}</a>
Centro-Norte de EE. UU.	<a href="https://northcentralus.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}">https://northcentralus.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}</a>
Europa del Norte	<a href="https://northeurope.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}">https://northeurope.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}</a>
Centro-Sur de EE. UU.	<a href="https://southcentralus.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}">https://southcentralus.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}</a>
Sudeste asiático	<a href="https://southeastasia.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}">https://southeastasia.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}</a>
Sur del Reino Unido 2	<a href="https://uksouth.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}">https://uksouth.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}</a>
Europa occidental	<a href="https://westeurope.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}">https://westeurope.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}</a>
Oeste de EE. UU.	<a href="https://westus.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}">https://westus.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}</a>

REGION	PUNTO DE CONEXIÓN
Oeste de EE. UU. 2	<code>https://westus2.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}</code>

## Encabezados de solicitud

En esta tabla se enumeran los encabezados obligatorios y opcionales para las solicitudes de texto a voz.

ENCABEZADO	DESCRIPCIÓN	OBLIGATORIO U OPCIONAL
<code>Authorization</code>	Un token de autorización precedido por la palabra <code>Bearer</code> . Para más información, consulte <a href="#">Autenticación</a> .	Obligatorio
<code>Content-Type</code>	Especifica el tipo de contenido para el texto proporcionado. Valor aceptable: <code>application/ssml+xml</code> .	Obligatorio
<code>X-Microsoft-OutputFormat</code>	Especifica el formato de salida del audio. Para obtener una lista completa de los valores aceptados, consulte <a href="#">salidas de audio</a> .	Obligatorio
<code>User-Agent</code>	Nombre de la aplicación. El valor proporcionado debe tener menos de 255 caracteres.	Obligatorio

## Salidas de audio

Esta es una lista de formatos de audio admitidos que se envían en cada solicitud como encabezado `X-Microsoft-OutputFormat`. Cada uno de ellos incorpora una velocidad de bits y el tipo de codificación. El servicio de voz admite salidas de audio de 24 kHz, 16 kHz y 8 kHz.

<code>raw-16khz-16bit-mono-pcm</code>	<code>raw-8khz-8bit-mono-mulaw</code>
<code>riff-8khz-8bit-mono-alaw</code>	<code>riff-8khz-8bit-mono-mulaw</code>
<code>riff-16khz-16bit-mono-pcm</code>	<code>audio-16khz-128kbitrate-mono-mp3</code>
<code>audio-16khz-64kbitrate-mono-mp3</code>	<code>audio-16khz-32kbitrate-mono-mp3</code>
<code>raw-24khz-16bit-mono-pcm</code>	<code>riff-24khz-16bit-mono-pcm</code>
<code>audio-24khz-160kbitrate-mono-mp3</code>	<code>audio-24khz-96kbitrate-mono-mp3</code>
<code>audio-24khz-48kbitrate-mono-mp3</code>	

### NOTE

Si la voz y el formato de salida seleccionados tienen velocidades de bits diferentes, se vuelve a muestrear el audio según sea necesario. Sin embargo, las voces de 24 kHz no admiten los formatos de salida `audio-16khz-16kbps-mono-siren` y `riff-16khz-16kbps-mono-siren`.

## Cuerpo de la solicitud

El cuerpo de cada solicitud `POST` se envía como [lenguaje de marcado de síntesis de voz \(SSML\)](#). SSML le permite elegir la voz y el idioma de la voz sintetizada que devuelve el servicio de texto a voz. Para ver una lista completa de voces compatibles, consulte [compatibilidad con idiomas](#).

### NOTE

Si usa una voz personalizada, el cuerpo de una solicitud puede enviarse como texto sin formato (ASCII o UTF-8).

## Solicitud de ejemplo

Esta solicitud HTTP utiliza SSML para especificar el idioma y la voz. El cuerpo no puede superar los 1000 caracteres.

```
POST /cognitiveservices/v1 HTTP/1.1
X-Microsoft-OutputFormat: raw-16khz-16bit-mono-pcm
Content-Type: application/ssml+xml
Host: westus.tts.speech.microsoft.com
Content-Length: 225
Authorization: Bearer [Base64 access_token]

<speak version='1.0' xml:lang='en-US'><voice xml:lang='en-US' xml:gender='Female'
name='en-US-JessaRUS'>
    Microsoft Speech Service Text-to-Speech API
</voice></speak>
```

Consulte nuestras guías de inicio rápido para ver ejemplos específicos del idioma:

- [.NET Core, C#](#)
- [Python](#)
- [Node.js](#)

## Códigos de estado HTTP

El estado HTTP de cada respuesta indica estados de corrección o error comunes.

CÓDIGO DE ESTADO HTTP	DESCRIPCIÓN	POSIBLE MOTIVO
200	OK	La solicitud es correcta; el cuerpo de la respuesta es un archivo de audio.
400	Bad Request	Falta un parámetro requerido, está vacío o es nulo. O bien, el valor pasado a un parámetro obligatorio u opcional no es válido. Un problema común es que el encabezado sea demasiado largo.
401	No autorizado	La solicitud no está autenticada. Asegúrese de que la clave de suscripción o el token sean válidos y de la región correcta.
413	Entidad de solicitud demasiado larga	La entrada de SSML tiene más de 1024 caracteres.
415	Tipo de medio no compatible	Es posible que se haya proporcionado el Content-Type incorrecto. Content-Type se debe establecer en application/ssml+xml .
429	Demasiadas solicitudes	Ha superado la cuota o la tasa de solicitudes permitidas para su suscripción.
502	Puerta de enlace incorrecta	Problema de red o de servidor. Podría indicar también encabezados no válidos.

Si el estado HTTP es 200 OK , el cuerpo de la respuesta contiene un archivo de audio en el formato solicitado. Este archivo se puede reproducir mientras se transfiere o guardarse en un búfer o en un archivo.

## Pasos siguientes

- [Obtenga su suscripción de prueba a Voz](#)
- [Personalización de modelos acústicos](#)
- [Personalización de modelos de lenguaje](#)

# Documentación de Swagger

13/01/2020 • 2 minutes to read • [Edit Online](#)

El servicio de voz ofrece una especificación de Swagger para interactuar con varias de las API REST que se usan para importar datos, crear modelos, probar la precisión de los modelos, crear puntos de conexión personalizados, poner en cola las transcripciones en lote y administrar las suscripciones. La mayoría de las operaciones disponibles a través del portal de Custom Speech se pueden completar mediante programación con estas API.

## NOTE

Se admiten las operaciones tanto de voz a texto como de texto a voz y están disponibles como API REST, que a su vez se documentan en la especificación de Swagger.

## Generación de código desde la especificación Swagger

La [especificación Swagger](#) tiene opciones que permiten probar rápidamente varias rutas de acceso. Sin embargo, a veces es conveniente generar código para todas las rutas de acceso, lo que crea una sola biblioteca de llamadas en la que se puedan basar las soluciones futuras. Echemos un vistazo al proceso para generar una biblioteca Python.

Tendrá que establecer Swagger en la misma región que su suscripción al servicio de voz. Puede confirmar la región en Azure Portal en el recurso de servicio de voz. Para ver una lista completa de las regiones admitidas, consulte [Regiones](#).

1. Vaya a <https://editor.swagger.io>.
2. Haga clic en **File** (Archivo) y, luego, en **Import** (Importar)
3. Escriba la dirección URL de Swagger, incluida la región de la suscripción del servicio de voz  
`https://<your-region>.cris.ai/docs/v2.0/swagger`.
4. Haga clic en **Generate Client** (Generar cliente) y seleccione Python
5. Guarde la biblioteca cliente

Puede usar la biblioteca Python que generó con los [ejemplos del servicio de voz en GitHub](#).

## Documentos de referencia

- [REST \(Swagger\): Batch transcription and customization](#) (API de REST: Transcripción y personalización de Azure Batch)
- [API REST: Speech-to-text](#) (API de REST: Voz a texto)
- [API REST: Text-to-speech](#) (API de REST: Texto a voz)

## Pasos siguientes

- [Ejemplos del servicio de voz en GitHub](#).
- [Obtenga una clave de suscripción gratuita a los servicios de Voz](#)

# Acerca del SDK de Voz

15/01/2020 • 6 minutes to read • [Edit Online](#)

El kit de desarrollo de software (SDK) de voz proporciona a sus aplicaciones acceso a las funciones del servicio Voz, lo que facilita el desarrollo de software habilitado para la voz. Actualmente, los SDK proporcionan acceso a **voz a texto**, **texto a voz**, **traducción de voz**, **reconocimiento de intenciones** y al **canal Direct Line Speech de Bot Framework**.

Puede capturar audio fácilmente desde un micrófono, leer de una secuencia o acceder a archivos de audio desde el almacenamiento con el SDK de voz. El SDK de voz admite WAV/PCM de 16 bits, 16 kHz u 8 kHz y audio de un solo canal para el reconocimiento de voz. Los formatos de audio adicionales se admiten mediante el [punto de conexión de REST de voz a texto](#) o el [servicio de transcripción por lotes](#).

Puede encontrar información general sobre las funcionalidades y las plataformas admitidas en la [página de entrada](#) de la documentación.

LENGUAJE DE PROGRAMACIÓN	PLATAFORMA	REFERENCIA DE API
C/C++	Windows, Linux, macOS	<a href="#">Browse</a>
C#	Windows, UWP, .NET Framework (Windows), .NET Core, Unity	<a href="#">Browse</a>
Java	Android, Windows, Linux, macOS	<a href="#">Browse</a>
Java*	Speech Devices SDK	<a href="#">Browse</a>
JavaScript/Node.js	Browser, Windows, Linux, macOS	<a href="#">Browse</a>
Objective-C	iOS, macOS	<a href="#">Browse</a>
Python	Windows, Linux, macOS	<a href="#">Browse</a>

\* El SDK de Java también está disponible como parte de [Speech Devices SDK](#).

## IMPORTANT

Al descargar cualquiera de los componentes del SDK de Voz de Azure Cognitive Services de esta página, acepta su licencia. Consulte los [términos de licencia del software de Microsoft para el SDK de Voz](#).

## Obtención del SDK

[Windows](#)

Para Windows, se admiten los siguientes lenguajes:

- C# (UWP y .NET), C++: Puede usar y hacer referencia a la versión más reciente del paquete de NuGet del SDK de Voz. El paquete incluye bibliotecas de cliente de 32 y de 64 bits, así como bibliotecas de cliente y administradas (.NET). El SDK se puede instalar en Visual Studio mediante NuGet, [Microsoft.CognitiveServices.Speech](#).
- Java: Puede usar y hacer referencia a la versión más reciente de nuestro paquete de Maven del SDK de Voz, que solo admite Windows x64. En el proyecto de Maven, agregue  
`https://csspeechstorage.blob.core.windows.net/maven/` como un repositorio adicional y la referencia  
`com.microsoft.cognitiveservices.speech:client-sdk:1.8.0` como una dependencia.

## Linux

### NOTE

Actualmente, solo se admite Ubuntu 16.04, Ubuntu 18.04 y Debian 9 en las siguientes arquitecturas de destino:

- x86, x64 y ARM64 para desarrollo de C++
- x64 y ARM64 para Java
- x64 para .NET Core y Python

Asegúrese de que tiene instaladas las bibliotecas necesarias. Para ello, ejecute los siguientes comandos de shell:

En Ubuntu:

```
sudo apt-get update
sudo apt-get install libssl1.0.0 libasound2
```

En Debian 9:

```
sudo apt-get update
sudo apt-get install libssl1.0.2 libasound2
```

- C#: Puede usar y hacer referencia a la versión más reciente del paquete de NuGet del SDK de Voz. Para hacer referencia a la SDK, agregue la siguiente referencia de paquete en el proyecto:

```
<PackageReference Include="Microsoft.CognitiveServices.Speech"
Version="1.8.0" />
```

- Java: Puede usar y hacer referencia a la versión más reciente del paquete de Maven del SDK de Voz. En el proyecto de Maven, agregue  
`https://csspeechstorage.blob.core.windows.net/maven/` como un repositorio adicional y la referencia  
`com.microsoft.cognitiveservices.speech:client-sdk:1.7.0` como una dependencia.

- C++: Descargue el SDK como un [paquete .tar](#) y descomprima los archivos en el directorio que prefiera. En la tabla siguiente se muestra la estructura de carpetas del SDK:

PATH	DESCRIPCIÓN
<code>license.md</code>	Licencia
<code>ThirdPartyNotices.md</code>	Avisos de terceros
<code>include</code>	Archivos de encabezado para C y C++
<code>lib/x64</code>	Biblioteca x64 nativa para vincular con la aplicación
<code>lib/x86</code>	Biblioteca x86 nativa para vincular con la aplicación

Para crear una aplicación, copie o mueva los binarios (y bibliotecas) necesarios a su entorno de desarrollo. Inclúyalos según sea necesario en el proceso de compilación.

## Android

El SDK de Java para Android está empaquetado como una [biblioteca de Android \(AAR\)](#), que incluye las bibliotecas necesarias, así como los permisos necesarios de Android. Se hospeda en un repositorio de Maven en <https://csspeechstorage.blob.core.windows.net/maven/> como un paquete `com.microsoft.cognitiveservices.speech:client-sdk:1.7.0`.

Para consumir el paquete desde el proyecto de Android Studio, haga los siguientes cambios:

- En el archivo build.gradle de nivel de proyecto, agregue lo siguiente a la sección `repository`:

```
maven { url 'https://csspeechstorage.blob.core.windows.net/maven/' }
```

- En el archivo build.gradle de nivel de módulo, agregue lo siguiente a la sección `dependencies`:

```
implementation 'com.microsoft.cognitiveservices.speech:client-sdk:1.7.0'
```

El SDK de Java es parte del [SDK de dispositivos de voz](#).

## Obtención de los ejemplos

Para ver los ejemplos más recientes, consulte el [repositorio de GitHub de ejemplos de código de Cognitive Services Speech SDK](#).

## Pasos siguientes

- [Obtenga su suscripción de prueba a Voz](#)

- Vea cómo funciona el reconocimiento de voz en C#

# Disponibilidad de escenarios

13/01/2020 • 2 minutes to read • [Edit Online](#)

El SDK de Voz presenta muchos escenarios en una amplia variedad de lenguajes y entornos de programación. No todos los escenarios están disponibles en todos los lenguajes de programación o todos los entornos. A continuación se indica la disponibilidad de cada escenario.

- **Reconocimiento de voz (SR), lista de frases, intención, traducción y contenedores locales**

- Todos los lenguajes de programación y entornos donde haya un vínculo de flecha ↗ en [esta tabla de inicio rápido](#).

- **Texto a voz (TTS)**

- C++/Windows y Linux
- C#/Windows, UWP y Unity
- Java (JRE y Android)
- Python
- Swift
- Objective-C
- La API REST de TTS puede usarse en todas las demás situaciones.

- **Detección de palabras clave (KWS)**

- C++/Windows y Linux
- C#/Windows & Linux
- Python/Windows y Linux
- Java/Windows, Linux y Android (SDK de dispositivos de voz)
- La funcionalidad de detección de palabras clave (KWS) podría funcionar con cualquier tipo de micrófono; no obstante, la compatibilidad oficial de KWS está limitada actualmente a las matrices de micrófonos que se encuentran en el hardware de Azure Kinect DK o el SDK de dispositivos de voz.

- **Asistentes de voz**

- C++/Windows, Linux y macOS
- C#/Windows
- Java/Windows, Linux, macOS y Android (SDK de dispositivos de voz)

- **Transcripción de conversaciones**

- C++/Windows y Linux
- C# (Framework y .NET Core)/Windows, UWP y Linux
- Java/Windows, Linux y Android (SDK de dispositivos de voz)

- **Transcripción para centros de llamadas**

- La API REST se puede usar en cualquier situación.

- **Entrada de audio comprimido con códec**

- C++/Linux
- C#/Linux
- Java/Linux, Android e iOS

# Inicio rápido: Crear un proyecto

13/01/2020 • 29 minutes to read • [Edit Online](#)

En este inicio rápido creará un proyecto vacío para su lenguaje de programación preferido que usará para completar un inicio rápido o para crear una aplicación.

## Selección del entorno de destino

- [Visual Studio](#)
- [Unity](#)
- [UWP](#)
- [Xamarin](#)

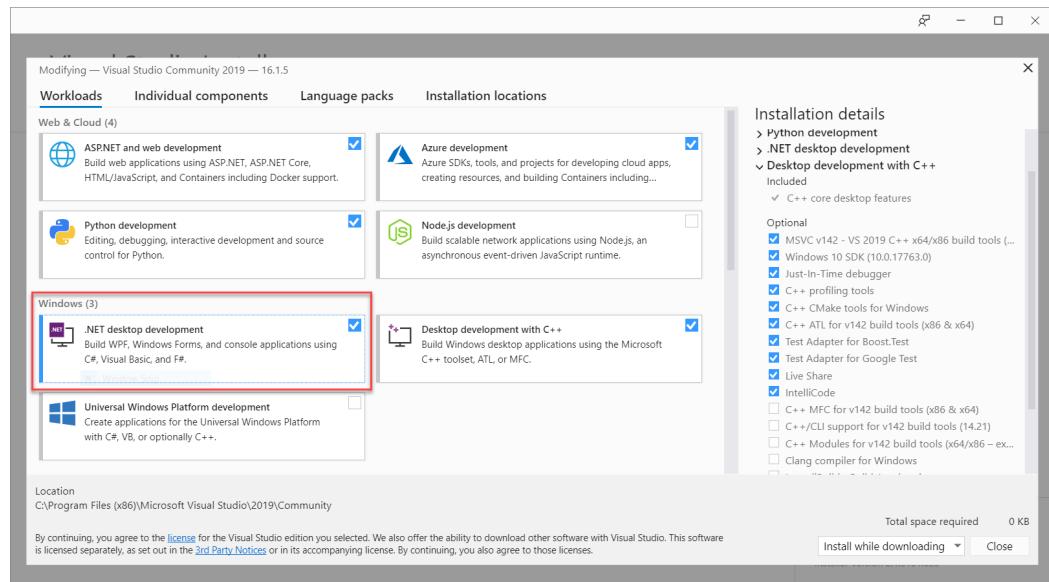
Para crear un proyecto de Visual Studio para el desarrollo de Windows, debe crear el proyecto, configurar Visual Studio para el desarrollo de escritorio de .NET, instalar el SDK de Voz y elegir la arquitectura de destino.

## Creación del proyecto e incorporación de la carga de trabajo

Para empezar, cree el proyecto en Visual Studio y asegúrese de que Visual Studio está configurado para el desarrollo de escritorio de .NET:

1. Abra Visual Studio 2019.
2. En la ventana Inicio, seleccione **Crear un proyecto**.
3. En la ventana **Crear un proyecto**, elija **Aplicación de consola (.NET Framework)** y seleccione **Siguiente**.
4. En la ventana **Configure su nuevo proyecto**, escriba *helloworld* en **Nombre del proyecto**, elija o cree la ruta de acceso del directorio en **Ubicación** y seleccione **Crear**.
5. En la barra de menús de Visual Studio, seleccione **Herramientas > Obtener herramientas y características**, que abre el Instalador de Visual Studio y muestra el cuadro de diálogo **Modificando**.
6. Compruebe si la carga de trabajo **Desarrollo de escritorio de .NET** está disponible. Si la carga de trabajo aún no se ha instalado, active la casilla que hay al lado y seleccione **Modificar** para iniciar la instalación. La descarga e instalación pueden tardar unos minutos.

Si la casilla que está junto a **Desarrollo de escritorio de .NET** ya está seleccionada, seleccione **Cerrar** para salir del cuadro de diálogo.

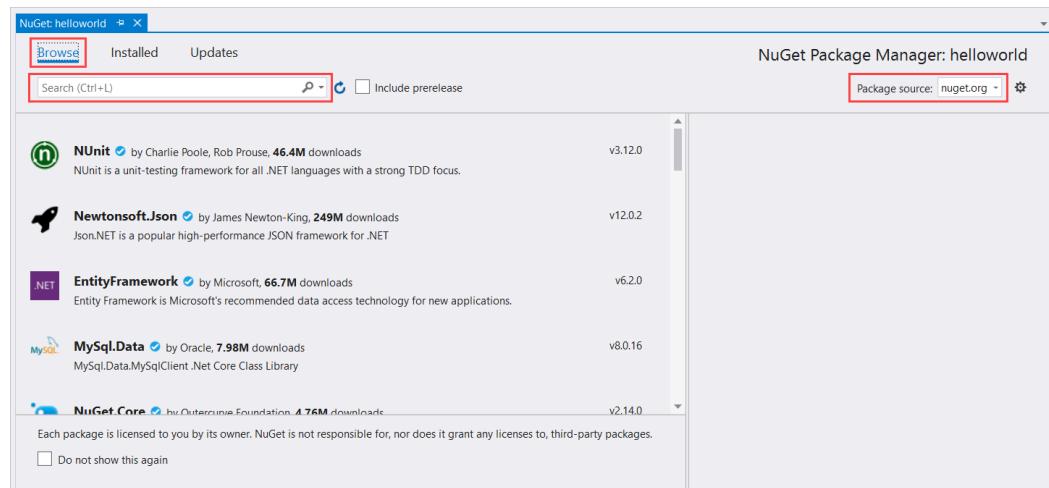


## 7. Cierre el Instalador de Visual Studio.

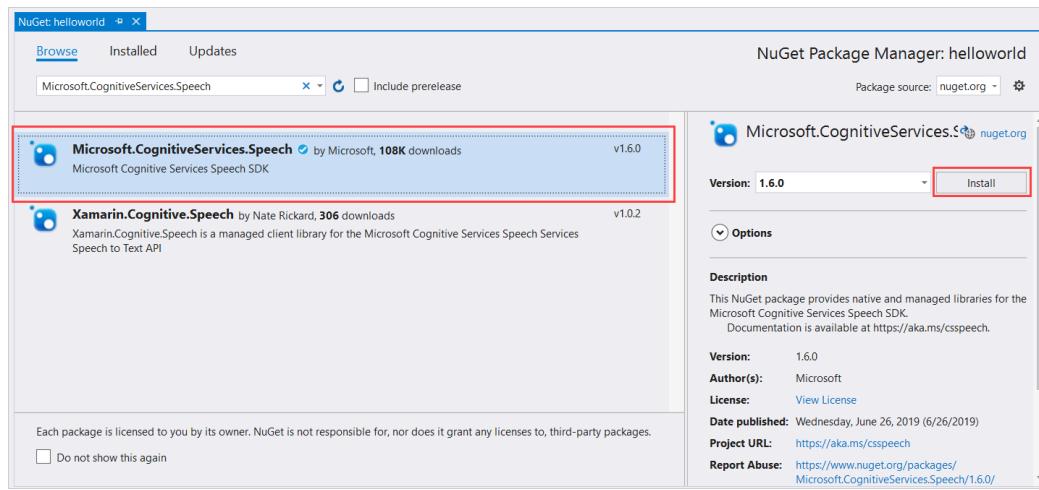
### Instalación de Speech SDK

El siguiente paso consiste en instalar el [paquete NuGet del SDK de Voz](#) para que pueda hacer referencia a él en el código.

1. En el Explorador de soluciones, haga clic con el botón derecho en el proyecto **helloworld** y seleccione **Administrar paquetes NuGet** para mostrar el Administrador de paquetes NuGet.



2. En la esquina superior derecha, busque el cuadro desplegable **Origen del paquete** y asegúrese de que **nuget.org** está seleccionado.
3. En la esquina superior izquierda, seleccione **Examinar**.
4. En el cuadro de búsqueda, escriba *Microsoft.CognitiveServices.Speech* y seleccione **Entrar**.
5. En los resultados de la búsqueda, seleccione el paquete **Microsoft.CognitiveServices.Speech** y, después, seleccione **Instalar** para instalar la versión estable más reciente.



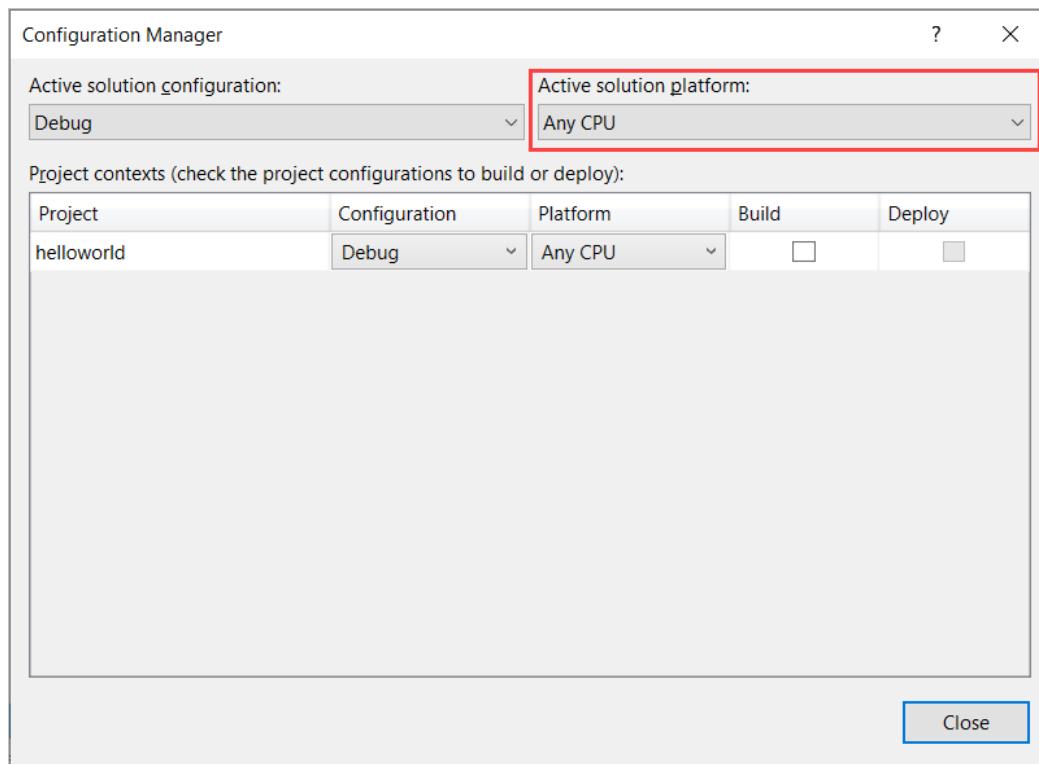
- Acepte todos los contratos y licencias para iniciar la instalación.

Después de instalar el paquete aparecerá una confirmación en la ventana **Consola del administrador de paquetes**.

#### Elección de la arquitectura de destino

Ahora, para compilar y ejecutar la aplicación de consola, cree una configuración de plataforma que coincida con la arquitectura del equipo.

- En la barra de menús, seleccione **Compilar > Administrador de configuración**. Aparecerá el cuadro de diálogo **Administrador de configuración**.



- En el cuadro desplegable **Active solution platform** (Plataforma de soluciones activas), seleccione **Nuevo**. Aparecerá el cuadro de diálogo **Nueva plataforma de solución**.
- En el cuadro desplegable **escriba o seleccione la nueva plataforma**:
  - Si está ejecutando Windows de 64 bits, seleccione **x64**.
  - Si está ejecutando Windows de 32 bits, seleccione **x86**.
- Seleccione **Aceptar** y, después, **Cerrar**.

#### Selección del entorno de destino

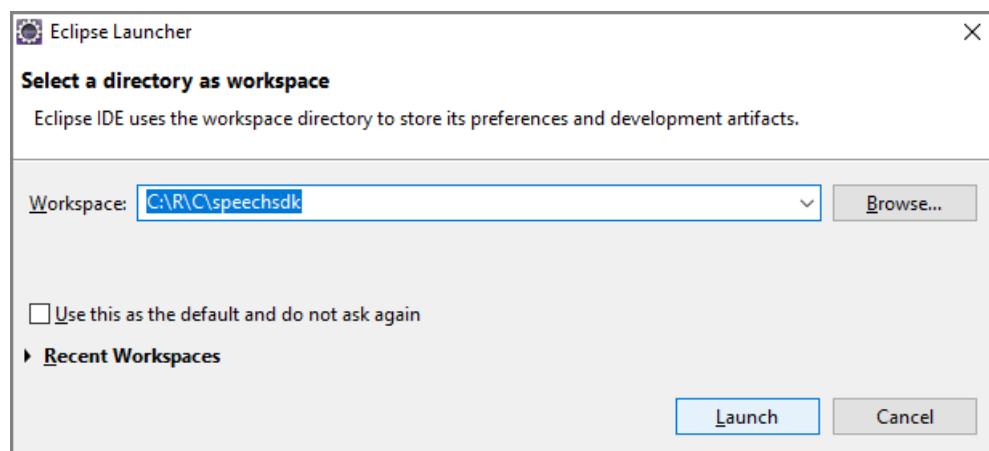
- Linux
- macOS
- Windows

En este ejemplo, se compilará con g++, por lo que todo lo que necesita para un proyecto vacío es crear un archivo helloworld.cpp con su editor de texto favorito.

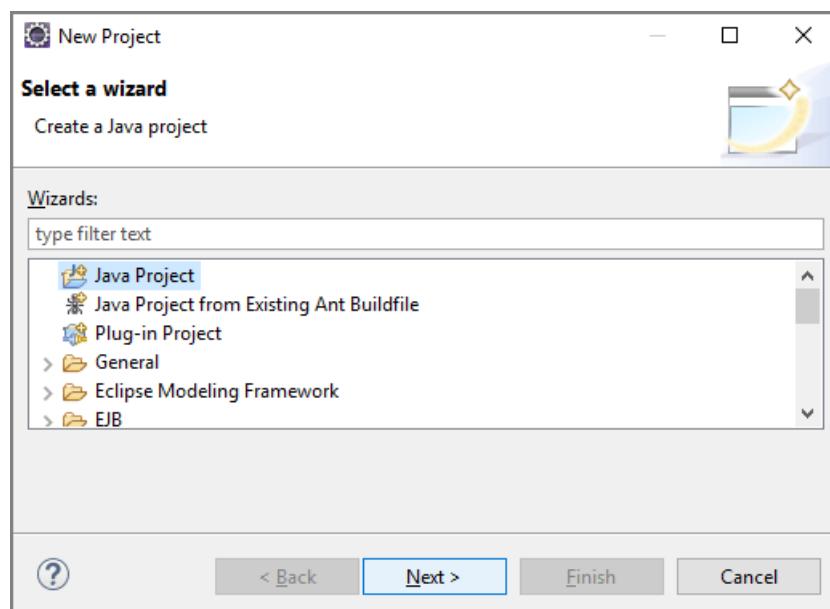
### Selección del entorno de destino

- Java Runtime
- Android

1. Inicie Eclipse.
2. En el selector de Eclipse, en el campo **Workspace** (Área de trabajo), escriba el nombre de un nuevo directorio de área de trabajo. Luego, seleccione **Launch** (Iniciar).

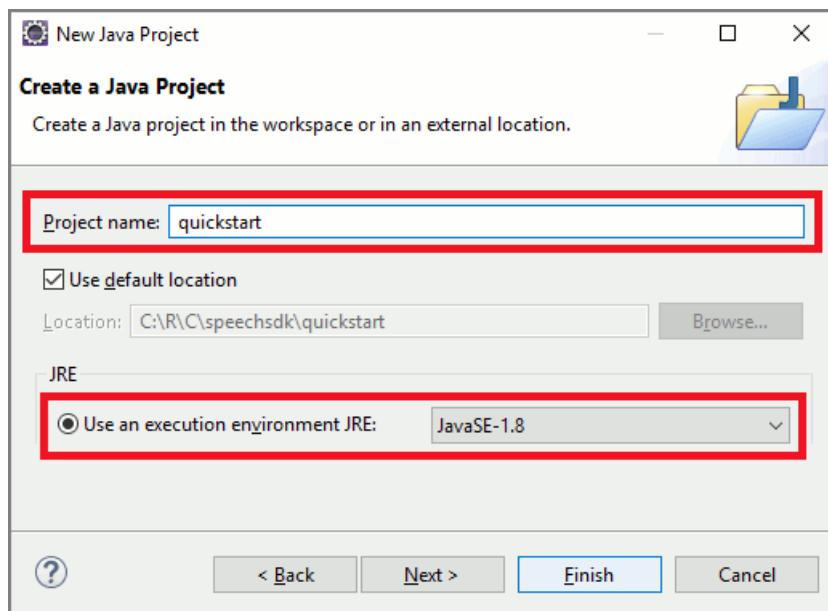


3. Al cabo de unos segundos, aparece la ventana principal del IDE de Eclipse. Cierre la pantalla de **bienvenida** si hay alguna.
4. En la barra de menús de Eclipse, cree un nuevo proyecto eligiendo **File (Archivo) > New (Nuevo) > Project (Proyecto)**.
5. Aparecerá el cuadro de diálogo **Nuevo proyecto**. Seleccione **Java Project** (Proyecto de Java) y seleccione **Next** (Siguiente).

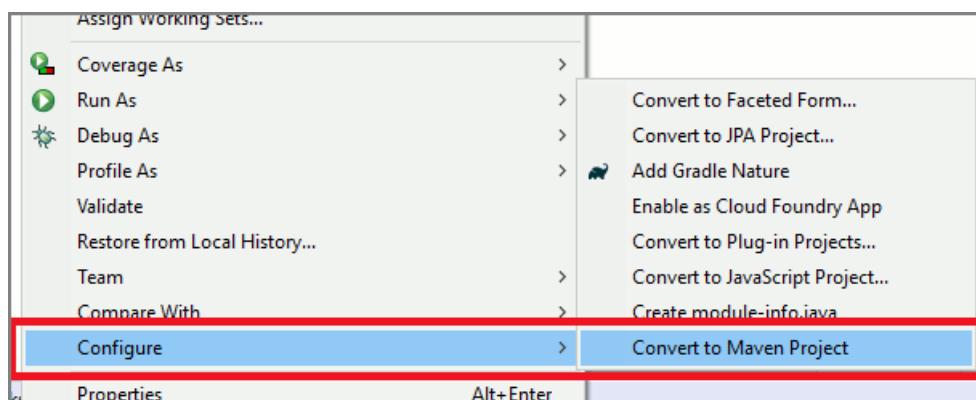


6. Se inicia el asistente para **nuevo proyecto de Java**. En el campo **Project name** (Nombre del

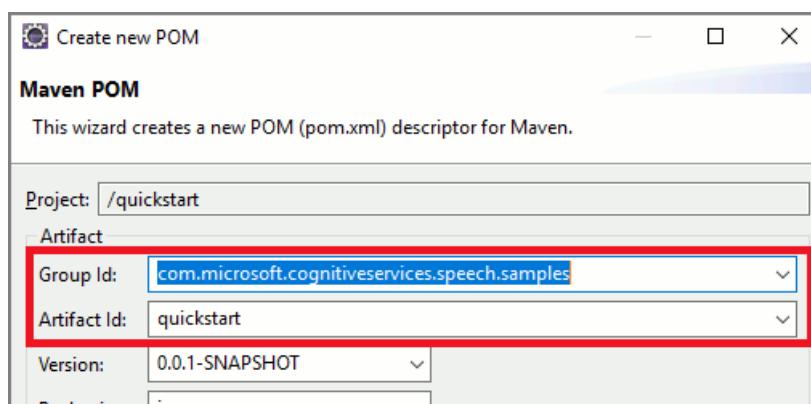
proyecto), escriba **quickstart** y elija **JavaSE-1.8** como el entorno de ejecución. Seleccione **Finalizar**.



7. Si aparece una ventana titulada **Open Associated Perspective?** (¿Abrir perspectiva asociada?), seleccione **Open Perspective** (Abrir perspectiva).
8. En el **Explorador de paquetes**, haga clic en el proyecto **quickstart**. Elija **Configure (Configurar) > Convert to Maven Project (Convertir en proyecto de Maven)** en el menú contextual.



9. Aparece la ventana **Create new POM** (Crear nuevo POM). En el campo **Group Id** (Identificador de grupo), escriba *com.microsoft.cognitiveservices.speech.samples*, y en el campo **Artifact Id** (Identificador de artefacto), escriba *quickstart*. A continuación, seleccione **Finish** (Finalizar).



10. Abra el archivo *pom.xml* y edítelo.

- Al final del archivo, antes cerrar la etiqueta de cierre `</project>`, cree un elemento `repositories` con una referencia al repositorio de Maven para el SDK de Voz, tal y como se muestra aquí:

```
<repositories>
<repository>
<id>maven-cognitiveservices-speech</id>
<name>Microsoft Cognitive Services Speech Maven Repository</name>
<url>https://csspeechstorage.blob.core.windows.net/maven/</url>
</repository>
</repositories>
```

- Agregue también un elemento `dependencies`, con la versión 1.7.0 del SDK de Voz como dependencia:

```
<dependencies>
<dependency>
<groupId>com.microsoft.cognitiveservices.speech</groupId>
<artifactId>client-sdk</artifactId>
<version>1.7.0</version>
</dependency>
</dependencies>
```

- Guarde los cambios.

En los inicios rápidos basados en Python, lo único que deberá hacer es crear un archivo llamado `helloworld.py` con su editor de texto o IDE favorito.

## Pasos siguientes

- [Inicio rápido: Reconocimiento de voz a través de un micrófono](#)
- [Inicio rápido: Reconocimiento de voz a partir de un archivo](#)
- [Inicio rápido: Traducción de voz a texto](#)
- [Inicio rápido: Síntesis de voz en texto](#)
- [Inicio rápido: Reconocimiento de intenciones](#)

# Inicio rápido: Configuración del entorno de desarrollo

15/01/2020 • 55 minutes to read • [Edit Online](#)

## Selección del entorno de destino

- [.NET](#)
- [.NET Core](#)
- [Unity](#)
- [UWP](#)
- [Xamarin](#)

En esta guía se muestra cómo instalar el [SDK de Voz para .NET Framework en Windows](#).

### IMPORTANT

Al descargar cualquiera de los componentes del SDK de Voz de Azure Cognitive Services de esta página, acepta su licencia. Consulte los [términos de licencia del software de Microsoft para el SDK de Voz](#).

## Prerequisites

Esta guía de inicio rápido requiere:

- [Visual Studio 2019](#)

## Creación de un proyecto de Visual Studio e instalación del SDK de Voz

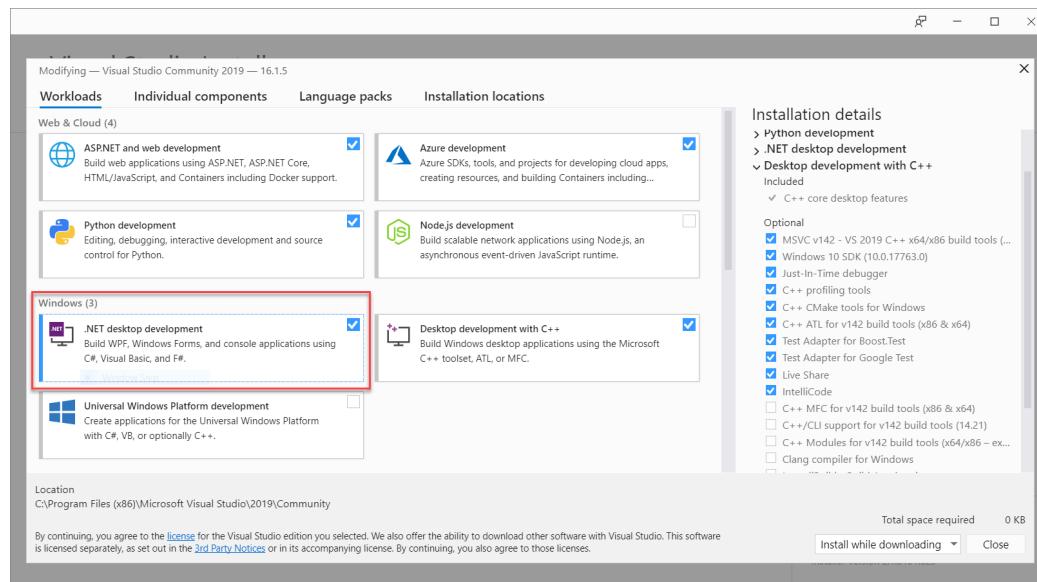
El siguiente paso consiste en instalar el [paquete NuGet del SDK de Voz](#) para que pueda hacer referencia a él en el código. Para ello, primero es necesario crear un proyecto **helloworld**. Si ya tiene un proyecto con la carga de trabajo de **desarrollo de escritorio de .NET** disponible, puede usar ese proyecto y pasar directamente a [Uso del Administrador de paquetes NuGet para instalar el SDK de Voz](#).

### Creación del proyecto helloworld

1. Abra Visual Studio 2019.
2. En la ventana Inicio, seleccione **Crear un proyecto**.
3. En la ventana **Crear un proyecto**, elija **Aplicación de consola (.NET Framework)** y seleccione **Siguiente**.
4. En la ventana **Configure su nuevo proyecto**, escriba *helloworld* en **Nombre del proyecto**, elija o cree la ruta de acceso del directorio en **Ubicación** y seleccione **Crear**.
5. En la barra de menús de Visual Studio, seleccione **Herramientas > Obtener herramientas y características**, que abre el Instalador de Visual Studio y muestra el cuadro de diálogo **Modificando**.
6. Compruebe si la carga de trabajo **Desarrollo de escritorio de .NET** está disponible. Si la

carga de trabajo aún no se ha instalado, active la casilla que hay al lado y seleccione **Modificar** para iniciar la instalación. La descarga e instalación pueden tardar unos minutos.

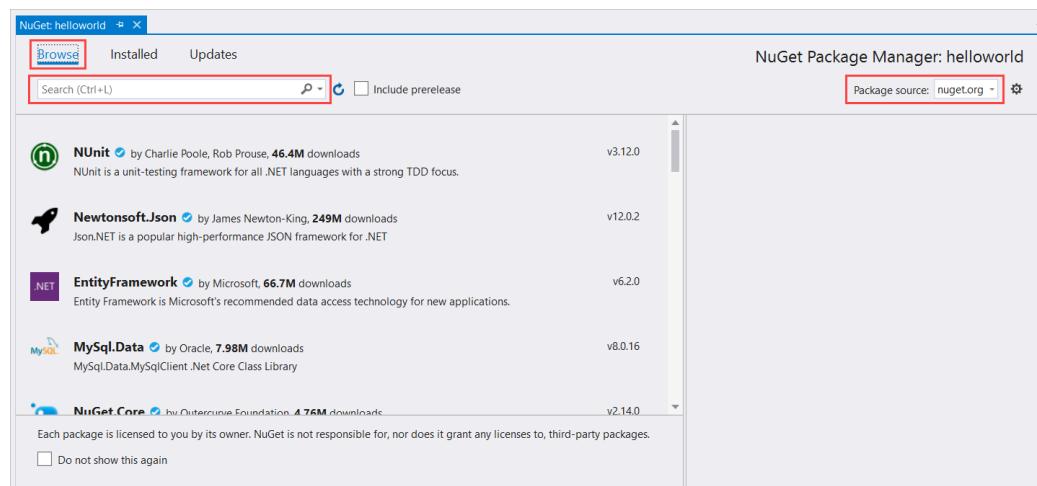
Si la casilla que está junto a **Desarrollo de escritorio de .NET** ya está seleccionada, seleccione **Cerrar** para salir del cuadro de diálogo.



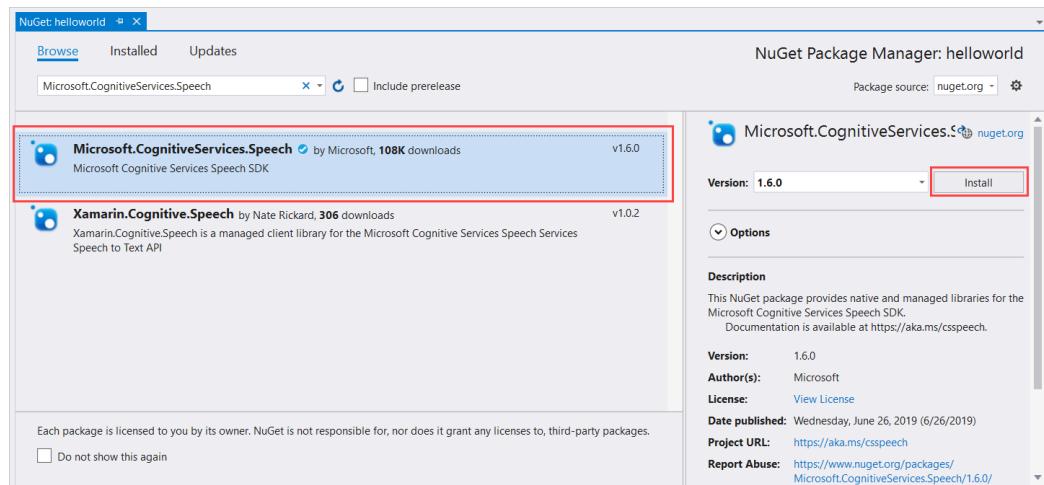
## 7. Cierre el Instalador de Visual Studio.

### Uso del Administrador de paquetes NuGet para instalar el SDK de Voz

1. En el Explorador de soluciones, haga clic con el botón derecho en el proyecto **helloworld** y seleccione **Administrar paquetes NuGet** para mostrar el Administrador de paquetes NuGet.



2. En la esquina superior derecha, busque el cuadro desplegable **Origen del paquete** y asegúrese de que `nuget.org` está seleccionado.
3. En la esquina superior izquierda, seleccione **Examinar**.
4. En el cuadro de búsqueda, escriba `Microsoft.CognitiveServices.Speech` y seleccione **Entrar**.
5. En los resultados de la búsqueda, seleccione el paquete **Microsoft.CognitiveServices.Speech** y, después, seleccione **Instalar** para instalar la versión estable más reciente.



## 6. Acepte todos los contratos y licencias para iniciar la instalación.

Después de instalar el paquete aparecerá una confirmación en la ventana **Consola del administrador de paquetes**.

Ahora puede continuar con la sección [Pasos siguientes](#).

## Pasos siguientes

- [Inicio rápido: Reconocimiento de voz a través de un micrófono](#)
- [Inicio rápido: Reconocimiento de voz a partir de un archivo](#)
- [Inicio rápido: Reconocimiento de voz de un Blob de Azure](#)
- [Inicio rápido: Traducción de voz a texto](#)
- [Inicio rápido: Síntesis de texto en un dispositivo de audio](#)
- [Inicio rápido: Síntesis de texto en un archivo](#)
- [Inicio rápido: Reconocimiento de intenciones](#)

### Selección del entorno de destino

- [Linux](#)
- [macOS](#)
- [Windows](#)

En esta guía se muestra cómo instalar el [SDK de Voz para Linux](#).

#### IMPORTANT

Al descargar cualquiera de los componentes del SDK de Voz de Azure Cognitive Services de esta página, acepta su licencia. Consulte los [términos de licencia del software de Microsoft para el SDK de Voz](#).

## Requisitos del sistema

Linux (Ubuntu 16.04, Ubuntu 18.04, Debian 9)

## Prerequisites

Para realizar este inicio rápido, necesita lo siguiente:

- [Visual Studio 2019](#)
- Las plataformas Linux admitidas requerirán la instalación de determinadas bibliotecas (

`libssl` para la compatibilidad con la capa de sockets seguros y `libasound2` para la compatibilidad con el audio). Consulte a continuación su distribución para saber cuáles son los comandos necesarios para instalar las versiones correctas de estas bibliotecas.

- En Ubuntu:

```
sudo apt-get update  
sudo apt-get install build-essential libssl1.0.0 libasound2 wget
```

- En Debian 9:

```
sudo apt-get update  
sudo apt-get install build-essential libssl1.0.2 libasound2 wget
```

## Instalación del SDK de Voz

El SDK de Voz para Linux se puede usar para crear aplicaciones de 64 bits y 32 bits. Las bibliotecas y los archivos de encabezado necesarios se pueden descargar como un archivo tar desde <https://aka.ms/csspeech/linuxbinary>.

Descargue e instale el SDK de la forma siguiente:

1. Seleccione el directorio al que desea extraer los archivos del SDK de Voz y configure la variable de entorno `SPEECHSDK_ROOT` para que apunte a ese directorio. Esta variable facilita la referencia al directorio en futuros comandos. Por ejemplo, si desea usar el directorio `speechsdk` en el directorio principal, use un comando similar al siguiente:

```
export SPEECHSDK_ROOT="$HOME/speechsdk"
```

2. Cree el directorio si aún no existe.

```
mkdir -p "$SPEECHSDK_ROOT"
```

3. Descargue y extraiga el archivo `.tar.gz` que contienen los archivos binarios del SDK de Voz:

```
wget -O SpeechSDK-Linux.tar.gz https://aka.ms/csspeech/linuxbinary  
tar --strip 1 -xzf SpeechSDK-Linux.tar.gz -C "$SPEECHSDK_ROOT"
```

4. Valide el contenido del directorio de nivel superior del paquete extraído:

```
ls -l "$SPEECHSDK_ROOT"
```

La lista de directorios debe contener los archivos de avisos y licencias de terceros, así como un directorio `include` que contenga archivos de encabezado (`.h`) y un directorio `lib` que contenga bibliotecas.

PATH	DESCRIPCIÓN
<code>license.md</code>	Licencia
<code>ThirdPartyNotices.md</code>	Avisos de terceros.

PATH	DESCRIPCIÓN
REDIST.txt	Aviso de redistribución.
include	Los archivos de encabezado necesarios para C y C++
lib/x64	Biblioteca nativa para x64 necesaria para vincular la aplicación
lib/x86	Biblioteca nativa para x86 necesaria para vincular la aplicación

Ahora puede continuar con la sección [Pasos siguientes](#).

## Pasos siguientes

- [Inicio rápido: Reconocimiento de voz a través de un micrófono](#)
- [Inicio rápido: Reconocimiento de voz a partir de un archivo](#)
- [Inicio rápido: Reconocimiento de voz de un Blob de Azure](#)
- [Inicio rápido: Traducción de voz a texto](#)
- [Inicio rápido: Síntesis de texto en un dispositivo de audio](#)
- [Inicio rápido: Síntesis de texto en un archivo](#)
- [Inicio rápido: Reconocimiento de intenciones](#)

### Selección del entorno de destino

- [Java Runtime](#)
- [Android](#)

En esta guía se muestra cómo instalar el [SDK de Voz](#) para Java 8 JRE de 64 bits.

#### NOTE

Para el SDK de dispositivos de Voz y el dispositivo Roobo, consulte [SDK de dispositivos de voz](#).

#### IMPORTANT

Al descargar cualquiera de los componentes del SDK de Voz de Azure Cognitive Services de esta página, acepta su licencia. Consulte los [términos de licencia del software de Microsoft para el SDK de Voz](#).

## Sistemas operativos admitidos

- El paquete del SDK de Voz para Java está disponible para estos sistemas operativos:
  - Windows: solo 64 bits
  - Mac: macOS X versión 10.13 o posterior
  - Linux: solo 64 bits en Ubuntu 16.04, Ubuntu 18.04 o Debian 9

## Prerequisites

- [Java 8 o JDK 8](#)

- [IDE de Java para Eclipse](#) (es necesario tener ya instalado Java)
- Las plataformas Linux admitidas requerirán la instalación de determinadas bibliotecas (`libssl` para la compatibilidad con la capa de sockets seguros y `libasound2` para la compatibilidad con el audio). Consulte a continuación su distribución para saber cuáles son los comandos necesarios para instalar las versiones correctas de estas bibliotecas.
  - En Ubuntu, ejecute los siguientes comandos para instalar los paquetes necesarios:

```
```sh
sudo apt-get update
sudo apt-get install build-essential libssl1.0.0 libasound2
```
```

```

- En Debian 9, ejecute los siguientes comandos para instalar los paquetes necesarios:

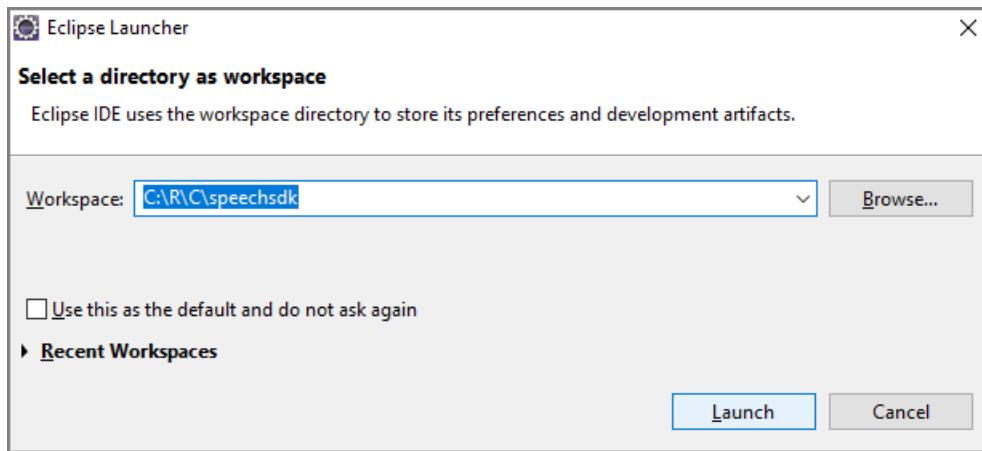
```
```sh
sudo apt-get update
sudo apt-get install build-essential libssl1.0.2 libasound2
```
```

```

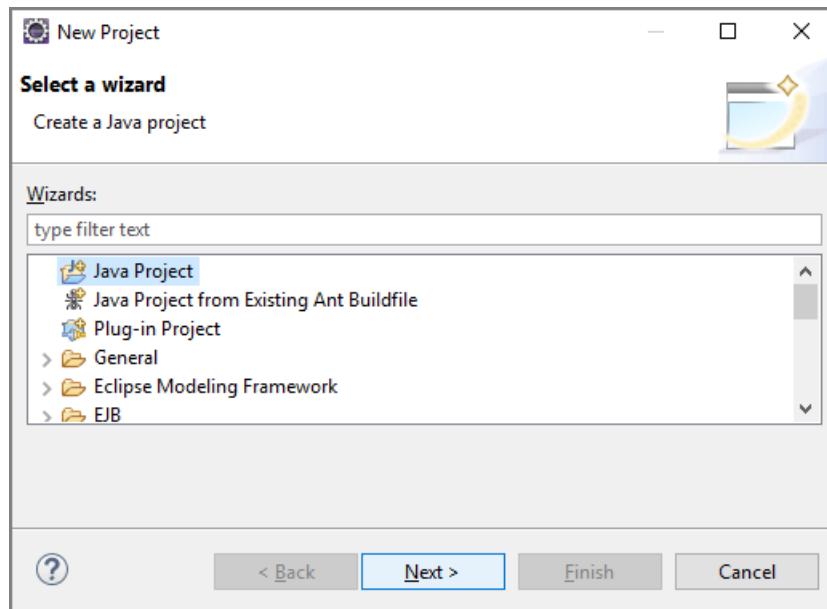
- En Windows, necesita [Microsoft Visual C++ Redistributable para Visual Studio 2019](#) para su plataforma. Tenga en cuenta que si es la primera vez que instala este paquete, puede que deba reiniciar Windows antes de seguir con esta guía.

## Creación de un proyecto de Eclipse e instalación del SDK de Voz

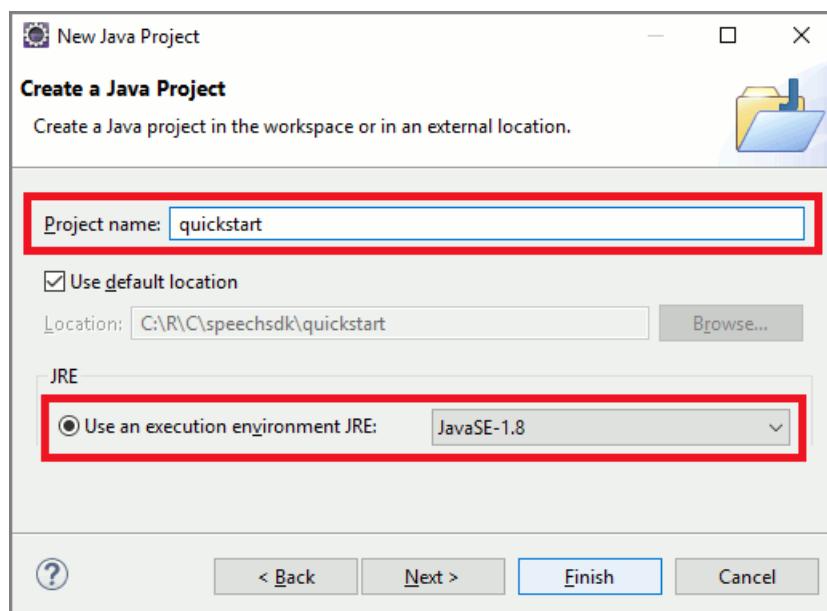
1. Inicie Eclipse.
2. En el selector de Eclipse, en el campo **Workspace** (Área de trabajo), escriba el nombre de un nuevo directorio de área de trabajo. Luego, seleccione **Launch** (Iniciar).



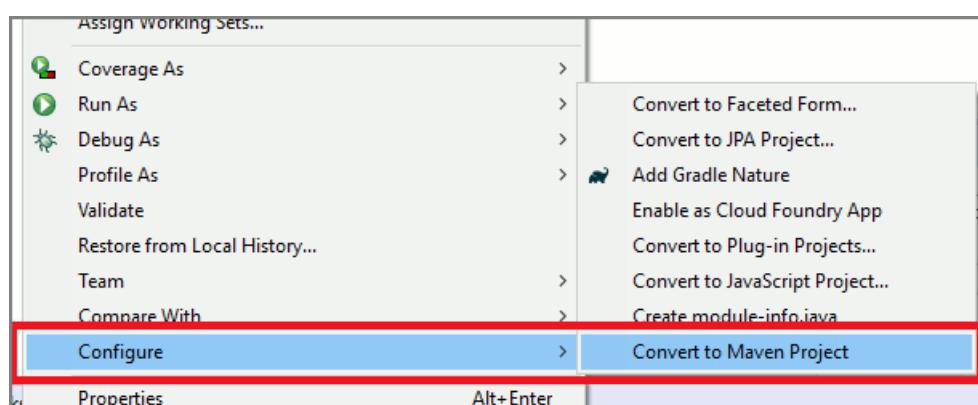
3. Al cabo de unos segundos, aparece la ventana principal del IDE de Eclipse. Cierre la pantalla de **bienvenida** si hay alguna.
4. En la barra de menús de Eclipse, cree un nuevo proyecto eligiendo **File (Archivo) > New (Nuevo) > Project (Proyecto)**.
5. Aparecerá el cuadro de diálogo **Nuevo proyecto**. Seleccione **Java Project** (Proyecto de Java) y seleccione **Next (Siguiente)**.



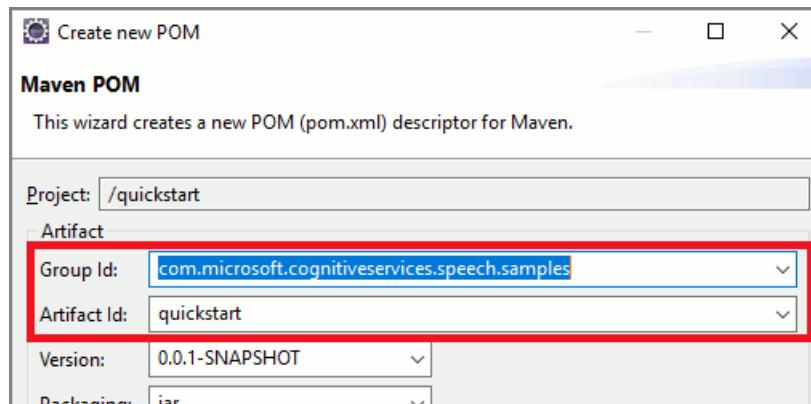
6. Se inicia el asistente para **nuevo proyecto de Java**. En el campo **Project name** (Nombre del proyecto), escriba **quickstart** y elija **JavaSE-1.8** como el entorno de ejecución. Seleccione **Finalizar**.



7. Si aparece una ventana titulada **Open Associated Perspective?** (¿Abrir perspectiva asociada?), seleccione **Open Perspective** (Abrir perspectiva).
8. En el **Explorador de paquetes**, haga clic en el proyecto **quickstart**. Elija **Configure (Configurar)** > **Convert to Maven Project (Convertir en proyecto de Maven)** en el menú contextual.



9. Aparece la ventana **Create new POM** (Crear nuevo POM). En el campo **Group Id** (Identificador de grupo), escriba `com.microsoft.cognitiveservices.speech.samples`, y en el campo **Artifact Id** (Identificador de artefacto), escriba `quickstart`. A continuación, seleccione **Finish** (Finalizar).



10. Abra el archivo `pom.xml` y edítelo.

- Al final del archivo, antes cerrar la etiqueta de cierre `</project>`, cree un elemento `repositories` con una referencia al repositorio de Maven para el SDK de Voz, tal y como se muestra aquí:

```
<repositories>
  <repository>
    <id>maven-cognitiveservices-speech</id>
    <name>Microsoft Cognitive Services Speech Maven Repository</name>
    <url>https://csspeechstorage.blob.core.windows.net/maven/</url>
  </repository>
</repositories>
```

- Agregue también un elemento `dependencies`, con la versión 1.7.0 del SDK de Voz como dependencia:

```
<dependencies>
  <dependency>
    <groupId>com.microsoft.cognitiveservices.speech</groupId>
    <artifactId>client-sdk</artifactId>
    <version>1.7.0</version>
  </dependency>
</dependencies>
```

- Guarde los cambios.

## Pasos siguientes

- [Inicio rápido: Reconocimiento de voz a través de un micrófono](#)
- [Inicio rápido: Reconocimiento de voz a partir de un archivo](#)
- [Inicio rápido: Reconocimiento de voz de un Blob de Azure](#)
- [Inicio rápido: Traducción de voz a texto](#)
- [Inicio rápido: Síntesis de texto en un dispositivo de audio](#)
- [Inicio rápido: Síntesis de texto en un archivo](#)
- [Inicio rápido: Reconocimiento de intenciones](#)

En esta guía se muestra cómo instalar el [SDK de Voz para Python](#).

#### IMPORTANT

Al descargar cualquiera de los componentes del SDK de Voz de Azure Cognitive Services de esta página, acepta su licencia. Consulte los [términos de licencia del software de Microsoft para el SDK de Voz](#).

## Sistemas operativos admitidos

- El paquete del SDK de Voz de Python está disponible para estos sistemas operativos:
  - Windows: x64 y x86
  - Mac: macOS X versión 10.12 o posterior
  - Linux: Ubuntu 16.04, Ubuntu 18.04, Debian 9 en x64

## Prerequisites

- Las plataformas Linux admitidas requerirán la instalación de determinadas bibliotecas (`libssl1` para la compatibilidad con la capa de sockets seguros y `libasound2` para la compatibilidad con el audio). Consulte a continuación su distribución para saber cuáles son los comandos necesarios para instalar las versiones correctas de estas bibliotecas.
  - En Ubuntu, ejecute los siguientes comandos para instalar los paquetes necesarios:

```
```sh
sudo apt-get update
sudo apt-get install build-essential libssl1.0.0 libasound2
```
```

- En Debian 9, ejecute los siguientes comandos para instalar los paquetes necesarios:

```
```sh
sudo apt-get update
sudo apt-get install build-essential libssl1.0.2 libasound2
```
```

- En Windows, necesita [Microsoft Visual C++ Redistributable para Visual Studio 2019](#) para su plataforma. Tenga en cuenta que si es la primera vez que instala este paquete, puede que deba reiniciar Windows antes de seguir con esta guía.
- Y, por último, necesitará [Python 3.5 o posterior](#). Para comprobar la instalación, abra un símbolo del sistema y escriba el comando `python --version` y compruebe el resultado. Si está instalado correctamente, obtendrá una respuesta "Python 3.5.1" o similar.

## Instalación del SDK de Voz mediante Visual Studio Code

1. Descargue e instale la última versión compatible de [Python](#) para la plataforma, la versión 3.5 o posterior.
  - Los usuarios de Windows deben asegurarse de que seleccionan "Add Python to your PATH" (Aregar Python a su RUTA) durante el proceso de instalación.
2. Descargue e instale [Visual Studio Code](#).
3. Abra Visual Studio Code e instale la extensión de Python. Seleccione **File > Preferences > Extensions** (Archivo > Preferencias > Extensiones) en el menú. Busque **Python** y haga clic en **Instalar**.



4. También desde dentro de Visual Studio Code, instale el paquete de Python del SDK de Voz desde la línea de comandos integrada:
  - a. Abra un terminal (en los menús desplegables **Terminal** > **Nuevo terminal**)
  - b. En el terminal que se abre, escriba el comando

```
python -m pip install azure-cognitiveservices-speech
```

Ya está listo para empezar a codificar en el SDK de Voz en Python y puede pasar a la sección [Pasos siguientes](#) a continuación. Si no está familiarizado con Visual Studio Code, consulte la [documentación más extensa sobre Visual Studio Code](#). Para más información sobre Visual Studio Code y Python, consulte el [tutorial de Python para Visual Studio Code](#).

## Instalación del SDK de Voz mediante la línea de comandos

Si no usa Visual Studio Code, el siguiente comando instala el paquete de Python desde [PyPI](#) para el SDK de Voz. Si usa Visual Studio Code, pase a la siguiente subsección.

```
pip install azure-cognitiveservices-speech
```

Si está en macOS, es posible que tenga que ejecutar el siguiente comando para que funcione el comando `pip` anterior:

```
python3 -m pip install --upgrade pip
```

Una vez que haya usado correctamente `pip` para instalar `azure-cognitiveservices-speech`, puede usar el SDK de Voz importando el espacio de nombres en los proyectos de Python. Por ejemplo:

```
import azure.cognitiveservices.speech as speechsdk
```

Esto se muestra con más detalle en los ejemplos de código que aparecen en la sección [Pasos siguientes](#) a continuación.

## Soporte técnico y actualizaciones

Las actualizaciones del paquete de Python del SDK de Voz se distribuirán mediante PyPI y se anunciarán en la página [Notas de la versión](#). Si hay disponible una nueva versión, puede actualizarse a ella con el comando `pip install --upgrade azure-cognitiveservices-speech`. Para comprobar qué versión está instalada actualmente, inspeccione la variable

```
azure.cognitiveservices.speech._version_.
```

Si tiene un problema o falta una característica, consulte las [opciones de ayuda y soporte técnico](#).

## Pasos siguientes

- Inicio rápido: Reconocimiento de voz a través de un micrófono
- Inicio rápido: Reconocimiento de voz a partir de un archivo
- Inicio rápido: Reconocimiento de voz de un Blob de Azure
- Inicio rápido: Traducción de voz a texto
- Inicio rápido: Síntesis de texto en un dispositivo de audio
- Inicio rápido: Síntesis de texto en un archivo
- Inicio rápido: Reconocimiento de intenciones

# Habilitar el registro en el SDK de voz

13/01/2020 • 5 minutes to read • [Edit Online](#)

El registro en un archivo es una característica opcional del SDK de voz. Durante el desarrollo, el registro proporciona información adicional y diagnósticos de los componentes principales del SDK de voz. Se puede habilitar estableciendo la propiedad `Speech_LogFilename` de un objeto de configuración de voz en la ubicación y el nombre del archivo de registro. El registro se activará globalmente una vez creado un reconocedor a partir de esa configuración que no se pueda deshabilitar más tarde. No se puede cambiar el nombre de un archivo de registro durante la ejecución de una sesión de registro.

## NOTE

El registro está disponible desde la versión 1.4.0 del SDK de voz en todos los lenguajes de programación admitidos a excepción de JavaScript.

## Muestra

El nombre del archivo de registro se especifica en un objeto de configuración. Tomando `SpeechConfig` como ejemplo y suponiendo que ha creado una instancia denominada `config`:

```
config SetProperty(PropertyId.Speech_LogFilename, "LogfilePathAndName");
```

```
config.setProperty(PropertyId.Speech_LogFilename, "LogfilePathAndName");
```

```
config->SetProperty(PropertyId::Speech_LogFilename, "LogfilePathAndName");
```

```
config.set_property(speechsdk.PropertyId.Speech_LogFilename, "LogfilePathAndName")
```

```
[config SetPropertyTo:@"LogfilePathAndName" byId:SPXSpeechLogFilename];
```

Puede crear un reconocedor a partir del objeto de configuración. Esto habilitará el registro de todos los reconocedores.

## NOTE

Si crea un `SpeechSynthesizer` a partir del objeto de configuración, no se habilitará el registro. No obstante, si el registro está habilitado, también recibirá los diagnósticos desde `SpeechSynthesizer`.

## Creación de un archivo de registro en distintas plataformas

Para Windows o Linux, el archivo de registro puede estar en cualquier ruta sobre la que el usuario tenga permiso de escritura. Los permisos de escritura en ubicaciones del sistema de archivos de otros sistemas operativos puede estar limitada o restringida de forma predeterminada.

## Plataforma universal de Windows (UWP)

Las aplicaciones de UWP requieren que los archivos de registro estén colocados en una de las ubicaciones de datos de la aplicación (local, roaming o temporal). Se puede crear un archivo de registro en la carpeta de la aplicación local:

```
StorageFolder storageFolder = ApplicationData.Current.LocalFolder;
StorageFile logFile = await storageFolder.CreateFileAsync("logfile.txt",
CreationCollisionOption.ReplaceExisting);
config SetProperty(PropertyId.Speech_LogFilename, logFile.Path);
```

[Aquí](#) hay más información sobre el permiso de acceso a los archivos de las aplicaciones de UWP.

## Android

Puede guardar un archivo de registro en un almacenamiento interno, externo o en el directorio de la memoria caché. Los archivos creados en el almacenamiento interno o en el directorio de la memoria caché son privados para la aplicación. Es preferible crear un archivo de registro en el almacenamiento externo.

```
File dir = context.getExternalFilesDir(null);
File logFile = new File(dir, "logfile.txt");
config SetProperty(PropertyId.Speech_LogFilename, logFile.getAbsolutePath());
```

El código anterior guarda un archivo de registro en el almacenamiento externo de la raíz de un directorio específico de la aplicación. Un usuario puede acceder al archivo con el administrador de archivos (normalmente en `Android/data/ApplicationName/logfile.txt`). El archivo se eliminará cuando se desinstale la aplicación.

También deberá solicitar el permiso `WRITE_EXTERNAL_STORAGE` sobre el archivo de manifiesto:

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android" package="...">
...
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
...
</manifest>
```

[Aquí](#) hay disponible más información sobre almacenamiento de datos y de archivos para aplicaciones de Android.

## iOS

Solo los directorios dentro del espacio aislado de la aplicación son accesibles. Los archivos se pueden crear en los directorios de documentos, biblioteca y temporales. Los archivos del directorio de documentos pueden estar a disposición de un usuario. El fragmento de código siguiente muestra la creación de un archivo de registro en el directorio de documentos de la aplicación:

```
NSString *filePath = [
[NSSearchPathForDirectoriesInDomains(NSDocumentDirectory, NSUserDomainMask, YES) firstObject]
stringByAppendingPathComponent:@"logfile.txt"];
[speechConfig SetPropertyTo:filePath byId:SPXSpeechLogFilename];
```

Para acceder a un archivo creado, agregue las siguientes propiedades a la lista de propiedades `Info.plist` de la aplicación:

```
<key>UIFileSharingEnabled</key>
<true/>
<key>LSSupportsOpeningDocumentsInPlace</key>
<true/>
```

[Aquí](#) hay más información disponible sobre el sistema de archivos de iOS.

## Pasos siguientes

[Exploración de ejemplos en GitHub](#)

# Seguimiento del uso de memoria del SDK de voz

15/01/2020 • 4 minutes to read • [Edit Online](#)

El SDK de voz se basa en una base de código nativo que se proyecta en varios lenguajes de programación a través de una serie de niveles de interoperabilidad. Cada proyección específica del lenguaje tiene características correctas de forma idiomática para administrar el ciclo de vida de los objetos. Además, el SDK de voz incluye herramientas de administración de memoria para realizar el seguimiento del uso de recursos con el registro de objetos y los límites de objetos.

## Cómo leer registros de objetos

Si [está habilitado el registro del SDK de voz](#), se emiten etiquetas de seguimiento para habilitar la observación histórica de objetos. Estas etiquetas incluyen:

- `TrackHandle` o `StopTracking`
- El tipo de objeto
- El número actual de objetos de los que se realiza el seguimiento del tipo del objeto y el número actual del que se realiza el seguimiento.

Aquí tiene un registro de muestra:

```
(284): 8604ms SPX_DBG_TRACE_VERBOSE: handle_table.h:90 TrackHandle
type=Microsoft::CognitiveServices::Speech::Impl::ISpxRecognitionResult handle=0x0x7f688401e1a0,
ptr=0x0x7f688401e1a0, total=19
```

## Defina un umbral de advertencia

Tiene la opción de crear un umbral de advertencia y, si se supera (suponiendo que el registro está habilitado), se registra un mensaje de advertencia. El mensaje de advertencia contiene un volcado de todos los objetos que existen junto con su recuento. Esta información se puede usar para comprender mejor los problemas.

Para habilitar un umbral de advertencia, debe especificarse en un objeto `SpeechConfig`. Este objeto se comprueba cuando se crea un nuevo reconocedor. En los ejemplos siguientes, supongamos que ha creado una instancia de `SpeechConfig` denominada `config`:

```
config SetProperty("SPEECH-ObjectCountWarnThreshold", "10000");
```

```
config->SetProperty("SPEECH-ObjectCountWarnThreshold", "10000");
```

```
config.setProperty("SPEECH-ObjectCountWarnThreshold", "10000");
```

```
speech_config.set_property_by_name("SPEECH-ObjectCountWarnThreshold", "10000")?
```

```
[config SetPropertyTo:@"10000" byName:"SPEECH-ObjectCountWarnThreshold"];
```

**TIP**

El valor predeterminado de esta propiedad es 10 000.

## Definir un umbral de error

Con el SDK de voz, puede establecer el número máximo de objetos permitidos en un momento dado. Si esta opción está habilitada, cuando se alcance el número máximo, se producirá un error al intentar crear nuevos objetos de reconocedor. Los objetos existentes seguirán funcionando.

Este es un error de muestra:

```
Runtime error: The maximum object count of 500 has been exceeded.  
The threshold can be adjusted by setting the SPEECH-ObjectCountErrorThreshold property on the SpeechConfig  
object.  
See http://https://docs.microsoft.com/azure/cognitive-services/speech-service/how-to-object-tracking-speech-sdk  
for more detailed information.  
Handle table dump by object type:  
class Microsoft::CognitiveServices::Speech::Impl::ISpxRecognitionResult 0  
class Microsoft::CognitiveServices::Speech::Impl::ISpxRecognizer 0  
class Microsoft::CognitiveServices::Speech::Impl::ISpxAudioConfig 0  
class Microsoft::CognitiveServices::Speech::Impl::ISpxSpeechConfig 0
```

Para habilitar un umbral de error, se debe especificar en un objeto `speechConfig`. Este objeto se comprueba cuando se crea un nuevo reconocedor. En los ejemplos siguientes, supongamos que ha creado una instancia de

`SpeechConfig` denominada `config`:

```
config SetProperty("SPEECH-ObjectCountErrorThreshold", "10000");
```

```
config->SetProperty("SPEECH-ObjectCountErrorThreshold", "10000");
```

```
config.setProperty("SPEECH-ObjectCountErrorThreshold", "10000");
```

```
speech_config.set_property_by_name("SPEECH-ObjectCountErrorThreshold", "10000")?
```

```
[config setPropertyTo:@"10000" byName:"SPEECH-ObjectCountErrorThreshold"];
```

**TIP**

El valor predeterminado de esta propiedad es el valor máximo específico de la plataforma para un tipo de datos `size_t`. Un reconocimiento típico utilizará entre 7 y 10 objetos internos.

## Pasos siguientes

- [Obtener la suscripción de evaluación gratuita del servicio de voz](#)
- [Más información sobre cómo reconocer la voz con un micrófono](#)

# Notas de la versión

15/01/2020 • 33 minutes to read • [Edit Online](#)

## SDK de Voz 1.8.0: Versión de noviembre de 2019

### Nuevas características

- Se ha agregado una API `FromHost()` para facilitar su uso con contenedores locales y nubes soberanas.
- Se ha agregado la detección automática de idioma de origen para el reconocimiento de voz (en Java y C++)
- Se ha agregado el objeto `SourceLanguageConfig` para el reconocimiento de voz, que se usa para especificar los idiomas de origen esperados (en Java y C++).
- Se ha agregado compatibilidad con `KeywordRecognizer` en Windows (UWP), Android e iOS mediante los paquetes de Nuget y Unity.
- Se ha agregado la API de Java de conversación remota para realizar la transcripción de conversaciones en lotes asíncronos.

### Cambios importantes

- Las funcionalidades de transcripción de conversaciones se han movido al espacio de nombres `Microsoft.CognitiveServices.Speech.Transcription`.
- Parte de los métodos de transcripción de conversaciones se han movido a la nueva clase `Conversation`.
- Compatibilidad eliminada para iOS de 32 bits (ARMv7 y x86)

### Correcciones de errores

- Se ha corregido un bloqueo si se usa `KeywordRecognizer` local sin una clave de suscripción válida al servicio de voz.

### Muestras

- Ejemplo de Xamarin para `KeywordRecognizer`
- Ejemplo de Unity para `KeywordRecognizer`
- Ejemplos de C++ y Java de detección automática de idioma de origen

## SDK de voz 1.7.0: versión de septiembre de 2019

### Nuevas características

- Compatibilidad con la versión beta agregada para Xamarin en la Plataforma universal de Windows (UWP), Android e iOS
- Compatibilidad con iOS agregada para Unity
- Se ha agregado compatibilidad con entradas `Compressed` para ALaw, Mulaw, FLAC en Android, iOS y Linux.
- Se ha agregado `SendMessageAsync` en la clase `Connection` para enviar un mensaje al servicio.
- Se ha agregado `SetMessageProperty` en la clase `Connection` para establecer la propiedad de un mensaje.
- TTS agregó compatibilidad para Java (JRE y Android), Python, Swift y Objective-C
- TTS agregó compatibilidad de reproducción para macOS, iOS y Android
- Se ha agregado información de "límite de palabras" para TTS

## Correcciones de errores

- Se ha corregido un problema de compilación de IL2CPP en Unity 2019 para Android
- Se ha corregido un problema con los encabezados con formato incorrecto en la entrada de archivo WAV que se procesa de forma incorrecta
- Se ha corregido un problema con UUID que no es único en algunas propiedades de conexión
- Se han corregido algunas advertencias sobre los especificadores de nulabilidad en los enlaces SWIFT (puede que se requieran pequeños cambios en el código)
- Se ha corregido un error que provocaba que las conexiones de WebSocket se cerraran de manera incorrecta en la carga de red
- Se ha corregido un problema en Android que a veces provoca que `DialogServiceConnector` use identificadores de impresión duplicados.
- Se han introducido mejoras en la estabilidad de las conexiones entre interacciones multiproceso y la generación de informes de errores (a través de eventos `Canceled`) cuando se producen con `DialogServiceConnector`.
- Los inicios de sesión de `DialogServiceConnector` ahora proporcionarán eventos correctamente, incluso si se llama a `ListenOnceAsync()` durante una operación `StartKeywordRecognitionAsync()` activa.
- Se ha resuelto un bloqueo asociado a la recepción de actividades `DialogServiceConnector`.

## Muestras

- Inicio rápido para Xamarin
- Se ha actualizado el inicio rápido de CPP con información de ARM64 de Linux
- Se ha actualizado el inicio rápido de Unity con información de iOS

# SDK de Voz 1.6.0: versión de junio de 2019

## Muestras

- Ejemplos de inicio rápido para Texto a voz en UWP y Unity
- Ejemplo de inicio rápido para Swift en iOS
- Ejemplos de Unity para Traducción y Reconocimiento de la intención comunicativa y Voz
- Ejemplos de inicios rápidos actualizados para `DialogServiceConnector`

## Mejoras y cambios

- Espacio de nombres de cuadro de diálogo:
  - El nombre de `SpeechBotConnector` ha cambiado a `DialogServiceConnector`
  - El nombre de `BotConfig` ha cambiado a `DialogServiceConfig`
  - `BotConfig::FromChannelSecret()` se ha reasignado a `DialogServiceConfig::FromBotSecret()`
  - Todos los clientes de Voz de Direct Line existentes siguen siendo compatibles después del cambio de nombre
- Actualización del adaptador REST de TTS para admitir una conexión persistente de proxy
- Un mejor mensaje de error cuando se pasa una región no válida
- Swift/Objective-C:
  - Mejores informes de errores: los métodos que pueden generar un error ahora se encuentran en dos versiones: una que expone un objeto `NSError` para el control de errores y una que genera una excepción. La primera se expone a Swift. Este cambio requiere adaptaciones en el código Swift existente.
  - Mejor control de eventos

## Correcciones de errores

- Corrección de TTS: donde el futuro de `SpeakTextAsync` se devolvió sin esperar al fin de la representación del audio
- Corrección para la serialización de las cadenas en C# para permitir la compatibilidad total con idiomas
- Corrección del problema de las aplicaciones centrales de .NET para cargar la biblioteca principal con un marco de destino net461 en ejemplos
- Corrección de problemas ocasionales para implementar bibliotecas nativas en la carpeta de salida en los ejemplos
- Corrección para cerrar el socket web de manera confiable
- Corrección de un posible bloqueo al abrir una conexión con una carga muy elevada en Linux
- Corrección de metadatos faltantes en el paquete de marcos para macOS
- Corrección de problemas con `pip install --user` en Windows

## Speech SDK 1.5.1

Se trata de una versión de corrección de errores y solo afecta al SDK nativo o administrado. No afecta a la versión de JavaScript del SDK.

### **Correcciones de errores**

- Corrección de FromSubscription cuando se usa con la transcripción de la conversación.
- Corrección de errores en la detección de palabras clave para los asistentes por voz.

## Speech SDK 1.5.0 Versión de mayo de 2019

### **Nuevas características:**

- La detección de palabras clave (KWS) ahora está disponible para Windows y Linux. La funcionalidad KWS podría funcionar con cualquier tipo de micrófono; no obstante, la compatibilidad oficial de KWS está limitada actualmente a las matrices de micrófonos que se encuentran en el hardware de Azure Kinect DK o el SDK de dispositivos de voz.
- La funcionalidad de sugerencia de frases está disponible a través del SDK. Para más información, consulte [esta página](#).
- La funcionalidad de transcripción de conversaciones está disponible a través del SDK. Consulte [aquí](#).
- Compatibilidad agregada con los asistentes por voz mediante el canal Direct Line Speech.

### **Muestras**

- Se han agregado ejemplos para nuevas características o nuevos servicios admitidos por el SDK.

### **Mejoras y cambios**

- Se han agregado varias propiedades de reconocimiento para ajustar el comportamiento del servicio o los resultados del servicio (por ejemplo, enmascaramiento de palabras soeces etc.).
- Ahora puede configurar el reconocimiento a través de las propiedades de configuración estándar, incluso si ha creado el valor de `FromEndpoint` del reconocedor.
- Objective-C: se agregó la propiedad `OutputFormat` a `SPXSpeechConfiguration`.
- El SDK ahora admite Debian 9 como una distribución de Linux.

### **Correcciones de errores**

- Se ha corregido un problema donde el recurso de altavoz se destruía demasiado pronto en la conversión de texto a voz.

## Speech SDK 1.4.2

Se trata de una versión de corrección de errores y solo afecta al SDK nativo o administrado. No afecta a la versión de JavaScript del SDK.

## Speech SDK 1.4.1

Esta es una versión solo para JavaScript. No se agregó ninguna característica. Se realizaron las siguientes correcciones:

- Se impide que el paquete web cargue https-proxy-agent.

## Speech SDK 1.4.0 Versión de abril de 2019

### Nuevas características:

- El SDK admite ahora el servicio de conversión de texto a voz en versión beta. Se admite en Windows y Linux Desktop desde C++ y C#. Para más información, consulte la [información general sobre la conversión de texto a voz](#).
- El SDK ahora admite archivos de audio MP3 y Opus/OGG como archivos de entrada de secuencia. Esta característica solo está disponible en Linux desde C++ y C# y está actualmente en versión beta (más detalles [aquí](#)).
- Speech SDK para Java, .NET Core, C++ y Objective-C ha conseguido compatibilidad con macOS. La compatibilidad de Objective-C con macOS está actualmente en versión beta.
- iOS: Speech SDK para iOS (Objective-C) ahora también se publica como una instancia de CocoaPod.
- JavaScript: compatibilidad con micrófono no predeterminada como dispositivo de entrada.
- JavaScript: compatibilidad con servidores proxy para Node.js.

### Muestras

- se han agregado ejemplos para usar Speech SDK con C++ y con Objective-C en macOS.
- Se han agregado ejemplos que muestran el uso del servicio de conversión de texto a voz.

### Mejoras y cambios

- Python: ahora se exponen propiedades adicionales de los resultados del reconocimiento mediante la propiedad `properties`.
- Para la compatibilidad adicional con el desarrollo y la depuración, puede redirigir la información de registro y diagnóstico del SDK a un archivo de registro (más información [aquí](#)).
- JavaScript: mejora del rendimiento del procesamiento de audio.

### Correcciones de errores

- Mac/iOS: se corrigió un error que daba lugar a una larga espera cuando no se podía establecer una conexión con el servicio de voz.
- Python: mejora del control de errores en los argumentos de las devoluciones de llamada de Python.
- JavaScript: se corrigieron los informes de estado erróneos de la voz que finalizaban en RequestSession.

## Speech SDK 1.3.1 Actualización de febrero de 2019

Se trata de una versión de corrección de errores y solo afecta al SDK nativo o administrado. No afecta a la versión de JavaScript del SDK.

### Corrección de error

- Se ha corregido una fuga de memoria cuando se usa la entrada de micrófono. No afecta a la entrada de archivos o basada en secuencias.

# Speech SDK 1.3.0: versión de febrero de 2019

## Nuevas características

- El SDK de voz admite la selección del micrófono de entrada mediante la clase `AudioConfig`. Esto permite transmitir datos de audio al servicio de voz desde un micrófono no predeterminado. Para más información, consulte la documentación en la que se describe cómo [seleccionar un dispositivo de entrada de audio](#). Esta característica aún no está disponible en JavaScript.
- Speech SDK ahora es compatible con Unity en una versión beta. Proporcione sus comentarios en la sección de problemas en el [repositorio de ejemplos de GitHub](#). Esta versión es compatible con Unity en Windows x86 y x64 (aplicaciones de escritorio o de la Plataforma universal de Windows) y Android (ARM32/64, x86). Puede encontrar más información en nuestra [guía de inicio rápido sobre Unity](#).
- El archivo `Microsoft.CognitiveServices.Speech.csharp.bindings.dll` (incluido en versiones anteriores) ya no es necesario. La funcionalidad está ahora integrada en el SDK principal.

## Muestras

El siguiente contenido nuevo está disponible en nuestro [repositorio de ejemplo](#):

- Ejemplos adicionales para `AudioConfig.FromMicrophoneInput`.
- Ejemplos adicionales de Python para traducción y reconocimiento de intenciones.
- Ejemplos adicionales para usar el objeto `Connection` en iOS.
- Ejemplos adicionales de Java para la traducción con la salida de audio.
- Nuevo ejemplo de uso de la [API de REST de transcripción de lotes](#).

## Mejoras y cambios

- Python
  - Mensajes de error y verificación de parámetros mejorada en `SpeechConfig`.
  - Adición de compatibilidad para el objeto `connection`.
  - Compatibilidad con Python (x86) de 32 bits en Windows.
  - Speech SDK para Python ya no está disponible como beta.
- iOS
  - El SDK ahora se compila en función de la versión 12.1 del SDK de iOS.
  - El SDK ahora es compatible con las versiones 9.2 y posteriores de iOS.
  - Documentación de referencia mejorada y corrección de varios nombres de propiedad.
- JavaScript
  - Adición de compatibilidad para el objeto `connection`.
  - Archivos de definición de tipos agregados para JavaScript agrupado.
  - Compatibilidad e implementación iniciales para sugerencias de frases.
  - Colección de propiedades devuelta con JSON del servicio para reconocimiento.
- Los archivos DLL de Windows contienen ahora un recurso de versión.
- Si crea un valor de `FromEndpoint` de reconocedor, puede agregar parámetros directamente a la dirección URL del punto de conexión. Con `FromEndpoint` no puede configurar el reconocedor mediante las propiedades de configuración estándar.

## Correcciones de errores

- La contraseña de proxy y el nombre de usuario de proxy vacíos no se administraron correctamente. Con esta versión, si establece el nombre de usuario de proxy y la contraseña de proxy en una cadena vacía, no se enviarán al conectarse al proxy.
- El identificador de sesión creado por el SDK no siempre es realmente aleatorio para algunos lenguajes

- o entornos. Se ha agregado la inicialización del generador aleatorio para corregir este problema.
- Control mejorado del token de autorización. Si desea usar un token de autorización, especifíquelo en `SpeechConfig` y deje la clave de suscripción vacía. A continuación, cree el reconocedor como de costumbre.
- En algunos casos, el objeto `Connection` no se publicó correctamente. Ahora se ha corregido.
- Se corrigió el ejemplo de JavaScript para admitir la salida de audio para la síntesis de traducción también en Safari.

## Speech SDK 1.2.1

Esta es una versión solo para JavaScript. No se agregó ninguna característica. Se realizaron las siguientes correcciones:

- Activar el final del flujo en `turn.end`, y no en `speech.end`.
- Corregir error de la bomba de audio por el que no se programaba el siguiente envío en caso de error del envío actual.
- Corregir el reconocimiento continuo con el token de autenticación.
- Corrección de errores de diferentes reconocedores y puntos de conexión.
- Mejoras en la documentación.

## Speech SDK 1.2.0: Versión de diciembre de 2018

### Nuevas características

- Python
  - La versión beta de la compatibilidad con Python (3.5 y versiones posteriores) está disponible con esta versión. Para más información, consulte [aquí](#)(quickstart-python.md).
- JavaScript
  - Speech SDK para JavaScript ha sido de código abierto. El código fuente está disponible en [GitHub](#).
  - Ya se admite Node.js; puede encontrar más información [aquí](#).
  - Se quitó la restricción de longitud para las sesiones de audio; la reconexión se realizará automáticamente en la portada.
- Objeto `Connection`
  - Desde el objeto `Recognizer`, puede acceder al objeto `Connection`. Este objeto le permite iniciar la conexión al servicio y suscribirse para conectar y desconectar eventos explícitamente. (Esta característica no está disponible aún ni en JavaScript ni en Python).
- Compatibilidad con Ubuntu 18.04.
- Android
  - Compatibilidad con ProGuard habilitada durante la generación del APK.

### Mejoras

- Mejoras en el uso de subprocessos internos, lo que reduce el número de subprocessos, bloqueos y exclusiones mutuas.
- Se mejoraron los informes de errores y la información. En algunos casos, los mensajes de error no se propagan totalmente.
- Se actualizaron las dependencias de desarrollo en JavaScript para usar los módulos actualizados.

### Correcciones de errores

- Se han corregido las fugas de causadas por un error de coincidencia de tipos en `RecognizeAsync`.

- En algunos casos, se perdieron excepciones.
- Corrección de las fugas de memoria en los argumentos de eventos de traducción.
- Se ha corregido un problema de bloqueo al volver a conectar en sesiones de larga ejecución.
- Se ha corregido un problema que podría dar lugar a que faltase el resultado final para las traducciones con errores.
- C#: Si no se esperaba una operación `async` en el subproceso principal, es posible que se pudiese desechar el reconocedor antes de completarse la tarea asíncrona.
- Java: Se ha corregido un problema que provocaba un bloqueo de la VM de Java.
- Objective-C: Se ha corregido la asignación de la enumeración; se devolvió `RecognizedIntent` en lugar de `RecognizingIntent`.
- JavaScript: Se ha establecido el formato de salida predeterminado en "simple" en `SpeechConfig`.
- JavaScript: Se ha quitado una incoherencia entre las propiedades del objeto de configuración en JavaScript y otros lenguajes.

## Muestras

- Se han actualizado y corregido varios ejemplos, como las voces de salida para la traducción, etc.
- Se han agregado ejemplos de Node.js en el [repositorio de ejemplo](#).

# Speech SDK 1.1.0

## Nuevas características

- Compatibilidad con Android x86/x64.
- Compatibilidad con proxy: En el objeto `SpeechConfig`, ahora puede llamar a una función para establecer la información del proxy (nombre de host, puerto, nombre de usuario y contraseña). Esta característica no está disponible aún en iOS.
- Mensajes y códigos de error mejorados. Si un reconocimiento devolvió un error, esto ya ha establecido `Reason` (en el evento cancelado) o `CancellationDetails` (en el resultado del reconocimiento) en `Error`. El evento cancelado ahora contiene dos miembros adicionales, `ErrorCode` y `ErrorDetails`. Si el servidor devolvió información de error adicional con el error notificado, ahora estará disponible en los nuevos miembros.

## Mejoras

- Verificación adicional agregada en la configuración del reconocedor y mensaje de error adicional agregado.
- Control mejorado del silencio prolongado en medio de un archivo de audio.
- Paquete NuGet: para proyectos de .NET Framework, evita la compilación con la configuración de AnyCPU.

## Correcciones de errores

- En los reconocedores se han encontrado varias excepciones corregidas. Además, las excepciones se detectan y se convierten en un evento `Canceled`.
- Corrección de una fuga de memoria en la administración de propiedades.
- Se corrigió el error en el que un archivo de entrada de audio podría bloquear el reconocedor.
- Se corrigió un error donde se podrían recibir eventos después de un evento de detención de la sesión.
- Se corrigieron algunas condiciones de subprocesos.
- Se corrigió un problema de compatibilidad de iOS que podría dar lugar a un bloqueo.
- Mejoras de estabilidad para la compatibilidad del micrófono en Android.
- Se corrigió un error donde un reconocedor en JavaScript ignoraría el lenguaje de reconocimiento.

- Se corrigió un error que impedía establecer el valor `EndpointId` (en algunos casos) en JavaScript.
- Se cambió el orden de los parámetros en `AddIntent` en JavaScript y se agregó la firma de JavaScript `AddIntent` que faltaba.

## Muestras

- Se han agregado ejemplos de C++ y C# para el uso de transmisiones de inserción y extracción en el [repositorio de ejemplos](#).

## Speech SDK 1.0.1

Mejoras en la confiabilidad y correcciones de errores:

- Corrección de un potencial error grave debido a una condición de carrera al desechar un reconocedor.
- Corrección de un potencial error grave en el caso de propiedades sin establecer.
- Comprobación adicional de errores y parámetros.
- Objective-C: corrección de posibles errores graves causados por la invalidación de nombres en `NSString`.
- Objective-C: ajuste de visibilidad en la API
- JavaScript: corrección con respecto a los eventos y sus cargas.
- Mejoras en la documentación.

Se ha agregado un nuevo ejemplo de Javascript en nuestro [repositorio de ejemplos](#).

## SDK de Voz 1.0.0 de Cognitive Services: Versión de septiembre de 2018

### Nuevas características:

- Compatibilidad con Objective-C en iOS. Consulte la [Guía de inicio rápido de Objective-C para iOS](#).
- Se admite JavaScript en el explorador. Consulte la [Guía de inicio rápido de JavaScript](#).

### Cambios importantes

- Con esta versión se presentan una serie de cambios importantes. Consulte [esta página](#) para más información.

## SDK de Voz 0.6.0 de Cognitive Services: Versión de agosto de 2018

### Nuevas características:

- Ahora, las aplicaciones de UWP creadas con SDK de Voz superan el Kit para la certificación de aplicaciones en Windows (WACK). Consulte la [Guía de inicio rápido de UWP](#).
- Compatibilidad con .NET Standard 2.0 en Linux (Ubuntu 16.04 x64).
- Experimental: compatibilidad con Java 8 en Windows (64 bits) y Linux (Ubuntu 16.04 x 64). Consulte la [Guía de inicio rápido de Java Runtime Environment](#).

### Cambios funcionales

- Se expone más información detallada sobre los errores de conexión.

### Cambios importantes

- En Java (Android), la función `SpeechFactory.configureNativePlatformBindingWithDefaultCertificate` ya no requiere un parámetro de ruta de acceso. Ahora, la ruta de acceso se detecta automáticamente en todas las plataformas compatibles.

- En Java y C#, se ha quitado el descriptor de acceso get- de la propiedad `EndpointUrl`.

### Correcciones de errores

- En Java, se implementa ahora el resultado de la síntesis de audio en el reconocedor de traducción.
- Se ha corregido un error que podía provocar subprocessos inactivos y un mayor número de sockets abiertos y sin usar.
- Se ha corregido un problema por el que un proceso de reconocimiento de larga ejecución podía terminar en mitad de la transmisión.
- Se ha corregido una condición de carrera en el proceso de apagado del reconocedor.

## SDK de Voz 0.5.0 de Cognitive Services: Versión de julio de 2018

### Nuevas características:

- Compatibilidad con la plataforma Android (API 23: Android Marshmallow 6.0 o posterior). Consulte el [inicio rápido de Android](#).
- Compatibilidad con .NET Standard 2.0 en Windows. Consulte el [inicio rápido de .NET Core](#).
- Experimental: compatibilidad con UWP en Windows (versión 1709 o posterior).
  - Consulte la [Guía de inicio rápido de UWP](#).
  - Nota: Las aplicaciones de UWP creadas con el SDK de Voz no pasan aún el Kit para la certificación de aplicaciones en Windows (WACK).
- Compatibilidad con el reconocimiento de ejecución prolongada con reconexión automática.

### Cambios funcionales

- `StartContinuousRecognitionAsync()` admite reconocimiento de ejecución prolongada.
- El resultado del reconocimiento contiene más campos. Tienen un desplazamiento desde el principio del audio y la duración (ambos en tics) del texto reconocido y valores adicionales que representan el estado de reconocimiento, por ejemplo, `InitialSilenceTimeout` e `InitialBabbleTimeout`.
- Compatibilidad con `AuthorizationToken` para la creación de instancias de fábrica.

### Cambios importantes

- Eventos de reconocimiento: el tipo de evento `NoMatch` se combina con el evento `Error`.
- `SpeechOutputFormat` en C# se llama ahora `OutputFormat` para concordar con C++.
- El tipo de valor devuelto de algunos métodos de la interfaz `AudioInputStream` se ha modificado ligeramente:
  - En Java, el método `read` ahora devuelve `long` en lugar de `int`.
  - En C#, el método `Read` ahora devuelve `uint` en lugar de `int`.
  - En C++, los métodos `Read` y `GetFormat` ahora devuelven `size_t` en lugar de `int`.
- C++: las instancias de secuencias de entrada de audio ahora solo se pueden pasar como un valor `shared_ptr`.

### Correcciones de errores

- Se han corregido los valores devueltos incorrectos cuando se agota el tiempo de espera de `RecognizeAsync()`.
- Se ha eliminado la dependencia de las bibliotecas de Media Foundation en Windows. El SDK ahora usa las API de audio básicas.
- Corrección de la documentación: se ha agregado una página de [regiones](#) para describir cuáles son las regiones admitidas.

### Problema conocido

- SDK de Voz para Android no informa de los resultados de la síntesis de voz para la traducción. Este problema se solucionará en la próxima versión.

## SDK de Voz 0.4.0 de Cognitive Services: Versión de junio de 2018

### Cambios funcionales

- `AudioInputStream`

Un reconocedor ahora puede consumir una secuencia como origen de audio. Para más información, consulte la [guía de procedimientos](#) relacionada.

- Formato de salida detallado

Al crear un elemento `SpeechRecognizer`, puede solicitar el formato de salida `Detailed` o `Simple`. `DetailedSpeechRecognitionResult` contiene una puntuación de confianza, texto reconocido, formato léxico sin formato, formato normalizado y formato normalizado con palabras soeces enmascaradas.

### Cambio importante

- En C# se cambia de `SpeechRecognitionResult.RecognizedText` a `SpeechRecognitionResult.Text`.

### Correcciones de errores

- Se ha corregido un posible problema de devolución de llamada en la capa USP durante el apagado.
- Si un reconocedor usaba un archivo de entrada de audio, mantenía el identificador de archivo más tiempo del necesario.
- Se han eliminado varios interbloqueos entre el suministro de mensajes y el reconocedor.
- Se desencadena un resultado `NoMatch` cuando se agota la respuesta del servicio.
- Las bibliotecas de Media Foundation en Windows son de carga retrasada. Esta biblioteca solo es necesaria para la entrada del micrófono.
- La velocidad de carga de los datos de audio se limita al doble de la velocidad de audio original.
- En Windows, los ensamblados .NET de C# ahora son de nombre seguro.
- Corrección de la documentación: `Region` necesita información para crear un reconocedor.

Se han agregado más ejemplos y se actualizan constantemente. Para obtener el conjunto más reciente de ejemplos, consulte el [repositorio de GitHub de ejemplos de SDK de Voz](#).

## SDK de Voz 0.2.12733 de Cognitive Services: Versión de mayo de 2018

Esta versión es la primera versión preliminar pública de SDK de Voz de Cognitive Services.

# Introducción a Speech Devices SDK

13/01/2020 • 4 minutes to read • [Edit Online](#)

El [servicio de voz](#) funciona con una amplia variedad de dispositivos y orígenes de audio. Ahora puede llevar las aplicaciones de voz al siguiente nivel con hardware y software coincidente. El SDK de dispositivos de voz es una biblioteca preajustada que está emparejada con kits de desarrollo específicos integrados de la matriz del micrófono.

El SDK de dispositivos de voz puede ayudarle a:

- Probar rápidamente nuevos escenarios de voz.
- Integrar más fácilmente el servicio de voz basado en la nube en el dispositivo.
- Crear una experiencia de usuario excepcional para sus clientes.

El SDK de dispositivos de voz consume el [SDK de voz](#). Usa el SDK de voz para enviar el audio que procesa nuestro algoritmo de procesamiento avanzado de audio desde la matriz del micrófono del dispositivo al [servicio de voz](#). Usa audio multicanal para proporcionar un [reconocimiento de voz](#) de campo lejano más preciso mediante la supresión de ruido, la cancelación del eco, la formación de haces y la eliminación de la reverberación.

También puede usar el SDK de dispositivos de voz para compilar dispositivos ambientales que tienen su propia [palabra clave personalizada](#), de modo que la indicación que inicia la interacción del usuario sea única para su marca.

El SDK de dispositivos de voz facilita diversos escenarios habilitados para voz, como [asistentes de voz](#), sistemas de pedidos de comida para llevar, [transcripción de conversaciones](#) y altavoces inteligentes. Puede responder a los usuarios con texto, hablarles con una voz predeterminada o [personalizada](#), proporcionar resultados de búsqueda, [traducir](#) a otros idiomas y mucho más. Estamos deseando ver lo que ha compilado.

## Obtener el SDK de dispositivos de voz

### Android

En el caso los dispositivos Android, descargue la versión más reciente del [SDK para dispositivos de voz de Android](#).

### Windows

En el caso de Windows, la aplicación de ejemplo que se proporciona es una aplicación de Java multiplataforma. Descargue la última versión del [SDK para dispositivos de voz de JRE](#). La aplicación se crea con el paquete del SDK de Voz y el entorno de desarrollo integrado de Java de Eclipse (v4) en Windows de 64 bits. Se ejecuta en un entorno de tiempo de ejecución de Java 8 (JRE) de 64 bits.

### Linux

En el caso de Linux, la aplicación de ejemplo que se proporciona es una aplicación de Java multiplataforma. Descargue la última versión del [SDK para dispositivos de voz de JRE](#). La aplicación se crea con el paquete del SDK de Voz y el entorno de desarrollo integrado de Java de Eclipse (v4) en Linux de 64 bits (Ubuntu 16.04, Ubuntu 18.04 y Debian 9). Se ejecuta en un entorno de tiempo de ejecución de Java 8 (JRE) de 64 bits.

## Pasos siguientes

[Elija el dispositivo de voz](#)

Obtenga una clave de suscripción gratuita a los servicios de Voz

# Obtener el SDK de dispositivos de voz de Cognitive Services

13/01/2020 • 2 minutes to read • [Edit Online](#)

El SDK de dispositivos de voz es una biblioteca preajustada diseñada para funcionar con kits de desarrollo específicos integrados y diversas configuraciones de matrices de micrófonos.

## Selección de un kit de desarrollo

DISPOSITIVOS	ESPECIFICACIÓN	DESCRIPCIÓN	ESCENARIOS
<a href="#">Roobo Smart Audio Dev Kit</a> <a href="#">Configuración / Inicio rápido</a> 	Matriz de 7 micrófonos, ARM SOC, Wi-Fi, salida de audio, E/S. <a href="#">Android</a>	El primer SDK de dispositivos de voz en adaptar la matriz de micrófonos de Microsoft y el SDK de procesamiento frontal para el desarrollo de escenarios de transcripción y voz de alta calidad	Transcripción de conversaciones, altavoz inteligente, agente por voz,ponible
<a href="#">Azure Kinect DK</a> <a href="#">Instalación / Inicio rápido</a> 	Matriz de 7 micrófonos, cámaras RGB y de profundidad <a href="#">Windows/Linux</a>	Un kit de desarrollo con sensores avanzados de inteligencia artificial (AI) para la creación de modernos modelos de voz y visión artificial. Combina una de las mejores matrices de micrófonos espaciales y cámaras de profundidad con una cámara de video y un sensor de orientación, todo ello en un pequeño dispositivo con varios modos, opciones y SDK para dar cabida a una amplia variedad de tipos de proceso.	Transcripción de conversaciones, robótica, edificios inteligentes
Roobo Smart Audio Dev Kit 2 	Matriz de 7 micrófonos, ARM SOC, Bluetooth, E/S. Linux	El SDK de dispositivos de voz de segunda generación que proporciona un sistema operativo alternativo y más características en un diseño de referencia rentable.	Transcripción de conversaciones, altavoz inteligente, agente por voz,ponible

DISPOSITIVOS	ESPECIFICACIÓN	DESCRIPCIÓN	ESCENARIOS
Placa de desarrollo URbetter T11 	Matriz de 7 micrófonos, ARM SOC, Wi-Fi, Ethernet, HDMI, cámara USB. Linux	Un SDK de dispositivos de voz de nivel industrial que se adapta a la matriz de micrófonos de Microsoft y es compatible con E/S extendida como HDMI/Ethernet y más periféricos USB	Transcripción de conversaciones, educación, hospitales, robots, caja OTT, agente por voz, servicio de comidas rápidas

## Descargar el SDK de dispositivos de voz

Descargue el [SDK de dispositivos de voz](#).

## Pasos siguientes

[Introducción al SDK de dispositivos de voz](#)

# Recomendaciones de matriz de micrófonos del SDK de dispositivos de voz

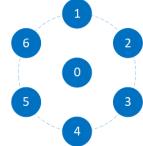
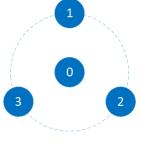
13/01/2020 • 9 minutes to read • [Edit Online](#)

En este artículo, aprenderá a diseñar una matriz de micrófonos para el SDK de dispositivos de voz.

El SDK de dispositivos de voz funciona mejor con una matriz de micrófonos que se ha diseñado según las directrices siguientes, incluida la selección de geometría y componentes del micrófono. También se proporcionan instrucciones sobre los aspectos de integración y eléctricos que se deben tener en cuenta.

## Geometría del micrófono

Se recomiendan las geometrías de matriz siguientes para su uso con la pila de audio de Microsoft. La ubicación de las fuentes de sonido y el rechazo del ruido ambiental se han mejorado con un número mayor de micrófonos con dependencias de aplicaciones y escenarios de usuario específicos, y el factor de forma del dispositivo.

	MATRIZ CIRCULAR	MATRIZ LINEAL	
# Micrófonos			
Geometría	6 externo, 1 centro, Radio = 42,5 mm, Espaciado uniforme	3 exterior, 1 centro, Radio = 42,5 mm, Espaciado uniforme	Longitud = 120 mm, Espaciado = 40 mm

Los canales de micrófono deben ordenarse según la numeración representada para cada matriz y subir desde 0. La pila de audio de Microsoft requerirá un flujo de referencia adicional de reproducción de audio para realizar la cancelación del eco.

## Selección de componentes

Los componentes de micrófono deben seleccionarse para reproducir con precisión una señal libre de ruido y distorsión.

Las propiedades recomendadas al seleccionar micrófonos son:

PARÁMETRO	RECOMENDADO
SNR	>= 65 dB (señal de 1 kHz a 94 dB SPL, ruido ponderado con A)
Coincidencia de amplitud	± 1 dB a 1 kHz
Coincidencia de fase	± 2° a 1 kHz

PARÁMETRO	RECOMENDADO
Punto de sobrecarga acústico (AOP)	>= 120 dB SPL (THD = 10 %)
Velocidad de bits	24 bits como mínimo
Frecuencia de muestreo	Mínimo 16 kHz*
Respuesta de frecuencia	+ 3 dB, máscara flotante 8000-200 Hz*
Confiabilidad	Intervalo de temperatura de almacenamiento: entre -40° C y 70° C Intervalo de temperatura de funcionamiento: entre -20° C y 55° C

\*Pueden ser necesarias velocidades de muestreo más altas o "más amplias" en aplicaciones de comunicaciones de alta calidad (VoIP)

Una buena selección de componentes debe ir acompañada de una buena integración electroacústica para evitar reducir el rendimiento de los componentes usados. Los casos de uso único también pueden necesitar requisitos adicionales (por ejemplo, intervalos de temperatura de funcionamiento).

## Integración de la matriz de micrófonos

El rendimiento de la matriz de micrófono cuando se integra en un dispositivo diferirá de la especificación del componente. Es importante asegurarse de que los micrófonos están bien adaptados después de la integración. Por tanto, el rendimiento del dispositivo medido después de cualquier ganancia fija o EQ debe cumplir las siguientes recomendaciones:

PARÁMETRO	RECOMENDADO
SNR	> 63 dB (señal de 1 kHz a 94 dB SPL, ruido ponderado con A)
Sensibilidad de salida	-26 dBFS/Pa a 1 kHz (recomendado)
Coincidencia de amplitud	± 2 dB, 200-8000 Hz
THD %*	≤ 1 %, 200-8000 Hz, 94 dB SPL, quinto orden
Respuesta de frecuencia	± 6 dB, máscara flotante a 200-8000 Hz**

\*\*Se requiere un altavoz de baja distorsión para medir THD (por ejemplo, Neumann KH120) .

\*\*Pueden ser necesarios intervalos de frecuencia "más amplios" para las aplicaciones de comunicaciones de alta calidad (VoIP) .

## Recomendaciones de integración del altavoz

Como la cancelación del eco es necesaria para los dispositivos de reconocimiento de voz que contienen altavoces, se proporcionan recomendaciones adicionales para la integración y la selección de altavoces.

PARÁMETRO	RECOMENDADO

PARÁMETRO	RECOMENDADO
Consideraciones sobre la linealidad	Sin procesamiento no lineal después de la referencia del altavoz, de lo contrario se requiere una secuencia de referencia de bucle invertido basada en hardware
Bucle invertido de altavoz	Se proporciona mediante WASAPI, API privadas, complemento ALSA personalizado (Linux), o se proporciona mediante el canal de firmware
THD %	Bandas de tercios de octava, mínimo quinto orden, reproducción 70 dBA a 0,8 m ≤ 6,3 %, 315-500 Hz ≤ 5 %, 630-5000 Hz
Acoplamiento de eco a los micrófonos	> TCLw -10 dB mediante el método ITU-T G.122 anexo B.4, normalizado a nivel de micrófono TCLw = TCLwmeasured + (nivel medido - sensibilidad de salida objetivo) TCLw = TCLwmeasured + (nivel medido - (-26))

## Arquitectura de diseño de integración

Las siguientes directrices sobre arquitectura son necesarias al integrar micrófonos en un dispositivo:

PARÁMETRO	RECOMENDACIÓN
Similitud del puerto de micrófono	Todos los puertos de micrófono tienen la misma longitud en la matriz
Dimensiones del puerto de micrófono	Tamaño de puerto Ø0,8-1,0 mm. Longitud de puerto/diámetro de puerto < 2
Sellado de micrófono	Juntas de sellado implementadas uniformemente en la pila. Se recomienda una relación de compresión del > 70 % para juntas de espuma.
Confiabilidad del micrófono	Se debe usar una malla para evitar el polvo y el acceso (entre el PCB de los micrófonos de puerto inferior y la junta de sellado/cubierta superior).
Aislamiento del micrófono	Juntas de goma y desacoplamiento de la vibración a través de la estructura, en especial para aislar las vías de vibración debido a los altavoces integrados.
Reloj de muestreo	El audio del dispositivo debe estar libre de vibración y cortes con desviación baja.
Funcionalidad de registro	El dispositivo debe poder grabar secuencias sin procesar de canales individuales de manera simultánea.
USB	Todos los dispositivos de entrada de audio USB deben establecer descriptores de acuerdo con la <a href="#">especificación Rev3 de dispositivos de audio USB</a> .
Geometría del micrófono	Los controladores deben implementar correctamente <a href="#">descriptores de geometría de matriz de micrófonos</a> .

PARÁMETRO	RECOMENDACIÓN
Detectabilidad	Los dispositivos no deben tener algoritmos de procesamiento de audio no lineales basados en hardware, firmware o software de terceros indetectables o incontrolables, ni hacia ni desde el dispositivo.
Formato de captura	Los formatos de captura deben usar una frecuencia de muestreo mínima de 16 kHz y una profundidad recomendada de 24 bits.

## Consideraciones sobre la arquitectura eléctrica

Si procede, las matrices pueden estar conectadas a un host USB (por ejemplo, un SoC que ejecuta la pila de audio de Microsoft) y las interfaces a servicios de voz u otras aplicaciones.

Los componentes de hardware, como la conversión de PDM a TDM, deben garantizar que se conserva el rango dinámico y el SNR de los micrófonos dentro de los remuestreadores.

Se debe admitir audio USB 2.0 de alta velocidad dentro de cualquier MCU de audio con el fin de proporcionar el ancho de banda necesario para hasta siete canales a frecuencias de muestreo y profundidades de bits mayores.

## Pasos siguientes

[Conozca el SDK de dispositivos de voz.](#)

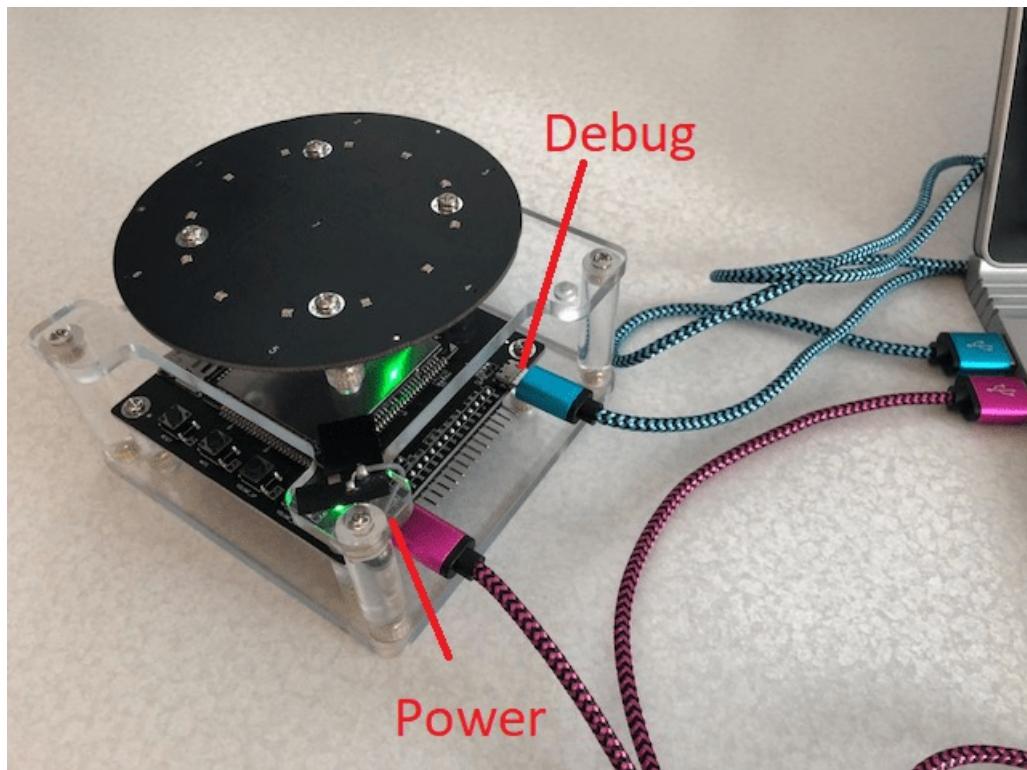
# Dispositivo: Roobo Smart Audio Dev Kit

13/01/2020 • 4 minutes to read • [Edit Online](#)

En este artículo se proporciona información específica de dispositivo para Roobo Smart Audio Dev Kit.

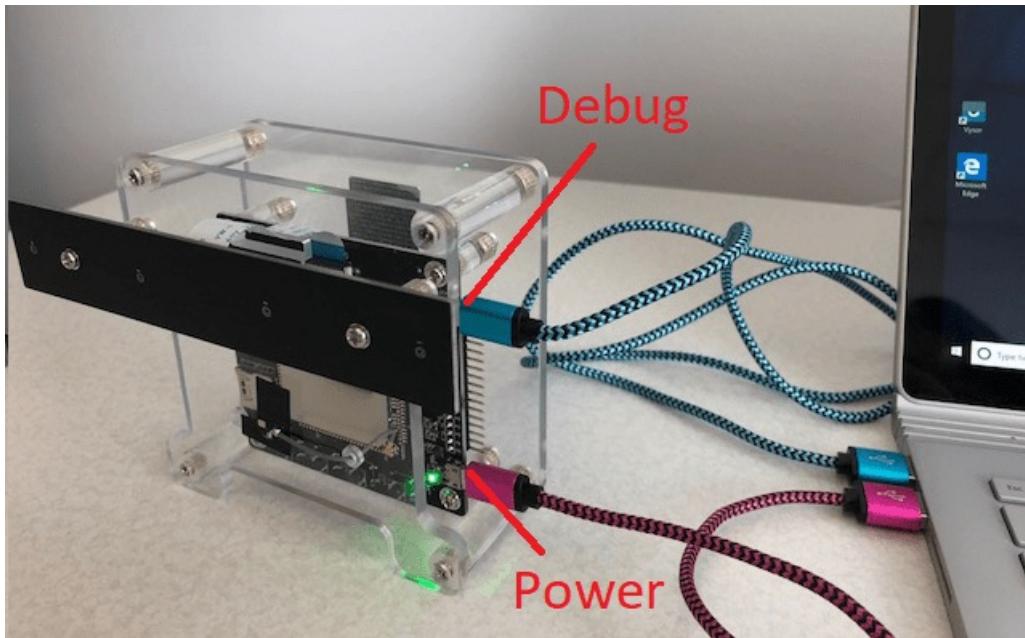
## Configuración del kit de desarrollo

1. El kit de desarrollo tiene dos conectores micro USB. El conector de la izquierda sirve para conectar el kit de desarrollo y aparece resaltado como Power (Conectar) en la siguiente imagen. El de la derecha sirve para controlarlo y aparece marcado como Debug (Depurar) en la imagen.



2. Conecte el kit de desarrollo mediante un cable micro USB para conectar el puerto de alimentación a un equipo o adaptador de corriente. Se iluminará un indicador de encendido verde en el panel superior.
3. Para controlar el kit de desarrollo, conecte el puerto de depuración a un equipo mediante un segundo cable micro USB. Es fundamental usar un cable de alta calidad para garantizar unas comunicaciones de confianza.
4. Oriente el kit de desarrollo para realizar la configuración circular o lineal.

CONFIGURACIÓN DEL KIT DE DESARROLLO	ORIENTACIÓN
Circular	Vertical, con los micrófonos orientados hacia el techo
Lineal	En el lateral, con los micrófonos mirando hacia usted (tal como se muestra en la siguiente imagen).



5. Instale los certificados y establezca los permisos del dispositivo de sonido. Escriba los siguientes comandos en una ventana del símbolo del sistema:

```
adb push C:\SDSDK\Android-Sample-Release\scripts\roobo_setup.sh /data/  
adb shell  
cd /data/  
chmod 777 roobo_setup.sh  
../roobo_setup.sh  
exit
```

#### NOTE

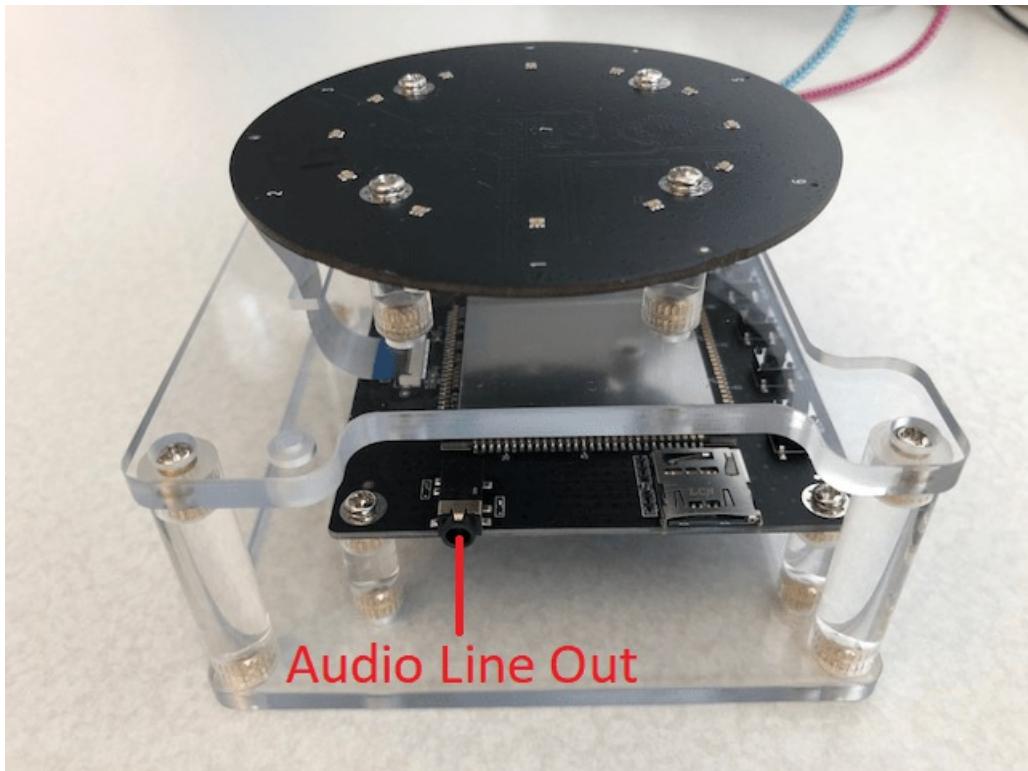
Estos comandos usan Android Debug Bridge, `adb.exe`, que forma parte de la instalación de Android Studio. Esta herramienta se encuentra en `C:\Users[nombre de usuario]\AppData\Local\Android\Sdk\platform-tools`. Puede agregar este directorio a la ruta de acceso para que sea más práctico invocar `adb`. En caso contrario, debe especificar la ruta de acceso completa a la instalación de `adb.exe` en todos los comandos que invoca `adb`.

Si ve un error `no devices/emulators found`, compruebe que el cable USB está conectado y que es un cable de alta calidad. Puede usar `adb devices` para comprobar que el equipo puede comunicarse con el kit de desarrollo ya que este devolverá una lista de dispositivos.

#### TIP

Silencie el micrófono y el altavoz del equipo para asegurarse de que está trabajando con los micrófonos del kit de desarrollo. De esta manera, no activará accidentalmente el dispositivo con el audio del equipo.

6. Si quiere asociar un altavoz al kit de desarrollo, puede conectarlo a la salida de línea de audio. Debe elegir un altavoz de buena calidad con un enchufe analógico 3,5 mm.



## Información de desarrollo

Para más información sobre desarrollo, consulte la [guía de desarrollo de Roobo](#).

## Audio

Roobo proporciona una herramienta que captura todo el audio en una memoria Flash. Esto puede serle de utilidad para ayudarle a solucionar problemas de audio. Se proporciona una versión de la herramienta para cada configuración del kit de desarrollo. Elija el dispositivo en el [sitio de Roobo](#) y, a continuación, haga clic en el vínculo **Roobo Tools**(Herramientas de Roobo) en la parte inferior de la página.

## Pasos siguientes

- [Ejecute la aplicación de ejemplo de Android](#).

# Inicio rápido: Ejecución de la aplicación de ejemplo de Speech Devices SDK en Windows

13/01/2020 • 9 minutes to read • [Edit Online](#)

En este inicio rápido, aprenderá a usar Speech Devices SDK para Windows para crear un producto habilitado para voz o para utilizarlo como un dispositivo de [transcripción de conversaciones](#). Para la transcripción de conversaciones, solo se admite el [Azure Kinect DK](#). En el caso de otros tipos de voz, se admiten matrices de micrófonos lineales que proporcionen una geometría de matriz de micrófonos.

La aplicación se crea con el paquete del SDK de Voz y el entorno de desarrollo integrado de Java de Eclipse (v4) en Windows de 64 bits. Se ejecuta en un entorno de tiempo de ejecución de Java 8 (JRE) de 64 bits.

Para esta guía se requiere una cuenta de [Azure Cognitive Services](#) con un recurso del servicio de voz. Si no tiene una cuenta, puede usar la [evaluación gratuita](#) para obtener una clave de suscripción.

El código fuente de la [aplicación de ejemplo](#) se incluye con Speech Devices SDK. También está [disponible en GitHub](#).

## Requisitos previos

Esta guía de inicio rápido requiere:

- Sistema operativo: Windows de 64 bits
- Una matriz de micrófonos, como [Azure Kinect DK](#)
- [IDE de Java de Eclipse](#)
- Solo [Java 8 o JDK 8](#).
- [Microsoft Visual C++ Redistributable](#)
- Una clave de suscripción de Azure para el servicio Voz. [Obtenga una gratis](#).
- Descargue la versión más reciente de [Speech Devices SDK](#) para Java y extraiga el archivo .zip en el directorio de trabajo.

### NOTE

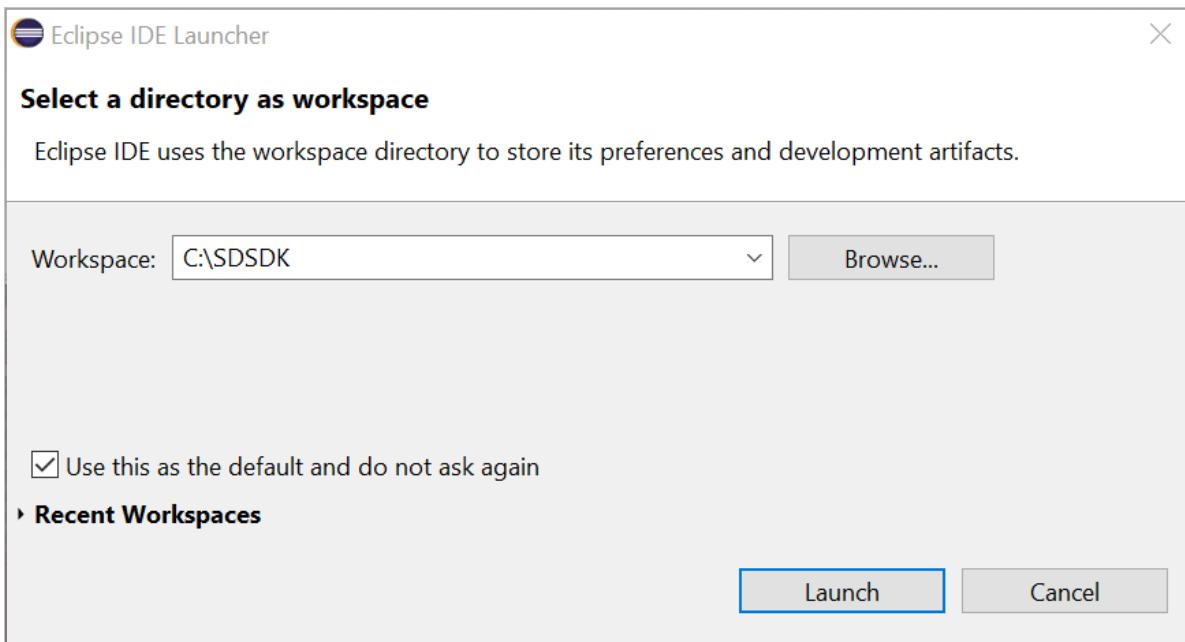
El archivo JRE-Sample-Release.zip incluye la aplicación de ejemplo de JRE y en este inicio rápido se da por supuesto que ha extraído la aplicación en C:\SDSDK\JRE-Sample-Release.

Transcripción de conversaciones solo está disponible actualmente para "en-US" y "zh-CN", en las regiones "centralus" y "eastasia". Debe tener una clave de voz en una de esas regiones para usar Transcripción de conversaciones.

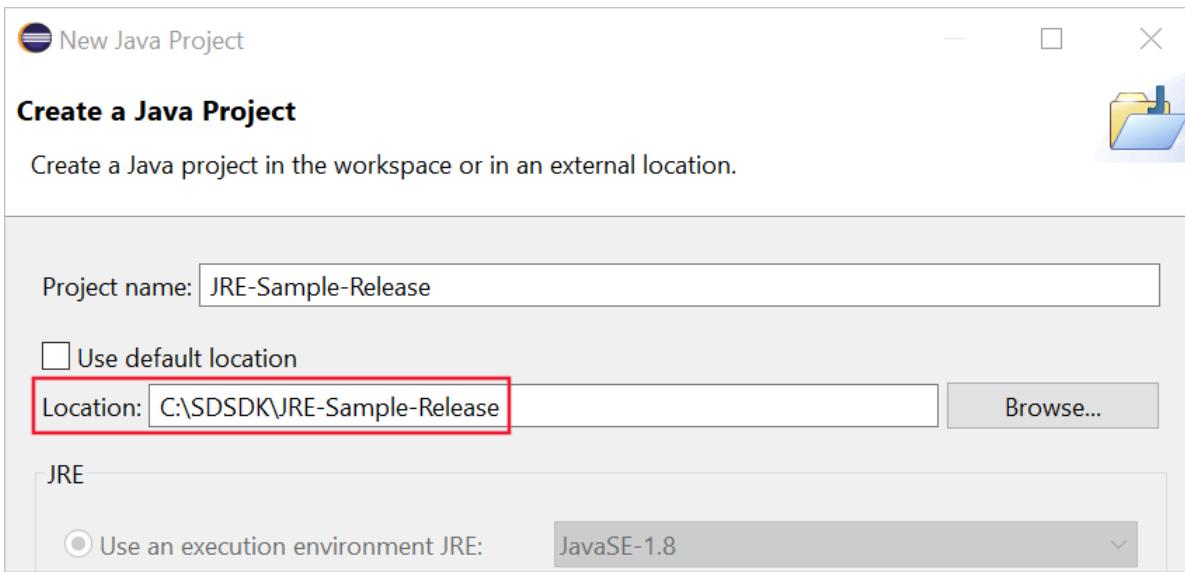
Si tiene previsto usar intenciones necesitará una suscripción a [Language Understanding Service \(LUIS\)](#). Para más información acerca de LUIS y el reconocimiento de intenciones, consulte [Reconocimiento de las intenciones de voz con LUIS, C#](#). Hay un [modelo de ejemplo de LUIS](#) disponible para esta aplicación.

## Creación y configuración del proyecto

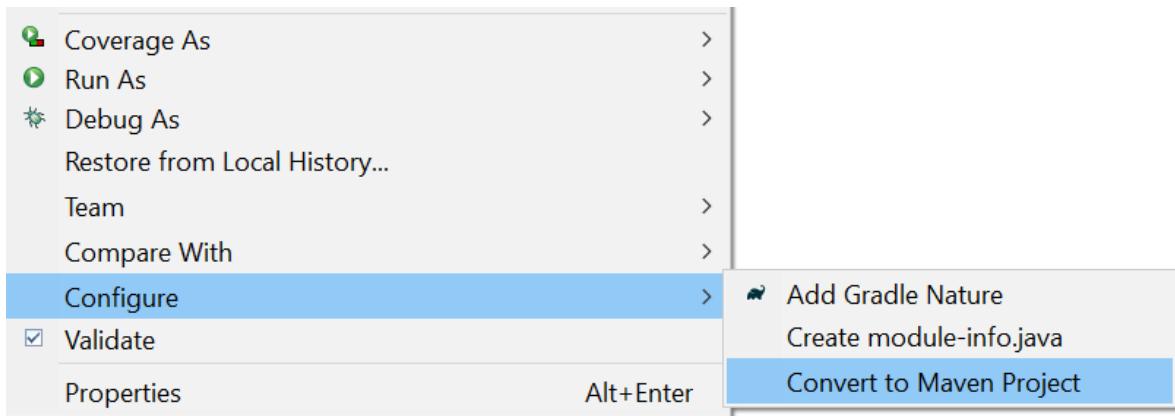
1. Inicie Eclipse.
2. En el **selector de IDE de Eclipse**, en el campo **Workspace** (Área de trabajo), escriba el nombre de un nuevo directorio de área de trabajo. Luego, seleccione **Launch** (Iniciar).



3. Al cabo de unos segundos, aparece la ventana principal del IDE de Eclipse. Cierre la pantalla de bienvenida si hay alguna.
4. En la barra de menús de Eclipse, cree un nuevo proyecto eligiendo **File (Archivo) > New (Nuevo) > Java Project (Proyecto de Java)**. Si no está disponible seleccione **Project** (Proyecto) y, a continuación, **Java Project** (Proyecto de Java).
5. Se inicia el asistente para **nuevo proyecto de Java. Busque** la ubicación del proyecto de ejemplo. Seleccione **Finalizar**.



6. En el **Explorador de paquetes**, haga clic con el botón derecho en el proyecto. Elija **Configure (Configurar) > Convert to Maven Project (Convertir en proyecto de Maven)** en el menú contextual. Seleccione **Finalizar**.



## 7. Abra el archivo pom.xml y edítelo.

Al final del archivo, antes de la etiqueta de cierre `</project>`, cree los elementos `repositories` y `dependencies`, como se muestra aquí, y asegúrese de que `version` coincida con la versión actual:

```
<repositories>
    <repository>
        <id>maven-cognitiveservices-speech</id>
        <name>Microsoft Cognitive Services Speech Maven Repository</name>
        <url>https://csspeechstorage.blob.core.windows.net/maven/</url>
    </repository>
</repositories>

<dependencies>
    <dependency>
        <groupId>com.microsoft.cognitiveservices.speech</groupId>
        <artifactId>client-sdk</artifactId>
        <version>1.7.0</version>
    </dependency>
</dependencies>
```

## 8. Copie el contenido de **Windows-x64** en la ubicación del proyecto de Java, por ejemplo **C:\SDSDK\JRE-Sample-Release**.

## 9. Copie `kws.table`, `participants.properties` y `Microsoft.CognitiveServices.Speech.extension.pma.dll` en la carpeta del proyecto **target\classes**

## Configurar la aplicación de ejemplo

### 1. Agregue la clave de suscripción de Voz al código fuente. Si quiere probar el reconocimiento de intenciones, agregue también la clave de suscripción y el identificador de aplicación del servicio [Language Understanding](#).

Para Voz y LUIS, la información se trasladará a `FunctionsList.java`:

```
// Subscription
private static String SpeechSubscriptionKey = "<enter your subscription info here>";
private static String SpeechRegion = "westus"; // You can change this if your speech region is different.
private static String LuisSubscriptionKey = "<enter your subscription info here>";
private static String LuisRegion = "westus2"; // you can change this, if you want to test the intent, and your LUIS region is different.
private static String LuisAppId = "<enter your LUIS AppId>";
```

Si usa la transcripción de conversaciones, la información de la clave y la región de Voz también se necesitará en `Cts.java`:

```
private static final String CTSKey = "<Conversation Transcription Service Key>";  
private static final String CTSRegion="<Conversation Transcription Service Region>";// Region may be  
"centralus" or "eastasia"
```

2. La palabra clave predeterminada (palabra clave) es "Equipo". Además, puede probar una de las otras palabras clave proporcionadas, como "Máquina" o "Asistente". Los archivos de recursos de estas palabras clave alternativas pueden encontrarse en Speech Devices SDK en la carpeta de palabras clave. Por ejemplo, `C:\SDSDK\JRE-Sample-Release\keyword\Computer` contiene los archivos utilizados para la palabra clave "Equipo".

**TIP**

También puede [crear una palabra clave personalizada](#).

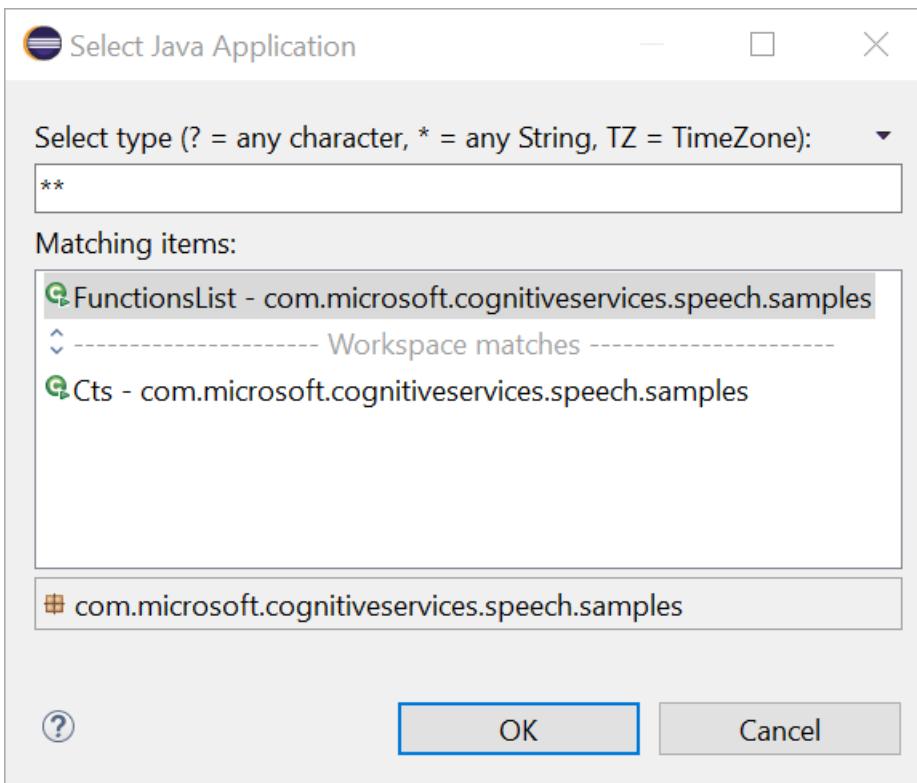
Para usar una nueva palabra clave, actualice la línea siguiente en `FunctionsList.java`, y copie la palabra clave en la aplicación. Por ejemplo, para usar la palabra clave "Máquina" desde el paquete de palabras clave `machine.zip`:

- Copie el archivo `kws.table` del paquete ZIP en la carpeta de proyecto **target/classes**.
- Actualice `FunctionsList.java` con el nombre de la palabra clave:

```
private static final String Keyword = "Machine";
```

## Ejecución de la aplicación de ejemplo de Eclipse

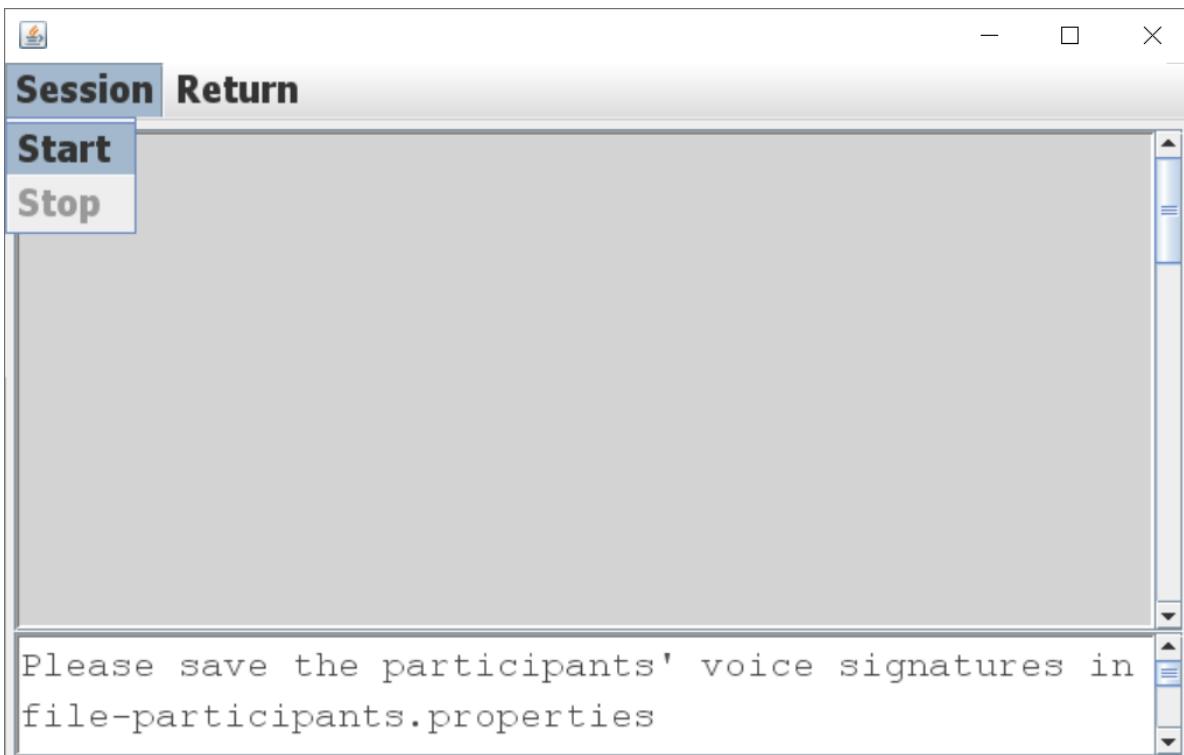
1. En la barra de menús de Eclipse seleccione, **Run (Ejecutar) > Run As (Ejecutar como) > Java Application (Aplicación de Java)**. A continuación, seleccione **FunctionsList** y **OK** (Aceptar).



2. Se inicia la aplicación de ejemplo del SDK de dispositivos de voz y muestra las siguientes opciones:

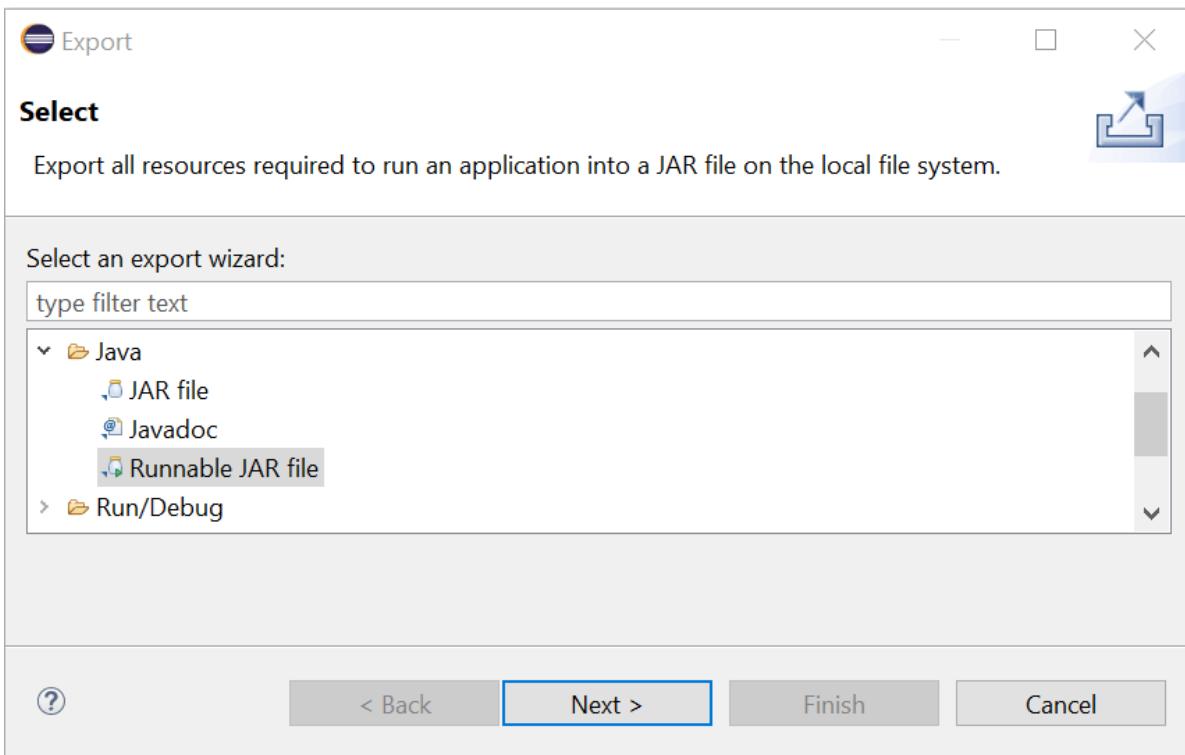


3. Pruebe la nueva demostración de **Transcripción de conversaciones**. Empiece a transcribir con **Sesión > Iniciar**. De forma predeterminada, todos los usuarios son invitados. Sin embargo, si dispone de las firmas de voz del participante, estas se pueden colocar en un archivo `participants.properties` en la carpeta del proyecto **target/classes**. Para generar la firma de voz, consulte [Transcripción de conversaciones \(SDK\)](#).

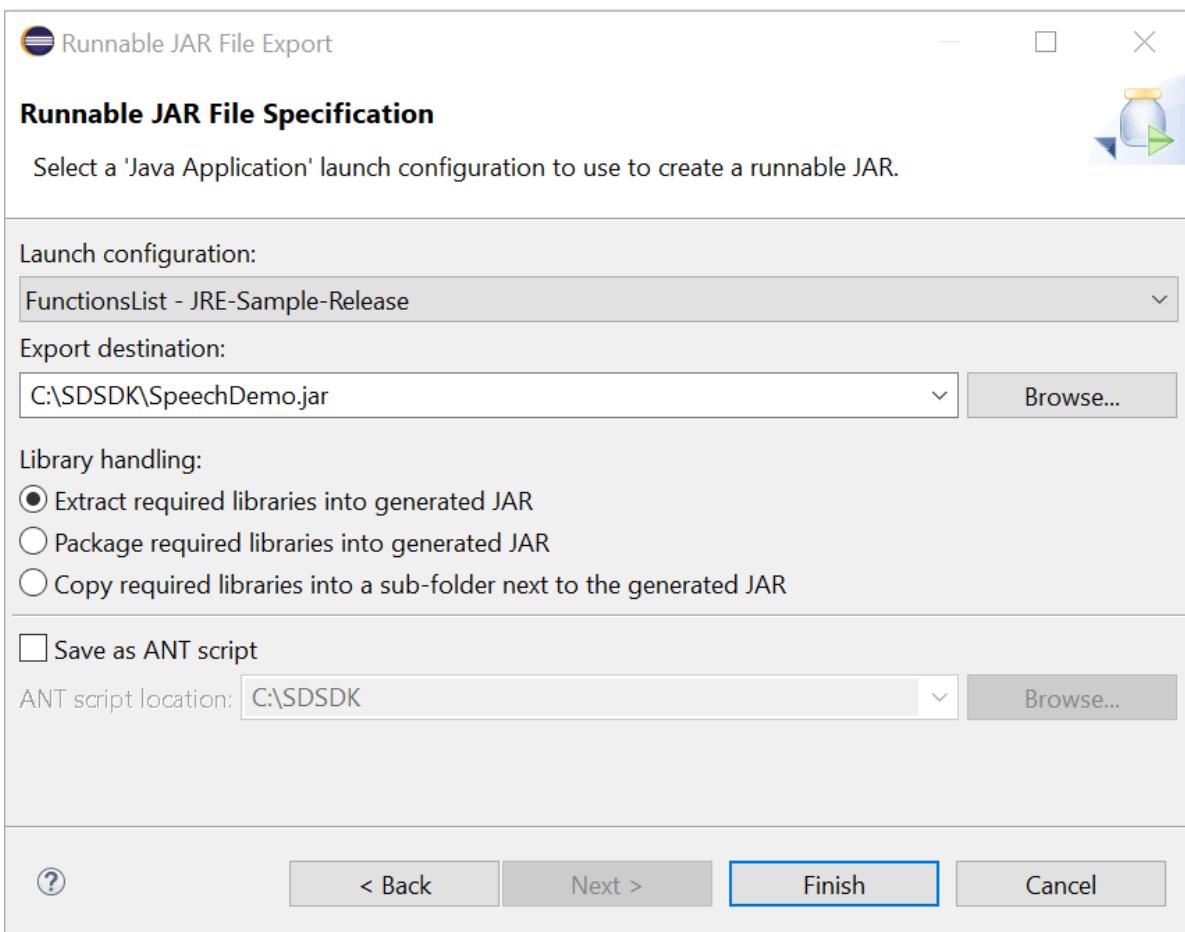


## Creación y ejecución de una aplicación independiente

1. En el **Explorador de paquetes**, haga clic con el botón derecho en el proyecto. Seleccione **Exportar**.
2. Aparecerá la ventana **Exportar**. Expanda **Java** y seleccione **Runnable JAR file** (Archivo JAR ejecutable) y, a continuación, seleccione **Siguiente**.



3. Aparece la ventana **Runnable JAR File Export** (Exportación de archivo JAR ejecutable). Elija un **Destino de exportación** para la aplicación y, a continuación, seleccione **Finalizar**.



4. Coloque `kws.table`, `participants.properties`, `unimic_runtime.dll`, `pma.dll` y `Microsoft.CognitiveServices.Speech.extension.pma.dll` en la carpeta de destino elegida anteriormente, ya que la aplicación necesita estos archivos.
5. Ejecución de la aplicación independiente

```
java -jar SpeechDemo.jar
```

## Pasos siguientes

[Revise las notas de la versión.](#)

# Inicio rápido: Ejecución de la aplicación de ejemplo de Speech Devices SDK en Linux

13/01/2020 • 10 minutes to read • [Edit Online](#)

En este inicio rápido, aprenderá a usar Speech Devices SDK para Linux para crear un producto habilitado para voz o para utilizarlo como un dispositivo de [transcripción de conversaciones](#). Actualmente, solo se admite [Azure Kinect DK](#).

La aplicación se crea con el paquete de Speech SDK y el IDE de Java de Eclipse (v4) en Linux de 64 bits (Ubuntu 16.04, Ubuntu 18.04, Debian 9). Se ejecuta en un entorno de tiempo de ejecución de Java 8 (JRE) de 64 bits.

Para esta guía se requiere una cuenta de [Azure Cognitive Services](#) con un recurso del servicio de voz. Si no tiene una cuenta, puede usar la [evaluación gratuita](#) para obtener una clave de suscripción.

El código fuente de la [aplicación de ejemplo](#) se incluye con Speech Devices SDK. También está [disponible en GitHub](#).

## Requisitos previos

Esta guía de inicio rápido requiere:

- Sistema operativo: Linux de 64 bits (Ubuntu 16.04, Ubuntu 18.04, Debian 9)
- [Azure Kinect DK](#)
- [IDE de Java de Eclipse](#)
- Solo [Java 8](#) o [JDK 8](#).
- Una clave de suscripción de Azure para el servicio Voz. [Obtenga una gratis](#).
- Descargue la versión más reciente de [Speech Devices SDK](#) para Java y extraiga el archivo .zip en el directorio de trabajo.

### NOTE

El archivo JRE-Sample-Release.zip incluye la aplicación de ejemplo de JRE y en este inicio rápido se da por supuesto que ha extraído la aplicación en /home/wcaltest/JRE-Sample-Release.

Asegúrese de que estén instaladas estas dependencias antes de iniciar Eclipse.

- En Ubuntu:

```
sudo apt-get update  
sudo apt-get install libssl1.0.0 libasound2
```

- En Debian 9:

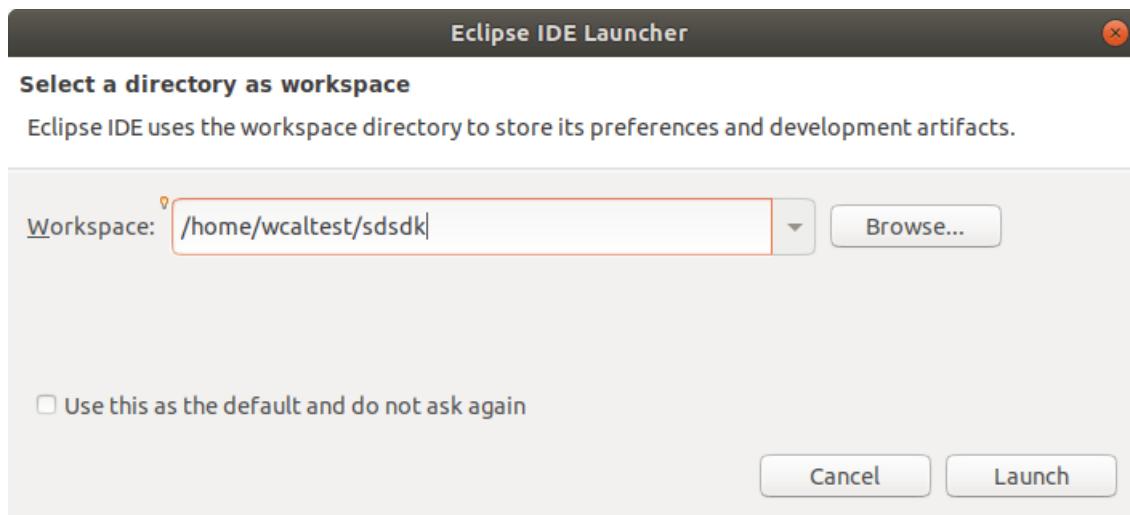
```
sudo apt-get update  
sudo apt-get install libssl1.0.2 libasound2
```

Transcripción de conversaciones solo está disponible actualmente para "en-US" y "zh-CN", en las regiones "centralus" y "eastasia". Debe tener una clave de voz en una de esas regiones para usar Transcripción de conversaciones.

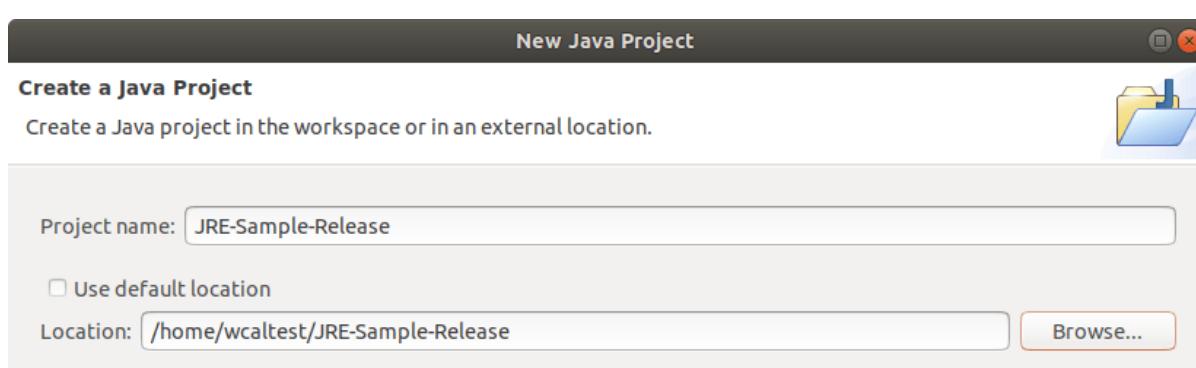
Si tiene previsto usar intenciones necesitará una suscripción a [Language Understanding Service \(LUIS\)](#). Para más información acerca de LUIS y el reconocimiento de intenciones, consulte [Reconocimiento de las intenciones de voz con LUIS, C#](#). Hay un [modelo de ejemplo de LUIS](#) disponible para esta aplicación.

## Creación y configuración del proyecto

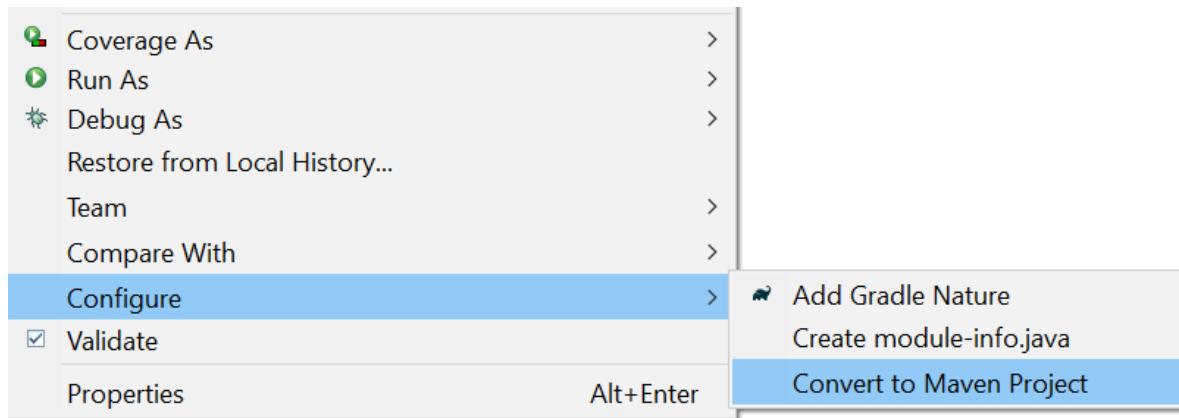
1. Inicie Eclipse.
2. En el **selector de IDE de Eclipse**, en el campo **Workspace** (Área de trabajo), escriba el nombre de un nuevo directorio de área de trabajo. Luego, seleccione **Launch** (Iniciar).



3. Al cabo de unos segundos, aparece la ventana principal del IDE de Eclipse. Cierre la pantalla de bienvenida si hay alguna.
4. En la barra de menús de Eclipse, cree un nuevo proyecto eligiendo **File (Archivo) > New (Nuevo) > Java Project (Proyecto de Java)**. Si no está disponible seleccione **Project** (Proyecto) y, a continuación, **Java Project** (Proyecto de Java).
5. Se inicia el asistente para **nuevo proyecto de Java**. Busque la ubicación del proyecto de ejemplo. Seleccione **Finalizar**.



6. En el **Explorador de paquetes**, haga clic con el botón derecho en el proyecto. Elija **Configure (Configurar) > Convert to Maven Project (Convertir en proyecto de Maven)** en el menú contextual. Seleccione **Finalizar**.



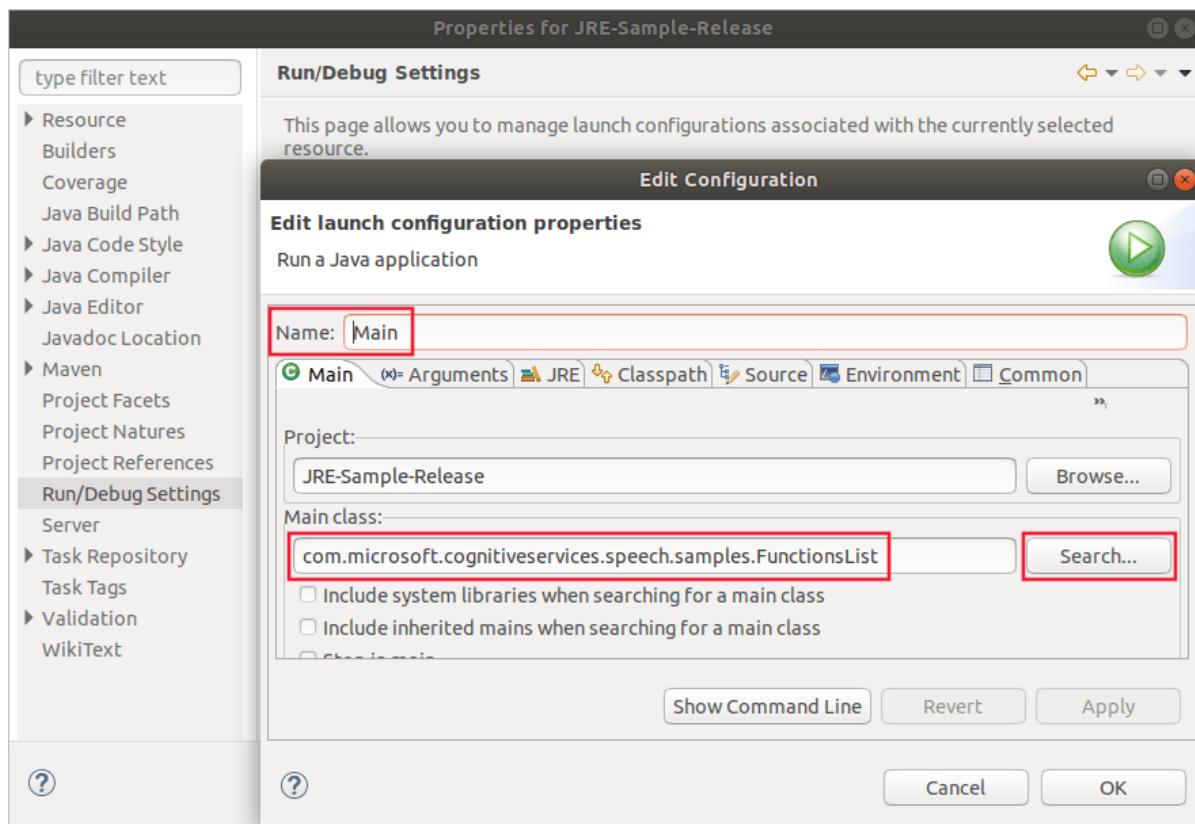
7. Abra el archivo pom.xml y edítelo.

Al final del archivo, antes de la etiqueta de cierre `</project>`, cree los elementos `repositories` y `dependencies`, como se muestra aquí, y asegúrese de que `version` coincida con la versión actual:

```
<repositories>
    <repository>
        <id>maven-cognitiveservices-speech</id>
        <name>Microsoft Cognitive Services Speech Maven Repository</name>
        <url>https://csspeechstorage.blob.core.windows.net/maven/</url>
    </repository>
</repositories>

<dependencies>
    <dependency>
        <groupId>com.microsoft.cognitiveservices.speech</groupId>
        <artifactId>client-sdk</artifactId>
        <version>1.7.0</version>
    </dependency>
</dependencies>
```

8. En el **Explorador de paquetes**, haga clic con el botón derecho en el proyecto. Elija **Propiedades** y, a continuación, **Configuración de depuración/ejecución** > **Nuevo...** > **Aplicación Java**.
9. Aparecerá la ventana **Edit configuration** (Editar configuración). En el campo **Nombre** escriba **Principal** y use **Búsqueda** para **Clase principal** para buscar y seleccionar **com.microsoft.cognitiveservices.speech.samples.FunctionsList**.



10. Copie los archivos binarios de audio de la arquitectura de destino, ya sea desde **Linux-arm** o desde **Linux-x64**, en la ubicación del proyecto de Java, por ejemplo: **/home/wcaltest/JRE-Sample-Release**
11. También en la ventana **Edit configuration** (Editar configuración), seleccione la página **Entorno** y **Nuevo**. Aparecerá la ventana **New Environment Variable** (Nueva variable de entorno). En el campo **Nombre** escriba **LD\_LIBRARY\_PATH** y en el campo **Valor** especifique la carpeta que contiene los archivos \*.so, por ejemplo **/home/wcaltest/JRE-Sample-Release**
12. Copie **kws.table** y **participants.properties** en la carpeta del proyecto **target\classes**

## Configurar la aplicación de ejemplo

1. Agregue la clave de suscripción de Voz al código fuente. Si quiere probar el reconocimiento de intenciones, agregue también la clave de suscripción y el identificador de aplicación del servicio [Language Understanding](#).

Para Voz y LUIS, la información se trasladará a **FunctionsList.java**:

```
// Subscription
private static String SpeechSubscriptionKey = "<enter your subscription info here>";
private static String SpeechRegion = "westus"; // You can change this if your speech region is
different.
private static String LuisSubscriptionKey = "<enter your subscription info here>";
private static String LuisRegion = "westus2"; // you can change this, if you want to test the intent,
and your LUIS region is different.
private static String LuisAppId = "<enter your LUIS AppId>";
```

Si usa la transcripción de conversaciones, la información de la clave y la región de Voz también se necesitará en **Cts.java**:

```
private static final String CTSKey = "<Conversation Transcription Service Key>";
private static final String CTSRegion=<Conversation Transcription Service Region>;// Region may be
"centralus" or "eastasia"
```

2. La palabra clave predeterminada (palabra clave) es "Equipo". Además, puede probar una de las otras palabras clave proporcionadas, como "Máquina" o "Asistente". Los archivos de recursos de estas palabras clave alternativas pueden encontrarse en Speech Devices SDK en la carpeta de palabras clave. Por ejemplo, `/home/wcaltest/JRE-Sample-Release/keyword/Computer` contiene los archivos utilizados para la palabra clave "Equipo".

**TIP**

También puede [crear una palabra clave personalizada](#).

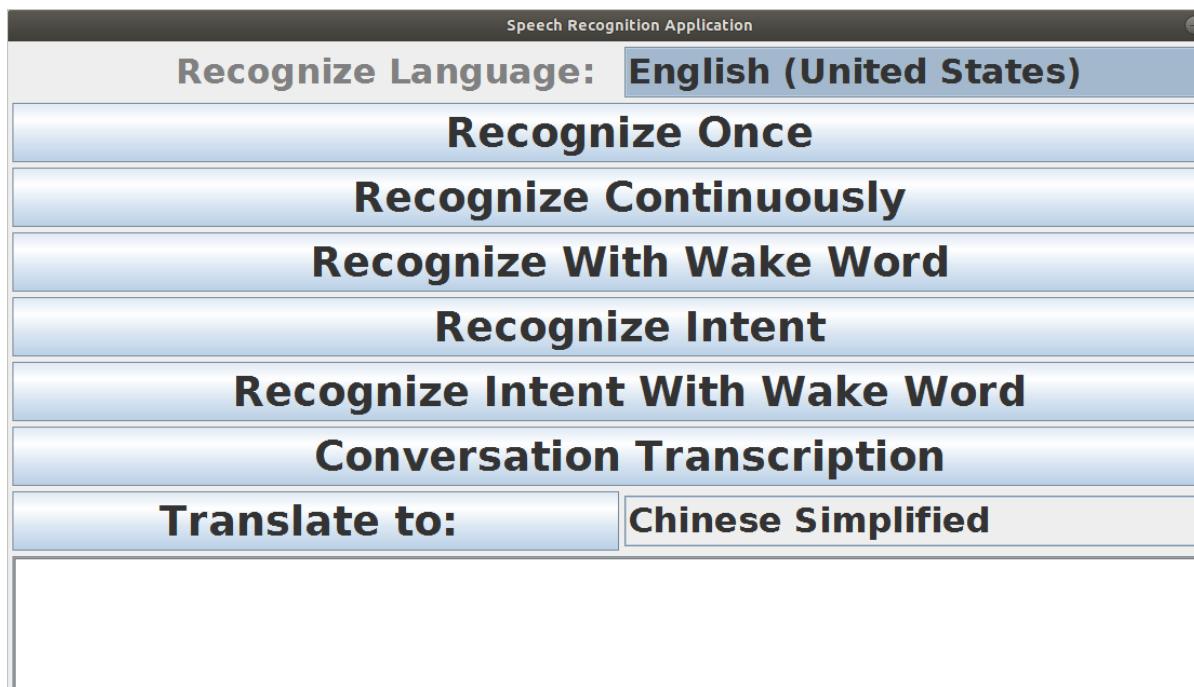
Para usar una nueva palabra clave, actualice la línea siguiente en `FunctionsList.java`, y copie la palabra clave en la aplicación. Por ejemplo, para usar la palabra clave "Máquina" desde el paquete de palabras clave `machine.zip`:

- Copie el archivo `kws.table` del paquete ZIP en la carpeta de proyecto **target/classes**.
- Actualice `FunctionsList.java` con el nombre de la palabra clave:

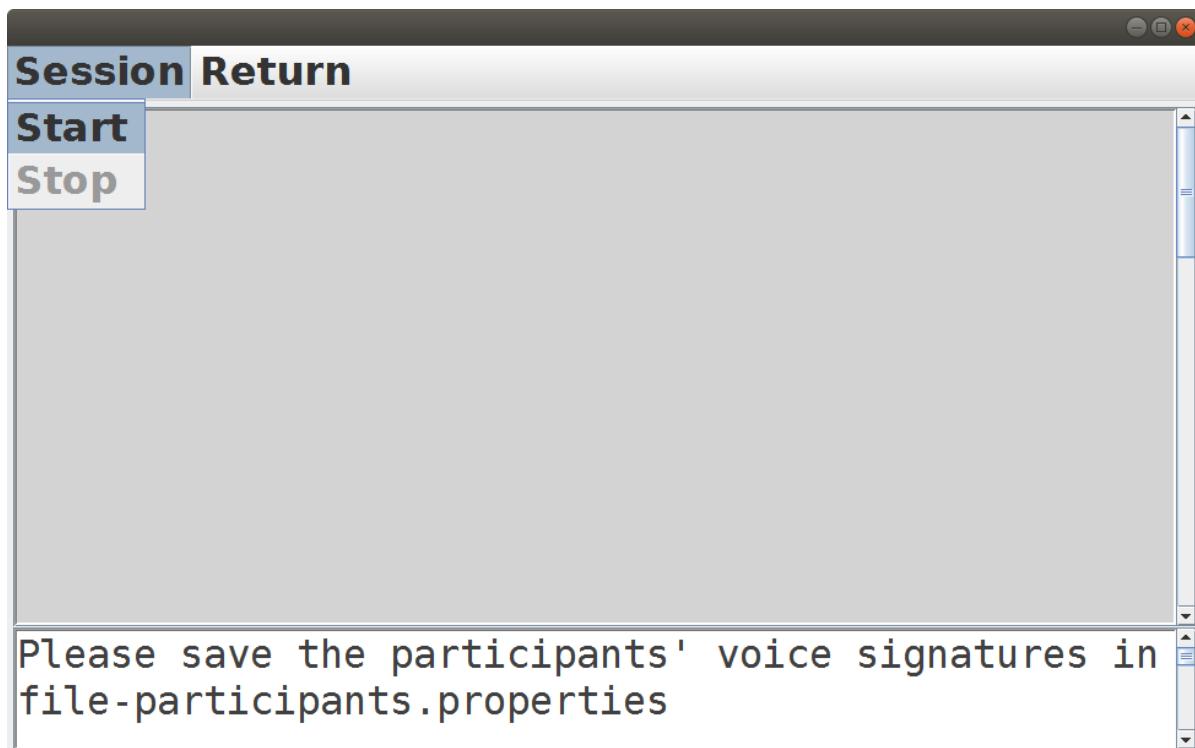
```
private static final String Keyword = "Machine";
```

## Ejecución de la aplicación de ejemplo de Eclipse

1. En la barra de menús de Eclipse seleccione, **Run (Ejecutar) > Run (Ejecutar como)** .
2. Se inicia la aplicación de ejemplo del SDK de dispositivos de voz y muestra las siguientes opciones:

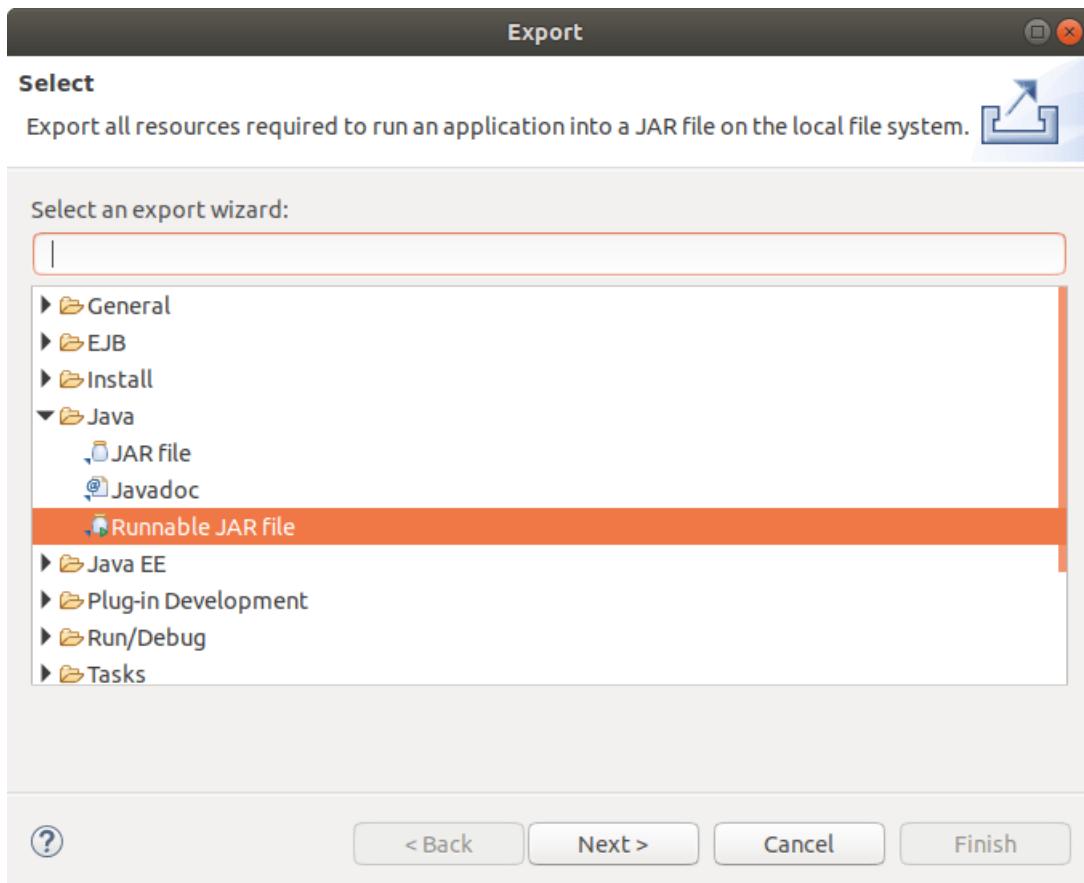


3. Pruebe la nueva demostración de **Transcripción de conversaciones**. Empiece a transcribir con **Sesión > Iniciar**. De forma predeterminada, todos los usuarios son invitados. Sin embargo, si dispone de las firmas de voz del participante, estas se pueden colocar en `participants.properties` en la carpeta del proyecto **target/classes**. Para generar la firma de voz, consulte [Transcripción de conversaciones \(SDK\)](#).

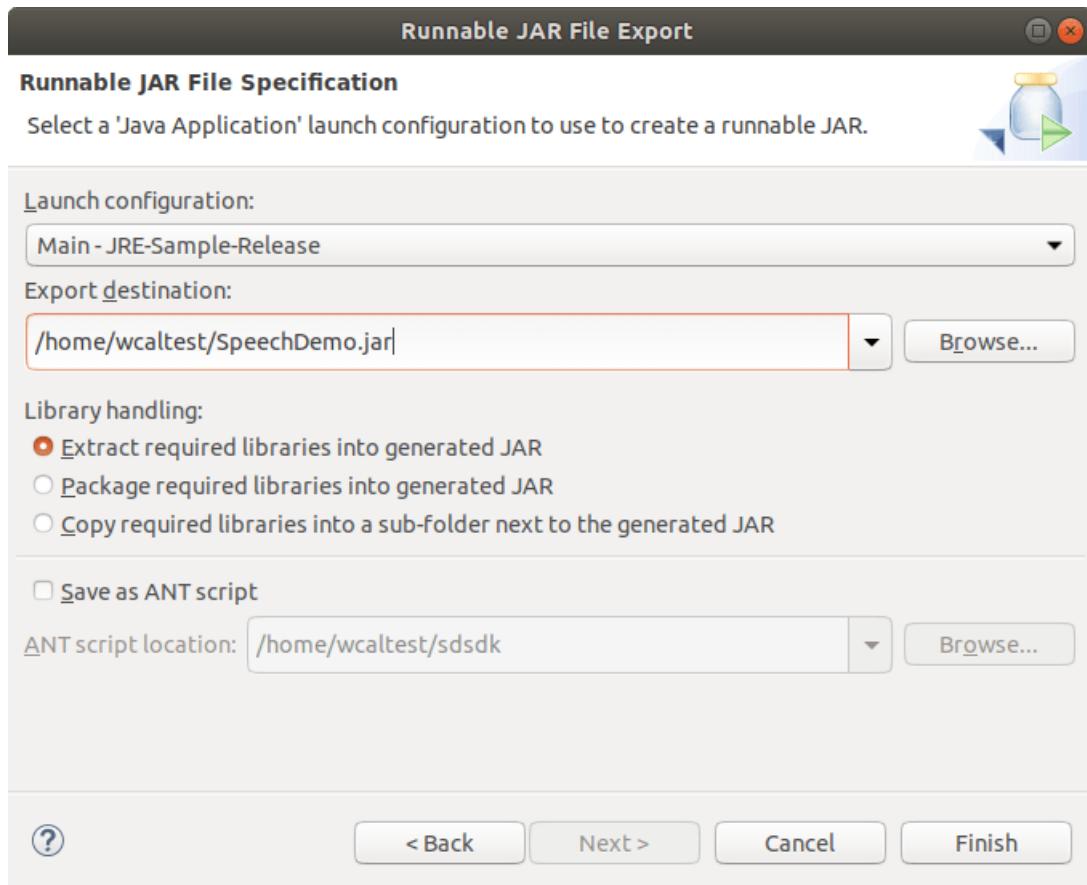


## Creación y ejecución de una aplicación independiente

1. En el **Explorador de paquetes**, haga clic con el botón derecho en el proyecto. Seleccione **Exportar**.
2. Aparecerá la ventana **Exportar**. Expanda **Java** y seleccione **Runnable JAR file** (Archivo JAR ejecutable) y, a continuación, seleccione **Siguiente**.



3. Aparece la ventana **Runnable JAR File Export** (Exportación de archivo JAR ejecutable). Elija un **Destino de exportación** para la aplicación y, a continuación, seleccione **Finalizar**.



4. Coloque `kws.table` y `participants.properties` en la carpeta de destino elegida anteriormente ya que estos archivos son necesarios para la aplicación.
5. Establezca `LD_LIBRARY_LIB` en la carpeta que contiene los archivos `*.so`

```
export LD_LIBRARY_PATH=/home/wcaltest/JRE-Sample-Release
```

6. Ejecución de la aplicación independiente

```
java -jar SpeechDemo.jar
```

## Pasos siguientes

Revise las notas de la versión.

# Inicio rápido: Ejecución de la aplicación de ejemplo de Speech Devices SDK en Android

13/01/2020 • 10 minutes to read • [Edit Online](#)

En este inicio rápido, aprenderá a usar Speech Devices SDK para Android para crear un producto habilitado para voz o para utilizarlo como un dispositivo de [transcripción de conversaciones](#).

Para esta guía se requiere una cuenta de [Azure Cognitive Services](#) con un recurso del servicio de voz. Si no tiene una cuenta, puede usar la [evaluación gratuita](#) para obtener una clave de suscripción.

El código fuente de la aplicación de ejemplo se incluye con el SDK de dispositivos de voz. También está [disponible en GitHub](#).

## Prerequisites

Para empezar a usar Speech Devices SDK, deberá:

- Seguir las instrucciones proporcionadas con su [kit de desarrollo](#) para encender el dispositivo.
- Descargar la versión más reciente de [Speech Devices SDK](#) y extraer el archivo .zip en el directorio de trabajo.

### NOTE

El archivo Android-Sample-Release.zip incluye la aplicación de ejemplo de Android y en este inicio rápido se da por supuesto que ha extraído la aplicación en C:\SDSDK\Android-Sample-Release.

- Para obtener una [clave de suscripción de Azure para el servicio de voz](#):
- Si tiene previsto usar la transcripción de conversaciones debe usar un [dispositivo de micrófono circular](#) y esta característica actualmente solo está disponible para los idiomas "en-US" y "zh-CN" en las regiones, "centralus" y "eastasia". Debe tener una clave de voz en una de esas regiones para usar Transcripción de conversaciones.
- Si tiene previsto usar el servicio de voz para identificar las intenciones (o acciones) a partir de las expresiones de los usuarios, necesitará una suscripción al [servicio Language Understanding \(LUIS\)](#). Para más información acerca de LUIS y el reconocimiento de intenciones, consulte [Reconocimiento de las intenciones de voz con LUIS, C#](#).

También puede [crear un modelo sencillo de LUIS](#) o usar el modelo LUIS de ejemplo, LUIS-example.json. El ejemplo de modelo LUIS está disponible en el [sitio de descarga del SDK de dispositivos de voz](#). Para cargar el archivo JSON del modelo en el [portal de LUIS](#), seleccione **Import new app** (Importar aplicación nueva) y elija el archivo JSON.

- Instale [Android Studio](#) y [Vysor](#) en su PC.

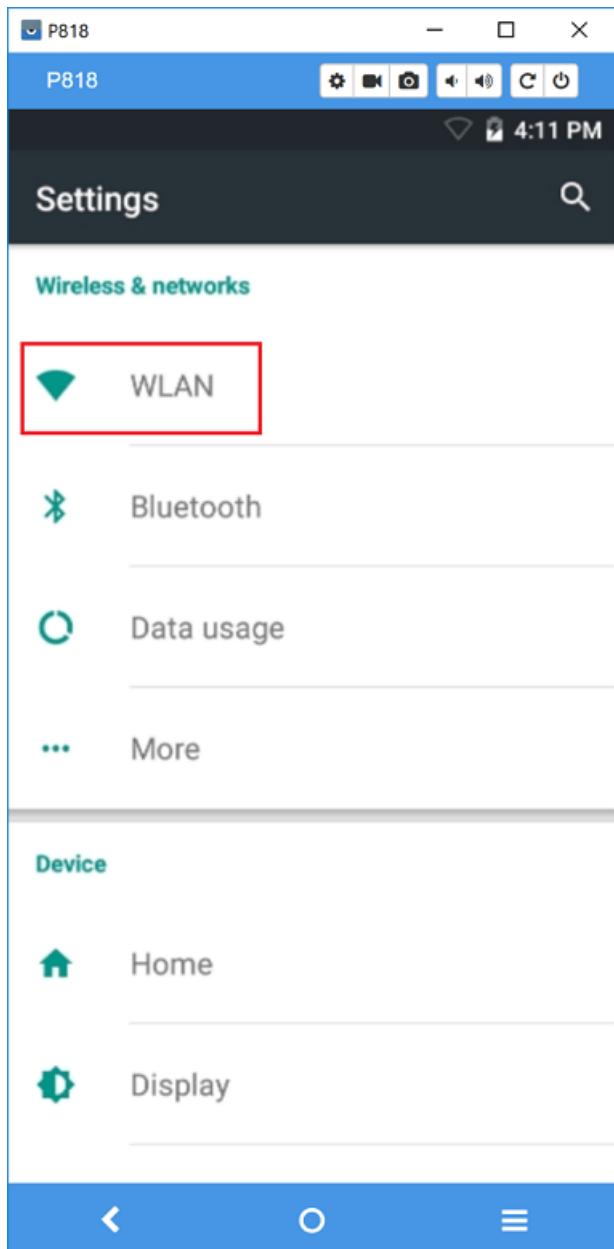
## Configuración del dispositivo

1. Inicie Vysor en el equipo.

The screenshot shows the Vysor application window. At the top, there's a blue header bar with the Vysor logo and standard window controls (minimize, maximize, close). Below the header is a light gray sidebar containing the word "Vysor". The main content area is divided into several sections:

- Choose a device:** A list box showing a single device entry: "P818" with "Serial: 4001000C0000026A". To the right of the list are four buttons: "View" (green), "Share" (white), "WLAN" (Wi-Fi icon), and "Settings" (gear icon).
- Settings:** A section titled "Settings" with a gear icon. It contains the following options:
  - International Keyboard:** An unchecked checkbox.
  - Share All Devices:** An unchecked checkbox.
  - Customize Vysor:** A link to "Manage Key Bindings and Buttons".
  - Start automatically:** A dropdown menu set to "Notification Prompt".
- Status:** A section titled "Status" containing the following information:
  - You've used Vysor for 3 hours.
  - An advertisement will be shown every 30 minutes while viewing an Android.
  - Purchase Vysor Pro to remove ads and unlock all features.
- Payment Options:** Icons for Credit Card, PayPal, and Google Play.
- Login:** A link to "Login for offline usage and Vysor Share".
- System Requirements:** Links for "Windows users need ADB Drivers.", "Using Android SDK ADB binary.", and "Vysor Version 1.9.0".
- Footer:** A navigation bar with icons for Home, Twitter, Developers, Support, Bug Report, Manual, Reload Vysor, and Reset Vysor.

2. El dispositivo debe figurar en la lista **Choose a device** (Seleccionar un dispositivo). Haga clic en el botón **View** (Vista) situado junto al dispositivo.
3. Para conectarse a la red inalámbrica, haga clic en el ícono de la carpeta y seleccione **Settings** (Configuración) > **WLAN**.



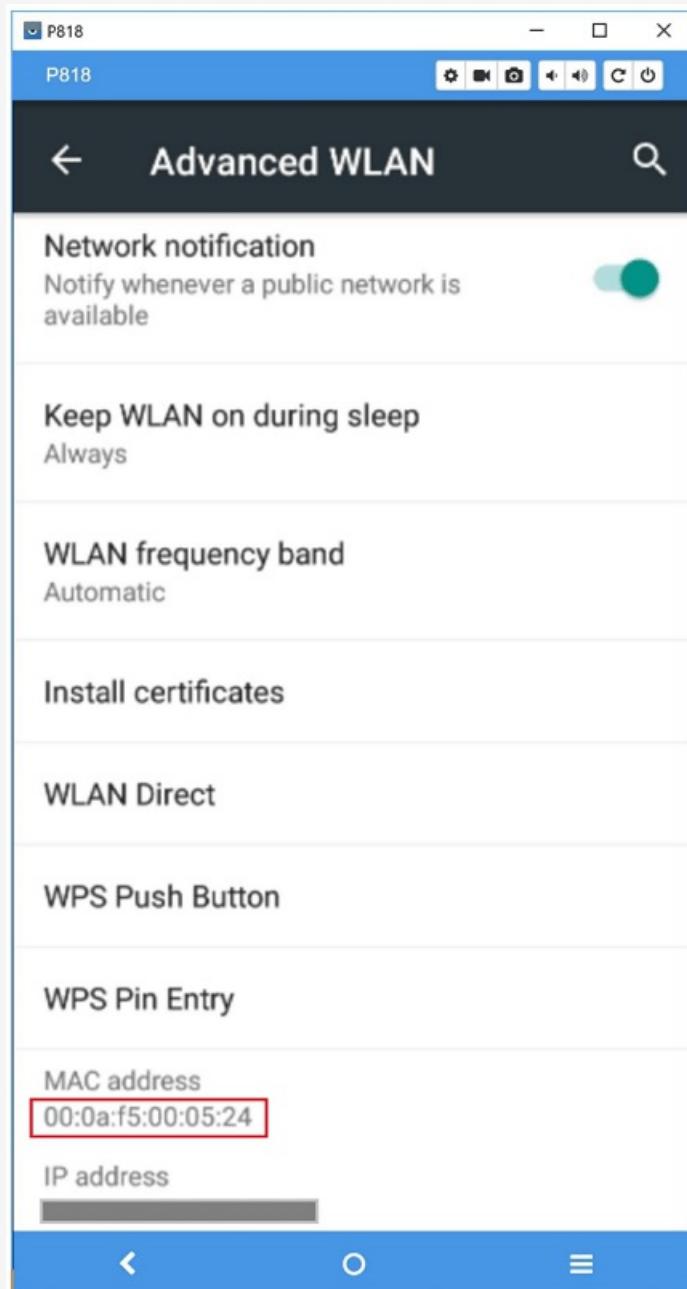
#### NOTE

Si su empresa tiene directivas respecto a la conexión de dispositivos al sistema Wi-Fi, deberá obtener la dirección MAC y ponerse en contacto con el departamento de TI para que le indiquen cómo conectarse al Wi-Fi de la empresa.

Para buscar la dirección MAC del kit de desarrollo, seleccione el ícono de la carpeta de archivos en el escritorio del kit de desarrollo.



Seleccione **Configuración**. Busque "mac address" (dirección MAC) y luego seleccione **Mac address** (Dirección MAC) > **Advanced WLAN** (WLAN avanzada). Anote la dirección MAC que aparece cerca de la parte inferior del cuadro de diálogo.

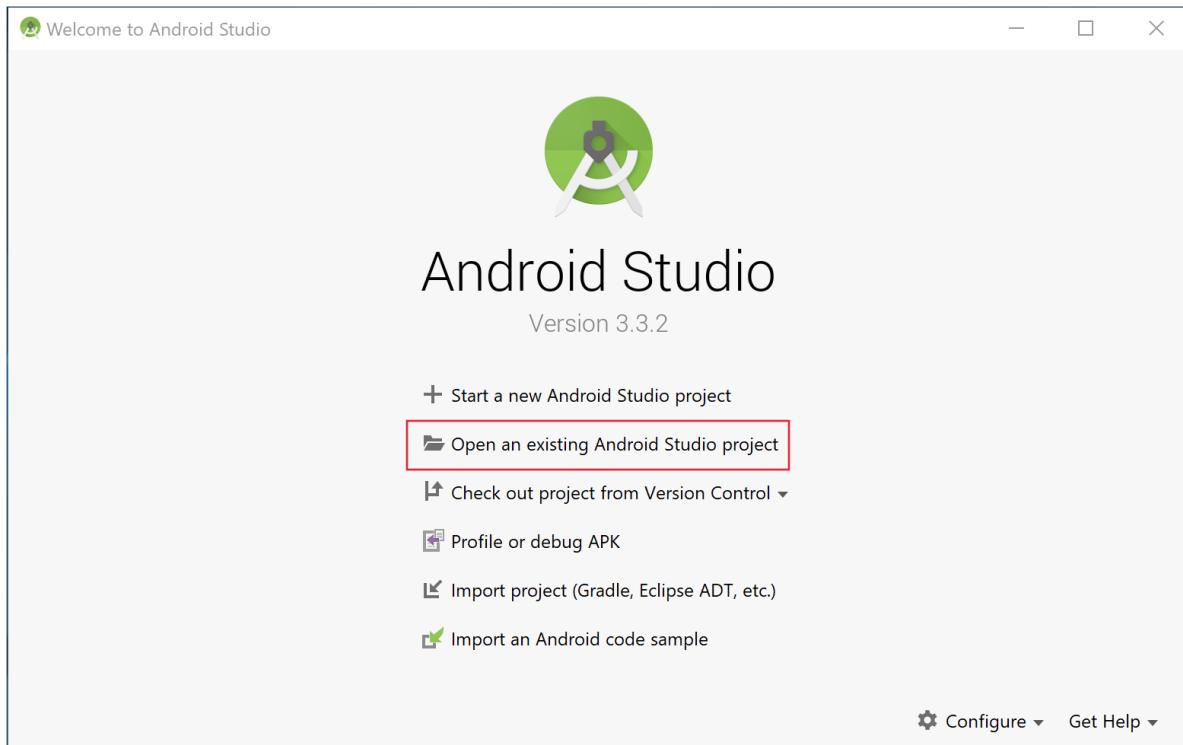


Es posible que algunas compañías tengan un tiempo límite en cuanto al tiempo que un dispositivo puede estar conectado a su sistema Wi-Fi. Además, puede que tras un determinado número de días deba ampliar el registro del kit de desarrollo en el sistema Wi-Fi.

# Ejecutar la aplicación de ejemplo

Para validar la configuración del kit de desarrollo, compile e instale la aplicación de ejemplo:

1. Inicie Android Studio.
2. Seleccione **Abrir un proyecto existente de Android Studio**.



3. Vaya a C:\SDSDK\Android-Sample-Release\example. Seleccione **Aceptar** para abrir el proyecto de ejemplo.
4. Agregue la clave de suscripción de Voz al código fuente. Si quiere probar el reconocimiento de intenciones, agregue también la clave de suscripción y el identificador de aplicación del [servicio Language Understanding](#).

Para Voz y LUIS, la información se trasladará a `MainActivity.java`:

```
// Subscription
private static String SpeechSubscriptionKey = "<enter your subscription info here>";
private static String SpeechRegion = "westus"; // You can change this if your speech region is different.
private static String LuisSubscriptionKey = "<enter your subscription info here>";
private static String LuisRegion = "westus2"; // you can change this, if you want to test the intent, and your LUIS region is different.
private static String LuisAppId = "<enter your LUIS AppId>";
```

Si usa la transcripción de conversaciones, la información de la clave y la región de Voz también se necesitará en `conversation.java`:

```
private static final String CTSKey = "<Conversation Transcription Service Key>";
private static final String CTSRegion="<Conversation Transcription Service Region>";// Region may be "centralus" or "eastasia"
```

5. La palabra clave predeterminada es "Equipo". Además, puede probar una de las otras palabras clave proporcionadas, como "Máquina" o "Asistente". Los archivos de recursos de estas palabras clave alternativas pueden encontrarse en Speech Devices SDK en la carpeta de palabras clave. Por ejemplo,

C:\SDSDK\Android-Sample-Release\keyword\Computer contiene los archivos que se usan para la palabra clave "Equipo".

**TIP**

También puede [crear una palabra clave personalizada](#).

Para usar una nueva palabra clave, actualice las dos líneas siguientes en `MainActivity.java` y copie el paquete de palabras clave en la aplicación. Por ejemplo, para usar la palabra clave "Máquina" desde el paquete de palabras clave `kws-machine.zip`:

- Copie el paquete de palabras clave en la carpeta "C:\SDSDK\Android-Sample-Release\example\app\src\main\assets".
- Actualice `MainActivity.java` con la palabra clave y el nombre del paquete:

```
private static final String Keyword = "Machine";
private static final String KeywordModel = "kws-machine.zip" // set your own keyword package name.
```

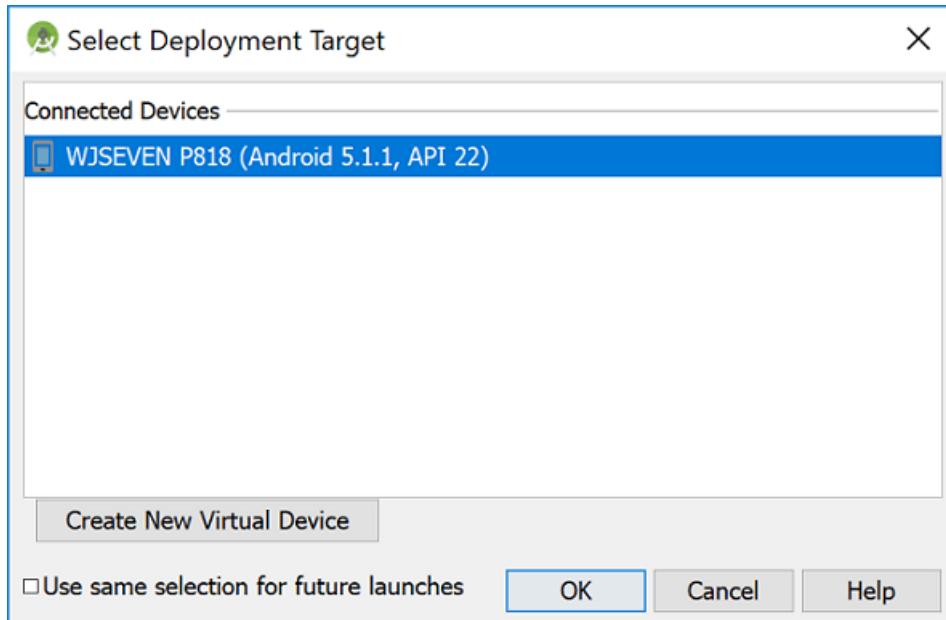
6. Actualice las siguientes líneas que contienen la configuración de geometría de matriz del micrófono:

```
private static final String DeviceGeometry = "Circular6+1";
private static final String SelectedGeometry = "Circular6+1";
```

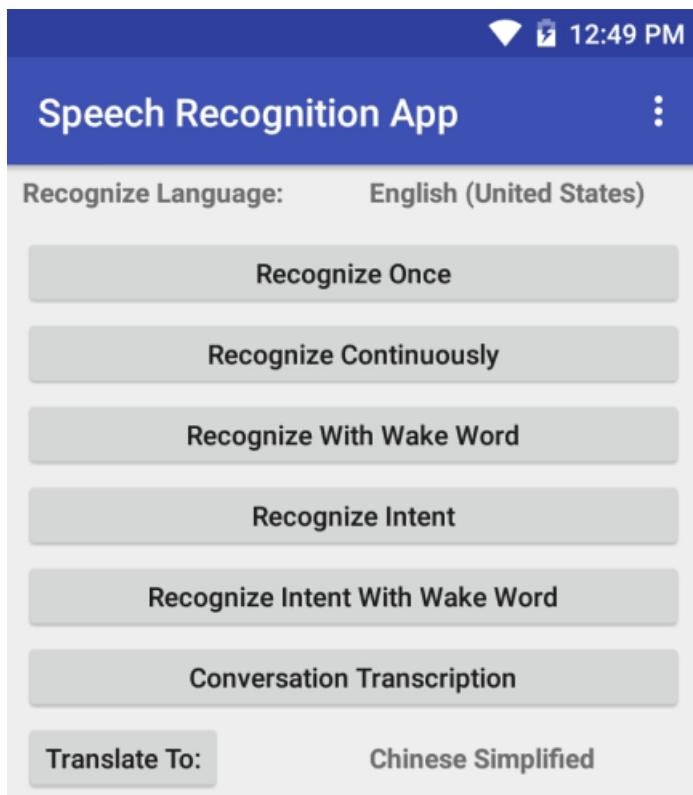
Esta tabla enumera los valores admitidos:

VARIABLE	SIGNIFICADO	VALORES DISPONIBLES
<code>DeviceGeometry</code>	Configuración del micrófono físico	Para un kit de desarrollo circular: <code>Circular6+1</code>
		Para un kit de desarrollo lineal: <code>Linear4</code>
<code>SelectedGeometry</code>	Configuración de micrófono de software	Para un kit de desarrollo circular que usa todos los micrófonos: <code>Circular6+1</code>
		Para un kit de desarrollo circular que usa cuatro micrófonos: <code>Circular3+1</code>
		Para un kit de desarrollo lineal que usa todos los micrófonos: <code>Linear4</code>
		Para un kit de desarrollo lineal que usa dos micrófonos: <code>Linear2</code>

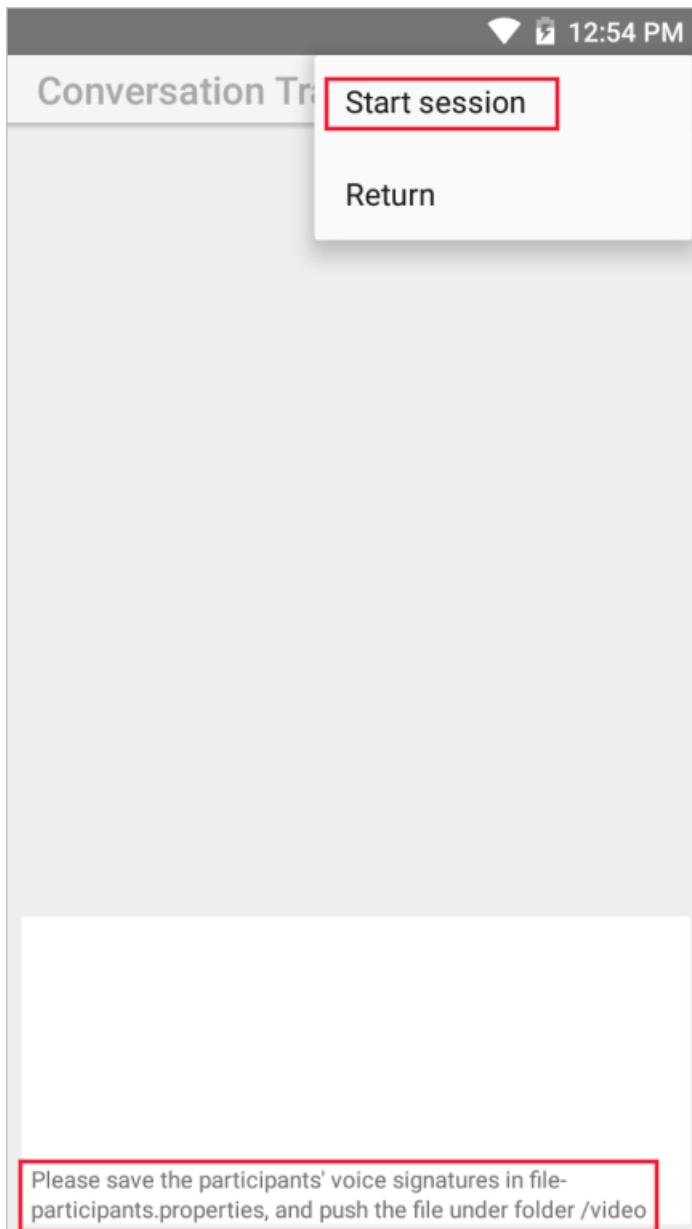
7. Para compilar la aplicación, en el menú **Run** (Ejecutar), seleccione **Run 'app'** (Ejecutar "aplicación"). Aparecerá el cuadro de diálogo **Select Deployment Target** (Seleccionar el destino de la implementación).
8. Elija el dispositivo y haga clic en **OK** (Aceptar) para implementar la aplicación en el dispositivo.



9. Se inicia la aplicación de ejemplo del SDK de dispositivos de voz y muestra las siguientes opciones:



10. Pruebe la nueva demostración de transcripción de conversaciones. Empiece a transcribir con "Iniciar sesión". De forma predeterminada, todos los usuarios son invitados. Sin embargo, si dispone de las firmas de voz del participante, estas se pueden colocar en un archivo `/video/participants.properties` en el dispositivo. Para generar la firma de voz, consulte [Transcripción de conversaciones \(SDK\)](#).



11. Experimente.

## Solución de problemas

Si no se puede conectar a Speech Devices: Escriba el siguiente comando en una ventana del símbolo del sistema. Devolverá una lista de resultados:

```
adb devices
```

### NOTE

Este comando usa Android Debug Bridge, `adb.exe`, que forma parte de la instalación de Android Studio. Esta herramienta se encuentra en `C:\Users[nombre de usuario]\AppData\Local\Android\Sdk\platform-tools`. Puede agregar este directorio a la ruta de acceso para que sea más práctico invocar `adb`. En caso contrario, debe especificar la ruta de acceso completa a la instalación de `adb.exe` en todos los comandos que invoca `adb`.

Si ve un error `no devices/emulators found`, compruebe que el cable USB está conectado y que es un cable de alta calidad.

## Pasos siguientes

[Revise las notas de la versión.](#)

# Solución de problemas de Speech Devices SDK

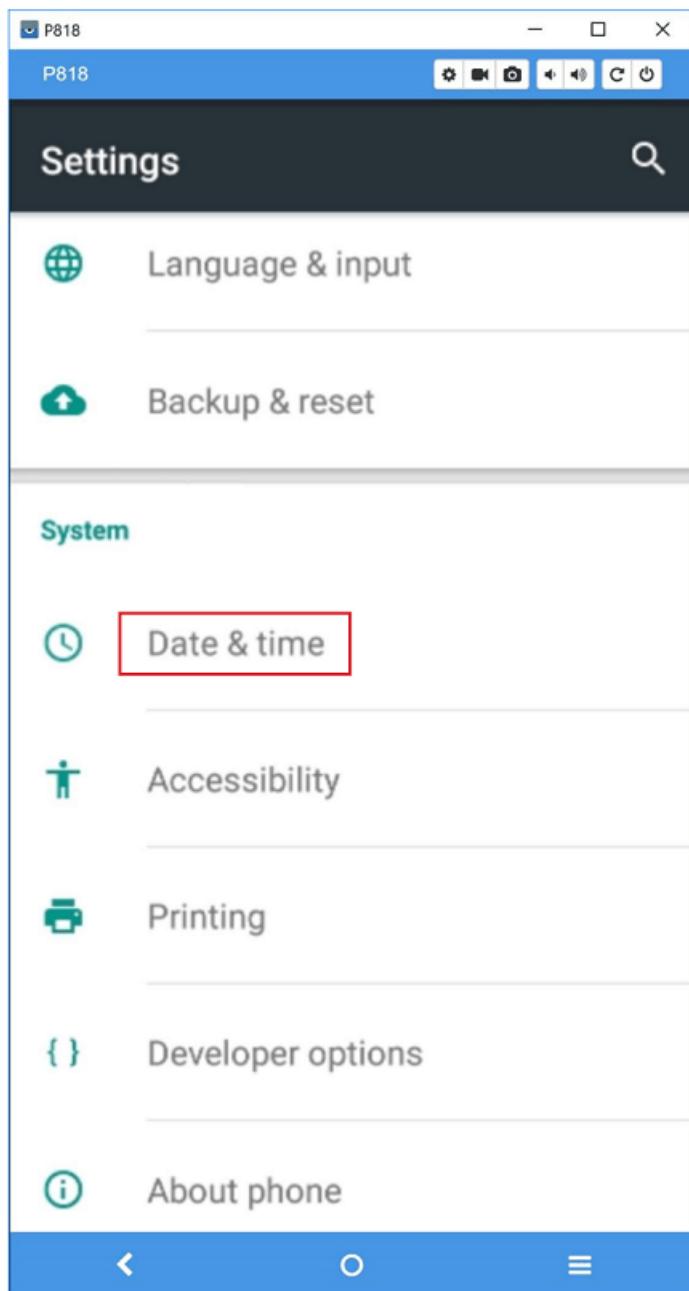
13/01/2020 • 2 minutes to read • [Edit Online](#)

En este artículo se proporciona información para ayudarlo a resolver los problemas que pueden surgir al usar el SDK de servicios de Voz.

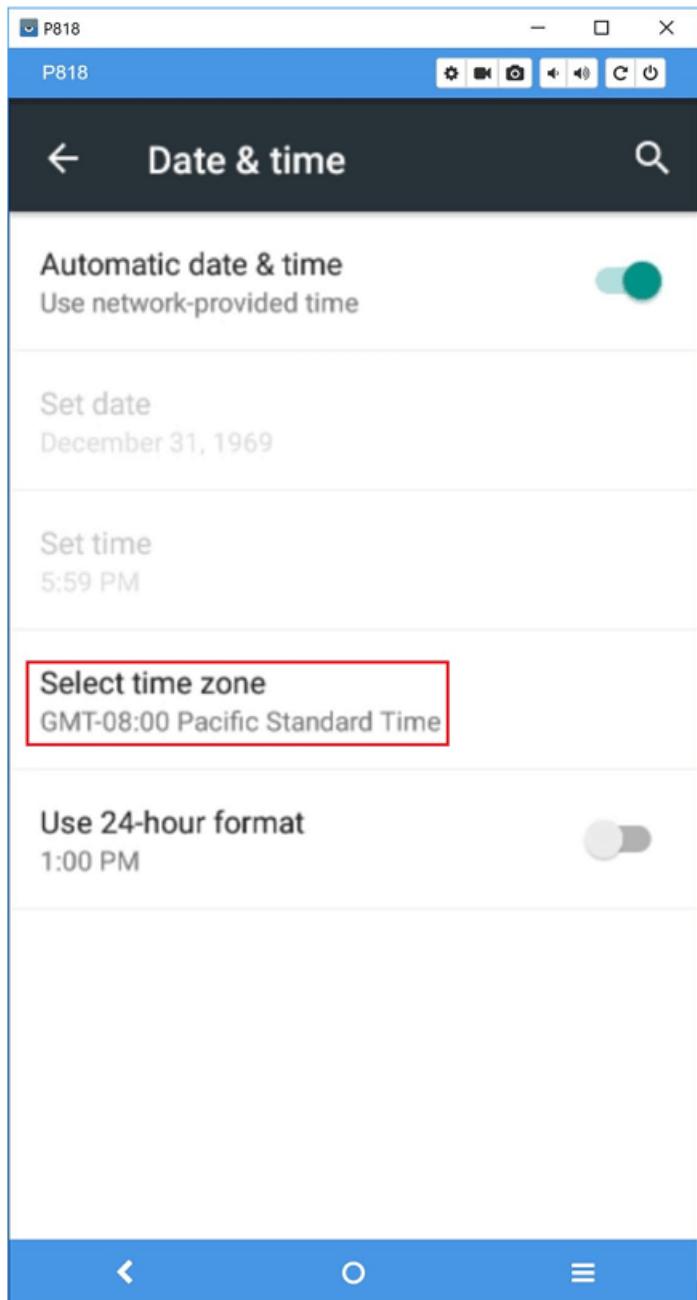
## Errores de certificado

Si recibe errores de certificado cuando use el servicio de voz, asegúrese de que el dispositivo tiene la fecha y hora correctas:

1. Vaya a **Configuración**. En **System** (Sistema), seleccione **Date & time** (Fecha y hora).



2. Mantenga la opción **Automatic date & time** (Fecha y hora automáticas) seleccionada. En **Select time zone** (Seleccionar zona horaria), seleccione la zona horaria actual.



Cuando vea que la hora del kit desarrollo coincide con la hora del equipo, sabrá que el kit de desarrollo está conectado a Internet.

## Pasos siguientes

- [Revise las notas de la versión.](#)

# Notas de la versión: SDK de dispositivos de voz

14/01/2020 • 6 minutes to read • [Edit Online](#)

En las siguientes secciones se indican los cambios en las versiones más recientes.

## SDK de dispositivos de voz 1.7.0:

- Ahora se admite Linux ARM.
- Se proporcionan archivos binarios iniciales para Roobo v2 (Linux ARM64).
- Los usuarios de Windows pueden usar `AudioConfig.fromDefaultMicrophoneInput()` o `AudioConfig.fromMicrophoneInput(deviceName)` para especificar el micrófono que se utilizará.
- Se ha optimizado el tamaño de la biblioteca.
- Compatibilidad con el reconocimiento de múltiples turnos mediante el mismo objeto reconocedor de voz/intención.
- Se ha corregido el bloqueo ocasional que se produciría al detener el reconocimiento.
- Las aplicaciones de ejemplo ahora contienen un archivo `participants.properties` de ejemplo para demostrar el formato del archivo.
- Se ha actualizado el componente [SDK de Voz](#) a la versión 1.7.0. Para más información, consulte sus [notas de la versión](#).

## SDK de dispositivos de voz 1.6.0:

- Compatibilidad con [Azure Kinect DK](#) en Windows y Linux con una [aplicación de ejemplo](#) común
- Se ha actualizado el componente [Speech SDK](#) a la versión 1.6.0. Para más información, consulte sus [notas de la versión](#).

## SDK de dispositivos de voz 1.5.1:

- Incluya [transcripción de conversaciones](#) en la aplicación de ejemplo.
- Se ha actualizado el componente [Speech SDK](#) a la versión 1.5.1. Para más información, consulte sus [notas de la versión](#).

## SDK de dispositivos de voz 1.5.0: Versión de mayo de 2019

- SDK de dispositivos de voz es ahora disponibilidad general y ya no está en versión preliminar controlada.
- Se ha actualizado el componente [Speech SDK](#) a la versión 1.5.0. Para más información, consulte sus [notas de la versión](#).
- La nueva tecnología de palabra clave proporciona importantes mejoras de calidad; consulte Cambios importantes.
- Nueva canalización de procesamiento de audio para el reconocimiento mejorado de campo lejano.

### Cambios importantes

- debido a la nueva tecnología de palabra clave, todas las palabras clave deben volver a crearse en nuestro portal mejorado de palabras clave. Para quitar completamente las palabras clave antiguas del dispositivo, desinstale la aplicación antigua.
  - adb uninstall com.microsoft.cognitiveservices.speech.samples.sdsdkstarterapp

## SDK de dispositivos de voz 1.4.0: Versión de abril de 2019

- Se ha actualizado el componente [Speech SDK](#) a la versión 1.4.0. Para más información, consulte sus [notas de la versión](#).

## SDK de dispositivos de voz 1.3.1: Versión de marzo de 2019

- Se ha actualizado el componente [Speech SDK](#) a la versión 1.3.1. Para más información, consulte sus [notas de la versión](#).
- Administración de palabras clave; consulte Cambios importantes.
- La aplicación de ejemplo agrega la opción de idioma para el reconocimiento y la traducción de voz.

### Cambios importantes

- La [instalación de una palabra clave](#) se ha simplificado, y ahora es parte de la aplicación (no necesita una instalación independiente en el dispositivo).
- El reconocimiento de la palabra clave ha cambiado, y se admiten dos eventos.
  - `RecognizingKeyword`, indica que el resultado de voz contiene texto de palabras clave (sin verificar).
  - `RecognizedKeyword` indica que el reconocimiento de la palabra clave terminó de reconocer la palabra clave especificada.

## SDK de dispositivos de voz 1.1.0: Versión de noviembre de 2018

- Se ha actualizado el componente [Speech SDK](#) a la versión 1.1.0. Para más información, consulte sus [notas de la versión](#).
- Ha aumentado la precisión del reconocimiento de voz a gran distancia gracias a nuestro algoritmo mejorado de procesamiento de audio.
- La aplicación de ejemplo ahora es compatible con el reconocimiento de voz en chino.

## SDK de dispositivos de voz 1.0.1: versión de octubre de 2018

- Se ha actualizado el componente [Speech SDK](#) a la versión 1.0.1. Para más información, consulte sus [notas de la versión](#).
- La precisión del reconocimiento de voz se verá mejorada gracias a nuestro algoritmo mejorado de procesamiento de audio.
- Se ha corregido un error en la sesión de audio de reconocimiento continuo.

### Cambios importantes

- Con esta versión se presentan una serie de cambios importantes. Consulte [esta página](#) para obtener detalles relacionados con las API.
- Los archivos de modelo de KWS no son compatibles con Speech Devices SDK 1.0.1. Los archivos existentes de palabras clave se eliminarán después de que los nuevos se escriban en el dispositivo.

## SDK de dispositivos de voz 0.5.0: versión de agosto de 2018

- Se ha mejorado la precisión del reconocimiento de voz mediante la corrección de un error en el código de procesamiento de audio.
- Se ha actualizado el componente [Speech SDK](#) a la versión 0.5.0. Para más información, consulte sus [notas de la versión](#).

## SDK de dispositivos de voz 0.2.12733: Versión de mayo de 2018

La primera versión preliminar pública de Speech Devices SDK de Cognitive Services.

# Migración de Bing Speech al servicio de voz

13/01/2020 • 9 minutes to read • [Edit Online](#)

Use este artículo para migrar sus aplicaciones de Bing Speech API al servicio de voz.

Este artículo describe las diferencias entre Bing Speech API y el servicio de voz, y sugiere estrategias para migrar las aplicaciones. El servicio de voz no funcionarán con su clave de suscripción de Bing Speech API, por lo que necesitará una nueva suscripción al servicio de voz.

Una única clave de suscripción al servicio de voz concede acceso a las siguientes características. Cada una se mide por separado, por lo que se le cobrará solo por las características que use.

- [Voz a texto](#)
- [Conversión de voz a texto personalizada](#)
- [Texto a voz](#)
- [Conversión de texto a voz personalizada](#)
- [Traducción de voz](#) (no incluye [traducción de texto](#))

El [Speech SDK](#) es un reemplazo funcional para las bibliotecas de cliente de Bing Speech, pero usa una API diferente.

## Comparación de características

El servicio de voz es similar en gran medida a Bing Speech, con las siguientes diferencias.

CARACTERÍSTICA	BING SPEECH	SERVICIO DE VOZ	DETALLES
SDK de C++	□	✓□	El servicio de voz admite Windows y Linux.
SDK de Java	✓□	✓□	El servicio de voz admite dispositivos Android y Speech.
SDK DE C#	✓□	✓□	El servicio de voz admite Windows 10, Plataforma universal de Windows (UWP) y .NET Standard 2.0.
Reconocimiento de voz continua	10 minutos	Ilimitado (con SDK)	Tanto los protocolos de WebSockets del servicio de voz como Bing Speech admiten hasta 10 minutos por llamada. Sin embargo, el Speech SDK se vuelve a conectar automáticamente en tiempo de expiración o desconexiones.
Resultados intermedios o parciales	✓□	✓□	Con el protocolo WebSockets o el SDK.

CARACTERÍSTICA	BING SPEECH	SERVICIO DE VOZ	DETALLES
Modelos de voz personalizados	✓ <input type="checkbox"/>	✓ <input type="checkbox"/>	Bing Speech requiere una suscripción independiente de Custom Speech.
Cuentas de voz personalizadas	✓ <input type="checkbox"/>	✓ <input type="checkbox"/>	Bing Speech requiere una suscripción independiente de Custom Voice.
Voces de 24 kHz	<input type="checkbox"/>	✓ <input type="checkbox"/>	
Reconocimiento de intenciones mediante voz	Requiere una llamada API de LUIS independiente	Integrada (con el SDK)	Puede usar una clave de LUIS con el servicio de voz.
Reconocimiento de intenciones simple	<input type="checkbox"/>	✓ <input type="checkbox"/>	
Transcripción de lotes de archivos de sonido de larga duración	<input type="checkbox"/>	✓ <input type="checkbox"/>	
Modo de reconocimiento	Manual a través de URI de punto de conexión	Automático	El modo de reconocimiento no está disponible en el servicio de voz.
Localidad de punto de conexión	Global	Regional	Los puntos de conexión regionales mejoran la latencia.
API de REST	✓ <input type="checkbox"/>	✓ <input type="checkbox"/>	Las API REST del servicio de voz son compatibles con Bing Speech (punto de conexión diferente). Las API de REST admiten la funcionalidad de texto a voz y la funcionalidad de voz a texto de forma limitada.
Protocolos de WebSockets	✓ <input type="checkbox"/>	✓ <input type="checkbox"/>	La API WebSockets del servicio de voz es compatible con Bing Speech (punto de conexión diferente). Migré a Speech SDK si es posible para simplificar el código.
Llamadas API de servicio a servicio	✓ <input type="checkbox"/>	<input type="checkbox"/>	Proporcionado en Bing Speech a través de la biblioteca de servicio de C#.
SDK de código abierto	✓ <input type="checkbox"/>	<input type="checkbox"/>	

El servicio de voz usa un modelo de precios basado en el tiempo (en lugar de un modelo basado en transacciones). Consulte los [precios del servicio de voz](#) para más información.

## Estrategias de migración

Si su organización o usted tienen aplicaciones en desarrollo o producción que usan la API Bing Speech, deben actualizarlas para que usen el servicio de voz tan pronto como sea posible. Vea la [documentación del servicio de voz](#) para información sobre los SDK disponibles, ejemplos de código y tutoriales.

Las [API REST](#) del servicio de voz son compatibles con las API Bing Speech. Si actualmente usa las API REST Bing Speech, solo debe cambiar el punto de conexión de REST y cambiar a una clave de suscripción del servicio de voz.

Los protocolos de WebSockets del servicio de voz también son compatibles con los utilizados por Bing Speech. Para nuevos desarrollos, se recomienda usar el SDK de Voz en lugar de WebSockets. También es una buena idea migrar el código existente al SDK. No obstante, como con las API de REST, el código existente que utiliza Bing Speech a través de WebSockets requiere solo un cambio en el punto de conexión y una clave actualizada.

Si usa una biblioteca de cliente de Bing Speech para un lenguaje de programación específico, migrar al [Speech SDK](#) requerirá cambios en la aplicación porque la API es diferente. El Speech SDK puede simplificar el código al mismo tiempo que le proporciona acceso a nuevas características.

Actualmente, el SDK de Voz es compatible con C# ([puede ver detalles aquí](#)), Java (dispositivos Android y personalizados), Objective C (iOS), C++ (Windows y Linux), y JavaScript. Las API en todas las plataformas son similares, lo que acelera el desarrollo multiplataforma.

El servicio de voz no ofrece un punto de conexión global. Determine si la aplicación funciona eficazmente con un único punto de conexión regional para todo su tráfico. Si no lo hace, use la ubicación geográfica para determinar el punto de conexión más eficaz. Necesitará una suscripción del servicio de voz independiente en cada región que use.

Si la aplicación usa conexiones de larga duración y no puede usar un SDK disponible, puede usar una conexión de WebSockets. Administre el límite de tiempo de espera de 10 minutos volviendo a conectarse en los momentos adecuados.

Para empezar a usar el Speech SDK:

1. Descargue el [SDK de Voz](#).
2. Utilice las [guías de inicio rápido](#) y los [tutoriales](#) del servicio de voz. Examine también los [ejemplos de código](#) para obtener experiencia con las nuevas API.
3. Actualice la aplicación para usar el servicio de voz.

## Soporte técnico

Los clientes de Bing Speech deben ponerse en contacto con el soporte al cliente mediante una [incidencia de soporte técnico](#). También puede ponerse contacto con nosotros si sus necesidades de soporte técnico requieren un [Plan de soporte técnico](#).

Para información sobre compatibilidad de API, SDK y el servicio de voz, visite la [página de soporte técnico](#) del servicio de voz.

## Pasos siguientes

- [Prueba gratuita del servicio de voz](#)
- [Inicio rápido: Reconocimiento de voz en una aplicación de UWP mediante el SDK de Voz](#)

## Otras referencias

- [Notas de la versión del servicio de voz](#)
- [¿Qué es el servicio Voz?](#)
- [Documentación del servicio de voz y del SDK de voz](#)

# Migración del servicio Custom Speech al servicio de VOZ

13/01/2020 • 2 minutes to read • [Edit Online](#)

Use este artículo para migrar las aplicaciones del servicio Custom Speech al servicio de voz.

El servicio Custom Speech ahora forma parte del servicio de voz. Cambie al servicio de voz para aprovechar las actualizaciones más recientes de características y calidad.

## Migración para los nuevos clientes

El modelo de precios es más sencillo, al pasar a un modelo de precios por hora para el servicio de voz.

1. Cree un recurso de Azure en cada región donde la aplicación esté disponible. El nombre del recurso de Azure es **Voz**. Puede usar un único recurso de Azure para los siguientes servicios en la misma región, en lugar de crear recursos independientes:
  - Voz a texto
  - Conversión de voz a texto personalizada
  - Texto a voz
  - Traducción de voz
2. Descargue el [SDK de Voz](#).
3. Siga las guías de inicio rápido y los ejemplos del SDK para usar las API correctas. Si usa las API de REST, también debe usar los puntos de conexión y las claves de recursos correctos.
4. Actualice la aplicación cliente para usar las API y el servicio Voz.

## Migración para los clientes actuales

Migre sus claves de recursos existentes al servicio de voz en el portal de este servicio. Para ello, siga los pasos que se describen a continuación:

### NOTE

Las claves de recursos solo se pueden migrar en la misma región.

1. Inicie sesión en el portal [cris.ai](#) y seleccione la suscripción en el menú superior derecho.
2. Seleccione **Migrate selected subscription** (Migrar suscripción seleccionada).
3. Escriba la clave de suscripción en el cuadro de texto y seleccione **Migrar**.

## Pasos siguientes

- [Prueba gratuita del servicio de voz](#).
- Obtenga información sobre los conceptos de [conversión de voz en texto](#).

## Otras referencias

- [¿Qué es el servicio Voz?](#)

- Documentación del servicio de voz y del SDK de voz

# Migración de Translator Speech API al servicio de voz

13/01/2020 • 4 minutes to read • [Edit Online](#)

Use este artículo para migrar las aplicaciones de Microsoft Translator Speech API al [servicio de voz](#). Esta guía describe las diferencias entre Translator Speech API y el servicio de voz, y sugiere estrategias para migrar las aplicaciones.

## NOTE

El servicio de voz no aceptará su clave de suscripción de Translator Speech API. Deberá crear una nueva suscripción al servicio de voz.

## Comparación de características

CARACTERÍSTICA	TRANSLATOR SPEECH API	SERVICIO DE VOZ	DETALLES
Traducción a texto	✓ <input type="checkbox"/>	✓ <input type="checkbox"/>	
Traducción a voz	✓ <input type="checkbox"/>	✓ <input type="checkbox"/>	
Punto de conexión global	✓ <input type="checkbox"/>	<input type="checkbox"/>	El servicio de voz no ofrece un punto de conexión global. Un punto de conexión global puede dirigir automáticamente el tráfico al punto de conexión regional más cercano, reduciendo la latencia en la aplicación.
Puntos de conexión regionales	<input type="checkbox"/>	✓ <input type="checkbox"/>	
Límite de tiempo de conexión	90 minutos	Ilimitado con SDK. 10 minutos con una conexión WebSockets	
Clave de autenticación de encabezado	✓ <input type="checkbox"/>	✓ <input type="checkbox"/>	
Varios idiomas que se traducen en una sola solicitud	<input type="checkbox"/>	✓ <input type="checkbox"/>	
SDK disponibles	<input type="checkbox"/>	✓ <input type="checkbox"/>	Vea la <a href="#">Documentación del servicio de voz</a> para obtener información de los SDK disponibles.
Conexiones de WebSockets	✓ <input type="checkbox"/>	✓ <input type="checkbox"/>	

CARACTERÍSTICA	TRANSLATOR SPEECH API	SERVICIO DE VOZ	DETALLES
API Languages	✓ <input type="checkbox"/>	<input type="checkbox"/>	El servicio de voz admite el mismo conjunto de idiomas que se describe en el artículo de <a href="#">referencia de idiomas de las API Translator</a> .
Marcador y filtro de obscenidad	<input type="checkbox"/>	✓ <input type="checkbox"/>	
.WAV/PCM como entrada	✓ <input type="checkbox"/>	✓ <input type="checkbox"/>	
Otros tipos de archivo como entrada	<input type="checkbox"/>	<input type="checkbox"/>	
Resultados parciales	✓ <input type="checkbox"/>	✓ <input type="checkbox"/>	
Información de control de tiempo	✓ <input type="checkbox"/>	<input type="checkbox"/>	
Id. de correlación	✓ <input type="checkbox"/>	<input type="checkbox"/>	
Modelos de voz personalizados	<input type="checkbox"/>	✓ <input type="checkbox"/>	El servicio de voz ofrece modelos de voz personalizados que le permiten personalizar el reconocimiento de voz con el vocabulario único de su organización.
Modelos de traducción personalizados	<input type="checkbox"/>	✓ <input type="checkbox"/>	Si se suscribe a Microsoft Translator Text API, podrá utilizar <a href="#">Traductor personalizado</a> , lo que le permitirá emplear sus propios datos para conseguir traducciones más precisas.

## Estrategias de migración

Si su organización o usted tienen aplicaciones en desarrollo o producción que usan la API Translator Speech, deben actualizarlas para que usen el servicio de voz. Vea la documentación del [servicio de voz](#) para obtener información sobre SDK disponibles, ejemplos de código y tutoriales. Tenga en cuenta lo siguiente cuando esté migrando:

- El servicio de voz no ofrece un punto de conexión global. Determine si la aplicación funciona eficazmente con un único punto de conexión regional para todo su tráfico. Si no lo hace, use la ubicación geográfica para determinar el punto de conexión más eficaz.
- Si la aplicación usa conexiones de larga duración y no puede usar un SDK disponible, puede usar una conexión de WebSockets. Administre el límite de tiempo de espera de 10 minutos volviendo a conectarse en los momentos adecuados.
- Si la aplicación utiliza las API Translator Text y Translator Speech para habilitar los modelos de traducción personalizada, puede agregar los identificadores "Categoría" directamente mediante el servicio de voz.

- A diferencia de la API Translator Speech, el servicio de voz puede completar las traducciones en varios idiomas en una sola solicitud.

## Pasos siguientes

- [Prueba gratuita del servicio de voz](#)
- [Inicio rápido: Reconocimiento de voz en una aplicación de UWP mediante el SDK de Voz](#)

## Otras referencias

- [¿Qué es el servicio Voz?](#)
- [Documentación del servicio de voz y del SDK de voz](#)

# Opciones de ayuda y soporte técnico

13/01/2020 • 6 minutes to read • [Edit Online](#)

¿Está empezando a explorar la funcionalidad del servicio Speech? ¿Está implementando una nueva característica a la aplicación? Estas son las sugerencias acerca de dónde puede obtener ayuda como desarrollador.

- Manténgase informado sobre los nuevos desarrollos en *Azure Cognitive Services* o busque las últimas noticias relacionadas con el *servicio de voz*.
- Las notas de la versión contienen información para todas las versiones.
- Realice una búsqueda para ver la comunidad ya ha tratado este problema o si ya existe documentación de la característica que quiere implementar.
- Si no encuentra una respuesta satisfactoria, formule una pregunta en *Stack Overflow*.
- Si encuentra algún problema con uno de los ejemplos de GitHub, puede abrir un problema en *GitHub*.
- Puede buscar una solución en el *foro de UserVoice*.

## Manténgase informado

Las noticias sobre Cognitive Services se recopilan en el [blog de Cognitive Services](#). Para obtener la información más reciente sobre el servicio de voz, siga el [blog del servicio de voz](#).

## Notas de la versión

Las [notas de la versión](#) se actualizan en cuanto está disponible una nueva versión. Las notas contienen información sobre las nuevas características, mejoras y correcciones de errores.

## Search

Puede encontrar la respuesta que necesita en la documentación, en los ejemplos o en las respuestas a las preguntas de [Stack Overflow](#).

### Búsqueda restringida

Para obtener resultados más rápido, limite su búsqueda a Stack Overflow, la documentación y los ejemplos de código utilizando la consulta siguiente en su [motor de búsqueda favorito](#):

```
{Your Search Terms} (site:stackoverflow.com OR site:docs.microsoft.com OR site:github.com/azure-samples)
```

Donde *{Your Search Terms}* son sus palabras clave de búsqueda.

## Creación de una solicitud de soporte técnico de Azure

Los clientes de Azure pueden crear y administrar las solicitudes de soporte técnico en Azure Portal.

- [Azure Portal](#)
- [Azure Portal para el gobierno de los Estados Unidos](#)

## Publique una pregunta en Stack Overflow

Stack Overflow es el canal preferido para las preguntas relacionadas con el desarrollo. Es donde los

miembros de la comunidad y los miembros del equipo de Microsoft se implican directamente para ayudarle a solucionar problemas.

Si tras buscar una respuesta a su problema no la encuentra, envíe una nueva pregunta a Stack Overflow mediante el uso de las etiquetas [\[microsoft-cognitive\]](#)[\[speech\]](#).

#### TIP

Las siguientes publicaciones de Stack Overflow contienen sugerencias sobre cómo formular preguntas y agregar código fuente. Seguir estas directrices puede ayudarle a aumentar las posibilidades de que los miembros de la comunidad evalúen y respondan a su pregunta rápidamente:

- [Cómo se puede formular una buena pregunta](#)
- [Cómo crear un ejemplo mínimo, completo y comprobable](#)

## Creación de un problema de GitHub

A menudo se envían ejemplos como código abierto. Para preguntas y problemas, cree un *problema* en el repositorio de GitHub correspondiente. También puede enviar una solicitud de incorporación de cambios. La lista siguiente contiene vínculos a los repositorios de ejemplos:

- [Acerca del SDK de Voz](#)
- [Speech Devices SDK](#)

Puede crear un informe de errores, solicitar una característica o formular una pregunta general y compartir procedimientos recomendados. Para los informes de errores, siga la plantilla proporcionada:

### Describa el error

Proporcione una descripción clara y concisa de lo que es el error.

### Pasos de reproducción

Pasos para reproducir el comportamiento:

1. ...
2. ...

### Comportamiento esperado

Proporcione una descripción clara y concisa de lo que esperaba que sucediera.

### Versión del SDK de Voz de Cognitive Services

¿Qué versión del SDK está usando?

### Plataforma, sistema operativo y lenguaje de programación

- SO: sea específico; por ejemplo, Windows, Linux, Android, iOS, etc.
- Hardware: x64, x86, ARM, etc.
- Explorador: sea específico; por ejemplo, Chrome, Safari (si procede)

### Contexto adicional

- Mensajes de error, información de registro, seguimiento de la pila, etc.
- Si informa de un error de interacción de un determinado servicio, indique los valores de SessionId y time (incluido timezone) de los incidentes notificados. SessionId se indica en todas las llamadas y eventos que recibe.

- Cualquier otra información adicional

## Foro de UserVoice

Comparta sus ideas para hacer que Cognitive Services y las API que lo acompañan funcionen mejor para las aplicaciones que desarrolla. Use nuestra creciente base de conocimiento para encontrar respuestas a preguntas habituales:

[UserVoice](#)

# Regiones admitidas del servicio de voz

13/01/2020 • 6 minutes to read • [Edit Online](#)

El servicio Voz permite que la aplicación convierta audio en texto, lleve a cabo la traducción de voz y convertir texto a voz. Este servicio está disponible en varias regiones con puntos de conexión únicos para SDK de Voz y API REST.

Asegúrese de usar el punto de conexión que coincide con la región de su suscripción.

## SDK de voz

En las regiones de [SDK de Voz](#), las regiones se especifican como una cadena (por ejemplo, como un parámetro `SpeechConfig.FromSubscription` en el SDK de Voz para C#).

### Voz a texto, texto a voz y traducción

El SDK de Voz está disponible en estas regiones para el **reconocimiento de voz**, el **texto a voz** y la **traducción**:

REGION	PARÁMETRO DEL SDK DE VOZ	PORTAL DE PERSONALIZACIÓN DE VOZ
Oeste de EE. UU.	<code>westus</code>	<a href="https://westus.cris.ai">https://westus.cris.ai</a>
Oeste de EE. UU. 2	<code>westus2</code>	<a href="https://westus2.cris.ai">https://westus2.cris.ai</a>
East US	<code>eastus</code>	<a href="https://eastus.cris.ai">https://eastus.cris.ai</a>
Este de EE. UU. 2	<code>eastus2</code>	<a href="https://eastus2.cris.ai">https://eastus2.cris.ai</a>
Centro de EE. UU.	<code>centralus</code>	<a href="https://centralus.cris.ai">https://centralus.cris.ai</a>
Centro-Norte de EE. UU.	<code>northcentralus</code>	<a href="https://northcentralus.cris.ai">https://northcentralus.cris.ai</a>
Centro-Sur de EE. UU.	<code>southcentralus</code>	<a href="https://southcentralus.cris.ai">https://southcentralus.cris.ai</a>
India Central	<code>centralindia</code>	<a href="https://centralindia.cris.ai">https://centralindia.cris.ai</a>
Asia oriental	<code>eastasia</code>	<a href="https://eastasia.cris.ai">https://eastasia.cris.ai</a>
Sudeste asiático	<code>southeastasia</code>	<a href="https://southeastasia.cris.ai">https://southeastasia.cris.ai</a>
Este de Japón	<code>japaneast</code>	<a href="https://japaneast.cris.ai">https://japaneast.cris.ai</a>
Corea Central	<code>koreacentral</code>	<a href="https://koreacentral.cris.ai">https://koreacentral.cris.ai</a>
Este de Australia	<code>australiaeast</code>	<a href="https://australiaeast.cris.ai">https://australiaeast.cris.ai</a>
Centro de Canadá	<code>canadacentral</code>	<a href="https://canadacentral.cris.ai">https://canadacentral.cris.ai</a>
Europa del Norte	<code>northeurope</code>	<a href="https://northeurope.cris.ai">https://northeurope.cris.ai</a>
Europa occidental	<code>westeurope</code>	<a href="https://westeurope.cris.ai">https://westeurope.cris.ai</a>
Sur de Reino Unido 2	<code>uksouth</code>	<a href="https://uksouth.cris.ai">https://uksouth.cris.ai</a>
Centro de Francia	<code>francecentral</code>	<a href="https://francecentral.cris.ai">https://francecentral.cris.ai</a>

### Reconocimiento de la intención

Las regiones disponibles para el **reconocimiento de la intención** mediante el SDK de Voz son las siguientes:

REGIÓN GLOBAL	REGION	PARÁMETRO DEL SDK DE VOZ
Asia	Asia oriental	<code>eastasia</code>
Asia	Sudeste asiático	<code>southeastasia</code>
Australia	Este de Australia	<code>australiaeast</code>
Europa	Europa del Norte	<code>northeurope</code>
Europa	Europa occidental	<code>westeurope</code>
Norteamérica	East US	<code>eastus</code>

REGION GLOBAL	REGION	PARÁMETRO DEL SDK DE VOZ
Norteamérica	Este de EE. UU. 2	eastus2
Norteamérica	Centro-Sur de EE. UU.	southcentralus
Norteamérica	Centro occidental de EE.UU.	westcentralus
Norteamérica	Oeste de EE. UU.	westus
Norteamérica	Oeste de EE. UU. 2	westus2
Sudamérica	Sur de Brasil	brazilsouth

Se trata de un subconjunto de las regiones de publicación compatibles con el [servicio Language Understanding \(LUIS\)](#).

#### Asistentes de voz

El [SDK de Voz](#) admite funcionalidades de **asistente de voz** en estas regiones:

REGION	PARÁMETRO DEL SDK DE VOZ
Oeste de EE. UU.	westus
Oeste de EE. UU. 2	westus2
East US	eastus
Este de EE. UU. 2	eastus2
Europa occidental	westeurope
Europa del Norte	northeurope
Sudeste asiático	southeastasia

## API de REST

El servicio Voz también expone puntos de conexión REST para las solicitudes de voz a texto y texto a voz.

#### Voz a texto

Para obtener la documentación de referencia sobre la opción de voz a texto, consulte las [API de REST de voz a texto](#).

REGION	PUNTO DE CONEXIÓN
Este de Australia	<a href="https://australiaeast.stt.speech.microsoft.com/speech/recognition/conversation">https://australiaeast.stt.speech.microsoft.com/speech/recognition/conversation</a>
Centro de Canadá	<a href="https://canadacentral.stt.speech.microsoft.com/speech/recognition/conversation">https://canadacentral.stt.speech.microsoft.com/speech/recognition/conversation</a>
Centro de EE. UU.	<a href="https://centralus.stt.speech.microsoft.com/speech/recognition/conversation">https://centralus.stt.speech.microsoft.com/speech/recognition/conversation</a>
Asia oriental	<a href="https://eastasia.stt.speech.microsoft.com/speech/recognition/conversation/">https://eastasia.stt.speech.microsoft.com/speech/recognition/conversation/</a>
Este de EE. UU.	<a href="https://eastus.stt.speech.microsoft.com/speech/recognition/conversation/conversation">https://eastus.stt.speech.microsoft.com/speech/recognition/conversation/conversation</a>
Este de EE. UU. 2	<a href="https://eastus2.stt.speech.microsoft.com/speech/recognition/conversation/conversation">https://eastus2.stt.speech.microsoft.com/speech/recognition/conversation/conversation</a>
Centro de Francia	<a href="https://francecentral.stt.speech.microsoft.com/speech/recognition/conversation">https://francecentral.stt.speech.microsoft.com/speech/recognition/conversation</a>
India central	<a href="https://centralindia.stt.speech.microsoft.com/speech/recognition/conversation">https://centralindia.stt.speech.microsoft.com/speech/recognition/conversation</a>
Este de Japón	<a href="https://japaneast.stt.speech.microsoft.com/speech/recognition/conversation">https://japaneast.stt.speech.microsoft.com/speech/recognition/conversation</a>
Corea Central	<a href="https://koreacentral.stt.speech.microsoft.com/speech/recognition/conversation">https://koreacentral.stt.speech.microsoft.com/speech/recognition/conversation</a>
Centro-Norte de EE. UU.	<a href="https://northcentralus.stt.speech.microsoft.com/speech/recognition/conversation">https://northcentralus.stt.speech.microsoft.com/speech/recognition/conversation</a>
Europa del Norte	<a href="https://northeurope.stt.speech.microsoft.com/speech/recognition/conversation">https://northeurope.stt.speech.microsoft.com/speech/recognition/conversation</a>
Centro-Sur de EE. UU.	<a href="https://southcentralus.stt.speech.microsoft.com/speech/recognition/conversation">https://southcentralus.stt.speech.microsoft.com/speech/recognition/conversation</a>
Sudeste asiático	<a href="https://southeastasia.stt.speech.microsoft.com/speech/recognition/conversation">https://southeastasia.stt.speech.microsoft.com/speech/recognition/conversation</a>
Sur del Reino Unido 2	<a href="https://uksouth.stt.speech.microsoft.com/speech/recognition/conversation">https://uksouth.stt.speech.microsoft.com/speech/recognition/conversation</a>

REGION	PUNTO DE CONEXIÓN
Europa occidental	<a href="https://westeurope.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1">https://westeurope.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1</a>
Oeste de EE. UU.	<a href="https://westus.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1">https://westus.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1</a>
Oeste de EE. UU. 2	<a href="https://westus2.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1">https://westus2.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1</a>

#### NOTE

El parámetro de idioma debe anexarse a la dirección URL para evitar la recepción de errores HTTP 4xx. Por ejemplo, el idioma definido a inglés de Estados Unidos con el punto de conexión del Oeste de EE. UU. es:

<https://westus.stt.speech.microsoft.com/speech/recognition/conversation/cognitiveservices/v1?language=en-US>.

#### Texto a voz

Para obtener documentación de referencia sobre la opción de texto a voz, consulte las [API de REST de texto a voz](#).

#### Voces estándares y neuronales

Utilice esta tabla para determinar la disponibilidad de las voces estándar y neuronales por región o punto de conexión:

REGION	PUNTO DE CONEXIÓN	VOCES ESTÁNDAR	VOCES NEURONALES
Este de Australia	<a href="https://australiaeast.tts.speech.microsoft.com/cognitiveservices/v1">https://australiaeast.tts.speech.microsoft.com/cognitiveservices/v1</a>	Sí	
Centro de Canadá	<a href="https://canadacentral.tts.speech.microsoft.com/cognitiveservices/v1">https://canadacentral.tts.speech.microsoft.com/cognitiveservices/v1</a>	Sí	
Centro de EE. UU.	<a href="https://centralus.tts.speech.microsoft.com/cognitiveservices/v1">https://centralus.tts.speech.microsoft.com/cognitiveservices/v1</a>	Sin	
Asia oriental	<a href="https://eastasia.tts.speech.microsoft.com/cognitiveservices/v1">https://eastasia.tts.speech.microsoft.com/cognitiveservices/v1</a>	Sin	
Este de EE. UU.	<a href="https://eastus.tts.speech.microsoft.com/cognitiveservices/v1">https://eastus.tts.speech.microsoft.com/cognitiveservices/v1</a>	Sí	
Este de EE. UU. 2	<a href="https://eastus2.tts.speech.microsoft.com/cognitiveservices/v1">https://eastus2.tts.speech.microsoft.com/cognitiveservices/v1</a>	Sin	
Centro de Francia	<a href="https://francecentral.tts.speech.microsoft.com/cognitiveservices/v1">https://francecentral.tts.speech.microsoft.com/cognitiveservices/v1</a>	Sin	
India central	<a href="https://centralindia.tts.speech.microsoft.com/cognitiveservices/v1">https://centralindia.tts.speech.microsoft.com/cognitiveservices/v1</a>	Sí	
Este de Japón	<a href="https://japaneast.tts.speech.microsoft.com/cognitiveservices/v1">https://japaneast.tts.speech.microsoft.com/cognitiveservices/v1</a>	Sin	
Corea Central	<a href="https://koreacentral.tts.speech.microsoft.com/cognitiveservices/v1">https://koreacentral.tts.speech.microsoft.com/cognitiveservices/v1</a>	Sin	
Centro-Norte de EE. UU.	<a href="https://northcentralus.tts.speech.microsoft.com/cognitiveservices/v1">https://northcentralus.tts.speech.microsoft.com/cognitiveservices/v1</a>	Sin	
Europa del Norte	<a href="https://northeurope.tts.speech.microsoft.com/cognitiveservices/v1">https://northeurope.tts.speech.microsoft.com/cognitiveservices/v1</a>	Sin	
Centro-Sur de EE. UU.	<a href="https://southcentralus.tts.speech.microsoft.com/cognitiveservices/v1">https://southcentralus.tts.speech.microsoft.com/cognitiveservices/v1</a>	Sí	
Sudeste asiático	<a href="https://southeastasia.tts.speech.microsoft.com/cognitiveservices/v1">https://southeastasia.tts.speech.microsoft.com/cognitiveservices/v1</a>	Sí	
Sur de Reino Unido 2	<a href="https://uksouth.tts.speech.microsoft.com/cognitiveservices/v1">https://uksouth.tts.speech.microsoft.com/cognitiveservices/v1</a>	Sí	
Europa occidental	<a href="https://westeurope.tts.speech.microsoft.com/cognitiveservices/v1">https://westeurope.tts.speech.microsoft.com/cognitiveservices/v1</a>	Sí	
Oeste de EE. UU.	<a href="https://westus.tts.speech.microsoft.com/cognitiveservices/v1">https://westus.tts.speech.microsoft.com/cognitiveservices/v1</a>	Sin	
Oeste de EE. UU. 2	<a href="https://westus2.tts.speech.microsoft.com/cognitiveservices/v1">https://westus2.tts.speech.microsoft.com/cognitiveservices/v1</a>	Sí	

#### Voces personalizadas

Si ha creado una fuente de voz personalizada, use el punto de conexión que ha creado. También puede utilizar los puntos de conexión que se indican a continuación; para ello, reemplace `{deploymentId}` por el identificador de implementación para el modelo de voz.

REGION	PUNTO DE CONEXIÓN
Este de Australia	<a href="https://australiaeast.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}">https://australiaeast.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}</a>
Centro de Canadá	<a href="https://canadacentral.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}">https://canadacentral.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}</a>

REGION	PUNTO DE CONEXIÓN
Centro de EE. UU.	<a href="https://centralus.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}">https://centralus.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}</a>
Asia oriental	<a href="https://eastasia.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}">https://eastasia.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}</a>
Este de EE. UU	<a href="https://eastus.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}">https://eastus.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}</a>
Este de EE. UU. 2	<a href="https://eastus2.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}">https://eastus2.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}</a>
Centro de Francia	<a href="https://francecentral.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}">https://francecentral.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}</a>
India central	<a href="https://centralindia.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}">https://centralindia.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}</a>
Este de Japón	<a href="https://japaneast.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}">https://japaneast.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}</a>
Corea Central	<a href="https://koreacentral.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}">https://koreacentral.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}</a>
Centro-Norte de EE. UU	<a href="https://northcentralus.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}">https://northcentralus.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}</a>
Europa del Norte	<a href="https://northeurope.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}">https://northeurope.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}</a>
Centro-Sur de EE. UU	<a href="https://southcentralus.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}">https://southcentralus.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}</a>
Sudeste asiático	<a href="https://southeastasia.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}">https://southeastasia.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}</a>
Sur del Reino Unido 2	<a href="https://uksouth.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}">https://uksouth.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}</a>
Europa occidental	<a href="https://westeurope.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}">https://westeurope.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}</a>
Oeste de EE. UU.	<a href="https://westus.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}">https://westus.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}</a>
Oeste de EE. UU. 2	<a href="https://westus2.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}">https://westus2.voice.speech.microsoft.com/cognitiveservices/v1?deploymentId={deploymentId}</a>