

Contents

[Documentación sobre Translator Text](#)

[Translator Text API](#)

[Información general](#)

[¿Qué es Translator Text API?](#)

[Guías de inicio rápido](#)

[Traducir texto](#)

[Transliteración de texto](#)

[Detectar idioma](#)

[Búsqueda de traducciones de palabras](#)

[Obtener idiomas admitidos](#)

[Determinación de la longitud de la oración](#)

[Tutoriales](#)

[Creación de una aplicación de WPF para Translator Text, en C#](#)

[Creación de una aplicación de traducción de Flask, en Python](#)

[Conceptos](#)

[Personalizar y mejorar la traducción del texto](#)

[Cómo la API cuenta los caracteres](#)

[Guías de procedimientos](#)

[Suscripción en Translator Text API](#)

[Migrar a Text API V3](#)

[Agregar filtrado de palabras soeces](#)

[Recibir información de alineación de palabras](#)

[Evitar la traducción del contenido](#)

[Usar la característica de diccionario dinámico](#)

[Uso de contenedores](#)

[Instalación y ejecución de contenedores](#)

[Configuración de contenedores](#)

[Traducción detrás de firewalls](#)

[Ejemplos](#)

[C#](#)

[Java](#)

[Python](#)

[Node.js](#)

[Go](#)

[PHP](#)

[Ruby](#)

[Referencia](#)

[Referencia de Translator Text API v3](#)

[Lenguajes](#)

[Translate](#)

[Transliterar](#)

[Detect](#)

[BreakSentence](#)

[Búsqueda en diccionario](#)

[Ejemplos de diccionario](#)

[Referencia de Translator Text API v2 \(en desuso\)](#)

[Transformar texto](#)

[Cómo usar los informes de Collaborative Translation Framework \(CTF\)](#)

[Devolver las mejores traducciones](#)

[Recursos](#)

[Ejemplos de código en GitHub](#)

[Preguntas más frecuentes](#)

[Compatibilidad de idioma y región](#)

[Precios](#)

[Disponibilidad regional](#)

[Cumplimiento normativo](#)

[Límites de solicitud](#)

[Stack Overflow](#)

[Custom Translator](#)

[Información general](#)

[¿Qué es Custom Translator?](#)

Guías de inicio rápido

- Compilación de un modelo personalizado

Conceptos

- Áreas de trabajo y proyectos

- Documentos paralelos

- Diccionario

- Formatos de documento

- Alineación de frases

- Filtrado de datos

- Entrenamiento y modelo

- Puntuación BLEU

Guías de procedimientos

- Crear un proyecto

- Administrar proyectos

- Cargar un documento

- Ver detalles del documento

- Entrenamiento de un modelo

- Ver detalles de los modelos

- Ver resultados de pruebas del sistema

- Administración de la configuración

- Migración desde el centro

- Implementaciones en lenguajes no compatibles

Recursos

- Compatibilidad con lenguajes

- Glosario

- Preguntas más frecuentes

- Precios

- Cumplimiento normativo

- Stack Overflow

¿Qué es Translator Text API?

13/01/2020 • 4 minutes to read • [Edit Online](#)

Translator Text API es fácil de integrar en sus aplicaciones, sitios web, herramientas y soluciones. Permite agregar experiencias de usuario multilingüe en [más de 60 idiomas](#) y se puede usar en cualquier plataforma de hardware con cualquier sistema operativo para la traducción de texto a texto.

Translator Text API forma parte de [Azure Cognitive Services API](#), una colección de algoritmos de aprendizaje automático y de inteligencia artificial en la nube, que se pueden consumir fácilmente en los proyectos de desarrollo.

Acerca de Microsoft Translator

Microsoft Translator es un servicio de traducción automática basado en la nube. En el centro de este servicio está Translator Text API, que da servicio a diversos productos y servicios de Microsoft y que utilizan miles de empresas en todo el mundo en sus aplicaciones y flujos de trabajo para que su contenido llegue a una audiencia mundial.

La traducción de voz, con la tecnología de Translator Text API, también está disponible mediante [Microsoft Speech Service](#). Combina la funcionalidad de Translator Speech API y Custom Speech Service en un servicio totalmente personalizable y unificado. Speech Service reemplaza a Translator Speech API, que se retirará el 15 de octubre de 2019.

Compatibilidad con idiomas

Microsoft Translator proporciona compatibilidad con varios idiomas de traducción, transliteración, detección de idioma y diccionarios. Consulte [Compatibilidad con idiomas](#) para obtener una lista completa o puede acceder a la lista mediante programación con la [API REST](#).

Traducción automática neuronal de Microsoft Translator

La traducción automática neuronal (NMT) es el nuevo estándar para las traducciones automáticas de alta calidad basadas en inteligencia artificial. Este estándar reemplaza la traducción automática estadística (SMT) que alcanzó un nivel estable a mediados de 2010.

NMT proporciona mejores traducciones que SMT, no solo a la hora de puntuar la calidad de la traducción bruta, sino también porque suenan más fluidas y humanas. El motivo principal de esta fluidez es que NMT usa el contexto completo de una frase para traducir las palabras. SMT solo tomaba el contexto inmediato de unas cuantas palabras antes y después de cada palabra.

Los modelos NMT se sitúan en el centro de la API y no son visibles para los usuarios finales. La única diferencia perceptible es una mejor calidad de la traducción, en especial en los idiomas chino, japonés y árabe.

Más información sobre [el funcionamiento de NMT](#)

Personalización de lenguaje

Custom Translator, una extensión del servicio Microsoft Translator, se puede usar en combinación con Translator Text API para ayudar a personalizar el sistema de traducción neuronal y mejorar la traducción para su terminología y estilo específicos.

Con Custom Translator puede crear sistemas de traducción que administran la terminología usada en su propio negocio o sector. Luego, los sistemas de traducción personalizados se pueden integrar fácilmente en las

aplicaciones, flujos de trabajo y sitios web ya existentes, en varios tipos de dispositivos, mediante el componente normal Microsoft Translator Text API, por medio del parámetro de categoría.

Más información sobre la [personalización de idioma](#)

Pasos siguientes

- [Regístrese](#) para obtener una clave de acceso.
- [Referencia de API](#) proporciona la documentación técnica de las API.
- [Detalles de precios](#)

Inicio rápido: Uso de Translator Text API para traducir texto

13/01/2020 • 38 minutes to read • [Edit Online](#)

En este inicio rápido, aprenderá a traducir una cadena de texto de inglés a alemán, italiano, japonés y tailandés mediante la API REST Translator Text.

En esta guía de inicio rápido, se requiere una [cuenta de Azure Cognitive Services](#) con un recurso de Translator Text. Si no tiene una cuenta, puede usar la [evaluación gratuita](#) para obtener una clave de suscripción.

Requisitos previos

Esta guía de inicio rápido requiere:

- C# 7.1 o posterior
- [SDK de .NET](#)
- [Paquete NuGet de Json.NET](#)
- [Visual Studio](#), [Visual Studio Code](#) o su editor favorito de código
- Una suscripción a Azure: [cree una cuenta gratuita](#).

Instalación

Creación de un recurso de Translator Text

Los servicios de Azure Cognitive Services se representan por medio de recursos de Azure a los que se suscribe. Cree un recurso para Translator Text mediante [Azure Portal](#) o la [CLI de Azure](#) en la máquina local. También puede:

- Obtener una [clave de prueba](#) válida durante siete días de forma gratuita. Después de registrarse, estará disponible en el sitio web de Azure.
- Vea un recurso existente en [Azure Portal](#).

Después de obtener una clave del recurso o la suscripción de evaluación, cree dos [variables de entorno](#):

- `TRANSLATOR_TEXT_SUBSCRIPTION_KEY`: la clave de suscripción del recurso de Translator Text.
- `TRANSLATOR_TEXT_ENDPOINT`: el punto de conexión global para Translator Text. Mediante `https://api.cognitive.microsofttranslator.com/`.

Creación de un proyecto de .NET Core

Abra un nuevo símbolo del sistema (o una sesión de terminal) y ejecute estos comandos:

```
dotnet new console -o translate-sample
cd translate-sample
```

El primer comando hace dos cosas. Crea una nueva aplicación de consola .NET y crea un directorio denominado `translate-sample`. El segundo comando cambia al directorio del proyecto.

A continuación, deberá instalar Json.Net. En el directorio del proyecto, ejecute:

```
dotnet add package Newtonsoft.Json --version 11.0.2
```

Selección de la versión del lenguaje C#

Esta guía de inicio rápido requiere C# 7.1 o posterior. Hay algunas maneras de cambiar la versión de C# del proyecto. En esta guía le mostraremos cómo ajustar el archivo `translate-sample.csproj`. Para todas las opciones disponibles, como el cambio de idioma en Visual Studio, consulte [Selección de la versión del lenguaje C#](#).

Abra el proyecto y, después, abra `translate-sample.csproj`. Asegúrese de que `LangVersion` está establecido en 7.1 o posterior. Si no hay un grupo de propiedades para la versión del lenguaje, añada estas líneas:

```
<PropertyGroup>
  <LangVersion>7.1</LangVersion>
</PropertyGroup>
```

Incorporación de los espacios de nombres necesarios al proyecto

El comando `dotnet new console` que ejecutó anteriormente creó un proyecto que incluía `Program.cs`. En este archivo es donde deberá colocar el código de la aplicación. Abra `Program.cs` y reemplace las instrucciones `using` existentes. Estas instrucciones garantizan que tiene acceso a todos los tipos necesarios para compilar y ejecutar la aplicación de ejemplo.

```
using System;
using System.Net.Http;
using System.Text;
using System.Threading.Tasks;
// Install Newtonsoft.Json with NuGet
using Newtonsoft.Json;
```

Creación de clases para la respuesta JSON

A continuación, vamos a crear un conjunto de clases que se utilizan al deserializar la respuesta JSON devuelta por Translator Text API.

```

/// <summary>
/// The C# classes that represents the JSON returned by the Translator Text API.
/// </summary>
public class TranslationResult
{
    public DetectedLanguage DetectedLanguage { get; set; }
    public TextResult SourceText { get; set; }
    public Translation[] Translations { get; set; }
}

public class DetectedLanguage
{
    public string Language { get; set; }
    public float Score { get; set; }
}

public class TextResult
{
    public string Text { get; set; }
    public string Script { get; set; }
}

public class Translation
{
    public string Text { get; set; }
    public TextResult Transliteration { get; set; }
    public string To { get; set; }
    public Alignment Alignment { get; set; }
    public SentenceLength SentLen { get; set; }
}

public class Alignment
{
    public string Proj { get; set; }
}

public class SentenceLength
{
    public int[] SrcSentLen { get; set; }
    public int[] TransSentLen { get; set; }
}

```

Obtención de información de la suscripción a partir de las variables de entorno

Agregue las líneas siguientes a la clase `Program`. Estas líneas leen la clave de suscripción y el punto de conexión de las variables de entorno y devuelven un error si se produce algún problema.


```
private const string key_var = "TRANSLATOR_TEXT_SUBSCRIPTION_KEY";
private static readonly string subscriptionKey = Environment.GetEnvironmentVariable(key_var);

private const string endpoint_var = "TRANSLATOR_TEXT_ENDPOINT";
private static readonly string endpoint = Environment.GetEnvironmentVariable(endpoint_var);

static Program()
{
    if (null == subscriptionKey)
    {
        throw new Exception("Please set/export the environment variable: " + key_var);
    }
    if (null == endpoint)
    {
        throw new Exception("Please set/export the environment variable: " + endpoint_var);
    }
}
// The code in the next section goes here.
```

Creación de una función para traducir texto

En la clase `Program`, cree una función asincrónica denominada `TranslateTextRequest()`. Esta función toma cuatro argumentos: `subscriptionKey`, `host`, `route` y `inputText`.

```
// This sample requires C# 7.1 or later for async/await.
// Async call to the Translator Text API
static public async Task TranslateTextRequest(string subscriptionKey, string endpoint, string route, string
inputText)
{
    /*
     * The code for your call to the translation service will be added to this
     * function in the next few sections.
     */
}
```

Serialización de la solicitud de traducción

A continuación, se debe crear y serializar el objeto JSON que incluye el texto que desea traducir. Tenga en cuenta que puede pasar más de un objeto en `body`.

```
object[] body = new object[] { new { Text = inputText } };
var requestBody = JsonConvert.SerializeObject(body);
```

Creación de una instancia de cliente y realización de una solicitud

Estas líneas crean instancias de `HttpClient` y `HttpRequestMessage`:

```
using (var client = new HttpClient())
using (var request = new HttpRequestMessage())
{
    // In the next few sections you'll add code to construct the request.
}
```

Construcción de la solicitud e impresión de la respuesta

Dentro de `HttpRequestMessage`:

- Declarará el método HTTP
- Construirá el URI de solicitud
- Insertará el cuerpo de la solicitud (el objeto JSON serializado)
- Agregará los encabezados necesarios
- Realizará una solicitud asíncrona
- Imprimirá la respuesta mediante las clases que creó anteriormente.

Agregue este código a `HttpRequestMessage` :

```
// Build the request.
// Set the method to Post.
request.Method = HttpMethod.Post;
// Construct the URI and add headers.
request.RequestUri = new Uri(endpoint + route);
request.Content = new StringContent(requestBody, Encoding.UTF8, "application/json");
request.Headers.Add("Ocp-Apim-Subscription-Key", subscriptionKey);

// Send the request and get response.
HttpResponseMessage response = await client.SendAsync(request).ConfigureAwait(false);
// Read response as a string.
string result = await response.Content.ReadAsStringAsync();
// Deserialize the response using the classes created earlier.
TranslationResult[] deserializedOutput = JsonConvert.DeserializeObject<TranslationResult[]>(result);
// Iterate over the deserialized results.
foreach (TranslationResult o in deserializedOutput)
{
    // Print the detected input language and confidence score.
    Console.WriteLine("Detected input language: {0}\nConfidence score: {1}\n", o.DetectedLanguage.Language,
o.DetectedLanguage.Score);
    // Iterate over the results and print each translation.
    foreach (Translation t in o.Translations)
    {
        Console.WriteLine("Translated to {0}: {1}", t.To, t.Text);
    }
}
```

Si usa una suscripción a varios servicios de Cognitive Services, también debe incluir `Ocp-Apim-Subscription-Region` en los parámetros de la solicitud. [Más información sobre la autenticación con la suscripción a varios servicios.](#)

Colocación de todo junto

El último paso consiste en llamar a `TranslateTextRequest()` en la función `Main`. En este ejemplo, vamos a traducir al alemán (`de`), italiano (`it`), japonés (`ja`) y tailandés (`th`). Busque `static void Main(string[] args)` y reemplácelo por este código:

```
static async Task Main(string[] args)
{
    // This is our main function.
    // Output languages are defined in the route.
    // For a complete list of options, see API reference.
    // https://docs.microsoft.com/azure/cognitive-services/translator/reference/v3-0-translate
    string route = "/translate?api-version=3.0&to=de&to=it&to=ja&to=th";
    // Prompts you for text to translate. If you'd prefer, you can
    // provide a string as textToTranslate.
    Console.Write("Type the phrase you'd like to translate? ");
    string textToTranslate = Console.ReadLine();
    await TranslateTextRequest(subscriptionKey, endpoint, route, textToTranslate);
    Console.WriteLine("Press any key to continue.");
    Console.ReadKey();
}
```

Observará que en `Main`, está declarando `subscriptionKey`, `endpoint` y `route`. Además, está pidiendo al usuario que indique `Console.ReadLine()` y asignando el valor a `textToTranslate`.

Ejecutar la aplicación de ejemplo

Eso es todo, ya está listo para ejecutar la aplicación de ejemplo. Desde la línea de comandos (o sesión de terminal), vaya al directorio del proyecto y ejecute:

```
dotnet run
```

Respuesta de muestra

Después de ejecutar el ejemplo, debería ver lo siguiente impreso en el terminal:

```
Detected input language: en
Confidence score: 1

Translated to de: Hallo Welt!
Translated to it: Salve, mondo!
Translated to ja: ハローワールド!
Translated to th: สวัสดีชาวโลก!
```

Este mensaje se crea a partir del JSON sin formato, y tendrá un aspecto similar al siguiente:

```
[
  {
    "detectedLanguage": {
      "language": "en",
      "score": 1.0
    },
    "translations": [
      {
        "text": "Hallo Welt!",
        "to": "de"
      },
      {
        "text": "Salve, mondo!",
        "to": "it"
      },
      {
        "text": "ハローワールド!",
        "to": "ja"
      },
      {
        "text": "สวัสดีชาวโลก!",
        "to": "th"
      }
    ]
  }
]
```

Limpieza de recursos

Asegúrese de quitar cualquier información confidencial del código fuente de la aplicación de ejemplo como, por ejemplo, las claves de suscripción.

Pasos siguientes

Eche un vistazo a la referencia de API para comprender todo lo que puede hacer con Translator Text API.

[Referencia de API](#)

Requisitos previos

Esta guía de inicio rápido requiere:

- [JDK 7 o posterior](#)
- [Gradle](#)
- Una suscripción a Azure: [cree una cuenta gratuita](#).

Instalación

Creación de un recurso de Translator Text

Los servicios de Azure Cognitive Services se representan por medio de recursos de Azure a los que se suscribe. Cree un recurso para Translator Text mediante [Azure Portal](#) o la [CLI de Azure](#) en la máquina local. También puede:

- Obtener una [clave de prueba](#) válida durante siete días de forma gratuita. Después de registrarse, estará disponible en el sitio web de Azure.
- Vea un recurso existente en [Azure Portal](#).

Después de obtener una clave del recurso o la suscripción de evaluación, cree dos [variables de entorno](#):

- `TRANSLATOR_TEXT_SUBSCRIPTION_KEY`: la clave de suscripción del recurso de Translator Text.
- `TRANSLATOR_TEXT_ENDPOINT`: el punto de conexión global para Translator Text. Mediante `https://api.cognitive.microsofttranslator.com/`.

Inicialización de un proyecto con Gradle

Comencemos por crear un directorio de trabajo para este proyecto. Desde la línea de comandos (o terminal), ejecute este comando:

```
mkdir translator-sample
cd translator-sample
```

A continuación, va a inicializar un proyecto de Gradle. Este comando creará archivos de compilación esenciales para Gradle, el más importante, el `build.gradle.kts`, que se utiliza en tiempo de ejecución para crear y configurar la aplicación. Ejecute este comando desde el directorio de trabajo:

```
gradle init --type basic
```

Cuando se le solicite que elija un **DSL**, seleccione **Kotlin**.

Configuración del archivo de compilación

Localice `build.gradle.kts` y ábralo con el IDE o editor de texto favorito. A continuación, se copia en esta configuración de compilación:

```
plugins {
    java
    application
}
application {
    mainClassName = "Translate"
}
repositories {
    mavenCentral()
}
dependencies {
    compile("com.squareup.okhttp:okhttp:2.5.0")
    compile("com.google.code.gson:gson:2.8.5")
}
```

Tenga en cuenta que este ejemplo tiene dependencias en OkHttp para las solicitudes HTTP, y Gson para administrar y analizar JSON. Si desea obtener más información acerca de las configuraciones de compilación, consulte [Creating New Gradle Builds](#) (Creación de nuevas compilaciones de Gradle).

Creación de un archivo Java

Vamos a crear una carpeta para la aplicación de ejemplo. En el directorio de trabajo, ejecute:

```
mkdir -p src/main/java
```

A continuación, en esta carpeta, cree un archivo denominado `Translate.java`.

Importación de bibliotecas necesarias

Abra `Translate.java` y agregue las siguientes instrucciones de importación:

```
import java.io.*;
import java.net.*;
import java.util.*;
import com.google.gson.*;
import com.squareup.okhttp.*;
```

Definición de variables

En primer lugar, deberá crear una clase pública para el proyecto:

```
public class Translate {
    // All project code goes here...
}
```

Agregue estas líneas a la clase `Translate`. Primero, se lee la clave de suscripción y el punto de conexión de las variables de entorno. A continuación, observe que junto con la `api-version`, se han anexoado dos parámetros adicionales a `url`. Estos parámetros se usan para establecer las salidas de la traducción. En este ejemplo, está establecida en alemán (`de`) e italiano (`it`).

```
private static String subscriptionKey = System.getenv("TRANSLATOR_TEXT_SUBSCRIPTION_KEY");
private static String endpoint = System.getenv("TRANSLATOR_TEXT_ENDPOINT");
String url = endpoint + "/translate?api-version=3.0&to=de,it";
```

Si usa una suscripción a varios servicios de Cognitive Services, también debe incluir `Ocp-Apim-Subscription-Region` en los parámetros de la solicitud. [Más información sobre la autenticación con la suscripción a varios servicios.](#)

Creación de un cliente y compilación de una solución

Agregue esta línea a la clase `Translate` para crear una instancia de `OkHttpClient` :

```
// Instantiates the OkHttpClient.
OkHttpClient client = new OkHttpClient();
```

A continuación, vamos a crear la solicitud POST. No dude en cambiar el texto para la traducción. El texto debe escaparse.

```
// This function performs a POST request.
public String Post() throws IOException {
    MediaType mediaType = MediaType.parse("application/json");
    RequestBody body = RequestBody.create(mediaType,
        "[{\n\t\"Text\": \"Welcome to Microsoft Translator. Guess how many languages I speak!\n}]]");
    Request request = new Request.Builder()
        .url(url).post(body)
        .addHeader("Ocp-Apim-Subscription-Key", subscriptionKey)
        .addHeader("Content-type", "application/json").build();
    Response response = client.newCall(request).execute();
    return response.body().string();
}
```

Creación de una función para analizar la respuesta

Esta sencilla función analiza y embellece la respuesta JSON del servicio Translator Text.

```
// This function prettifies the json response.
public static String prettify(String json_text) {
    JsonParser parser = new JsonParser();
    JsonElement json = parser.parse(json_text);
    Gson gson = new GsonBuilder().setPrettyPrinting().create();
    return gson.toJson(json);
}
```

Colocación de todo junto

El último paso consiste en realizar una solicitud y obtener una respuesta. Agregue estas líneas al proyecto:

```
public static void main(String[] args) {
    try {
        Translate translateRequest = new Translate();
        String response = translateRequest.Post();
        System.out.println(prrettify(response));
    } catch (Exception e) {
        System.out.println(e);
    }
}
```

Ejecutar la aplicación de ejemplo

Eso es todo, ya está listo para ejecutar la aplicación de ejemplo. Desde la línea de comandos (o sesión de terminal), navegue hasta la raíz del directorio de trabajo y ejecute:

```
gradle build
```

Cuando la compilación se complete, ejecute lo siguiente:

```
gradle run
```

Respuesta de muestra

```
[
  {
    "detectedLanguage": {
      "language": "en",
      "score": 1.0
    },
    "translations": [
      {
        "text": "Willkommen bei Microsoft Translator. Erraten Sie, wie viele Sprachen ich spreche!",
        "to": "de"
      },
      {
        "text": "Benvenuti a Microsoft Translator. Indovinate quante lingue parlo!",
        "to": "it"
      }
    ]
  }
]
```

Pasos siguientes

Eche un vistazo a la referencia de API para comprender todo lo que puede hacer con Translator Text API.

[Referencia de API](#)

Requisitos previos

Esta guía de inicio rápido requiere:

- [Python 2.7.x o 3.x](#)
- Una suscripción a Azure: [cree una cuenta gratuita](#).

Instalación

Creación de un recurso de Translator Text

Los servicios de Azure Cognitive Services se representan por medio de recursos de Azure a los que se suscribe. Cree un recurso para Translator Text mediante [Azure Portal](#) o la [CLI de Azure](#) en la máquina local. También puede:

- Obtener una [clave de prueba](#) válida durante siete días de forma gratuita. Después de registrarse, estará disponible en el sitio web de Azure.
- Vea un recurso existente en [Azure Portal](#).

Después de obtener una clave del recurso o la suscripción de evaluación, cree dos [variables de entorno](#):

- `TRANSLATOR_TEXT_SUBSCRIPTION_KEY`: la clave de suscripción del recurso de Translator Text.
- `TRANSLATOR_TEXT_ENDPOINT`: el punto de conexión global para Translator Text. Mediante `https://api.cognitive.microsofttranslator.com/`.

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 `translate-text.py`. Asegúrese de que el intérprete del IDE hace referencia a la versión correcta de Python para evitar que no se reconozcan las bibliotecas.

```
# -*- coding: utf-8 -*-
import os, requests, uuid, json
```

NOTE

Si no ha usado estos módulos deberá instalarlos antes de ejecutar el programa. Para instalar estos paquetes, ejecute:

```
pip install requests uuid
```

El primer comentario le indica al intérprete de Python que debe usar la codificación UTF-8. Después, se importan los módulos necesarios para leer la clave de suscripción desde una variable de entorno, construir la solicitud HTTP, crear un identificador único y controlar la respuesta JSON que devuelve Translator Text API.

Establecimiento de la clave de suscripción, el punto de conexión y la ruta de acceso

En este ejemplo se intenta leer la clave de suscripción y el punto de conexión de Translator Text desde las variables de entorno `TRANSLATOR_TEXT_KEY` y `TRANSLATOR_TEXT_ENDPOINT`. Si no está familiarizado con las variables de entorno, puede establecer `subscription_key` y `endpoint` como cadenas y convertir en comentario las instrucciones condicionales.

Copie este código en el proyecto:

```
key_var_name = 'TRANSLATOR_TEXT_SUBSCRIPTION_KEY'
if not key_var_name in os.environ:
    raise Exception('Please set/export the environment variable: {}'.format(key_var_name))
subscription_key = os.environ[key_var_name]

endpoint_var_name = 'TRANSLATOR_TEXT_ENDPOINT'
if not endpoint_var_name in os.environ:
    raise Exception('Please set/export the environment variable: {}'.format(endpoint_var_name))
endpoint = os.environ[endpoint_var_name]
```

El punto de conexión global de Translator Text se establece como `endpoint`. `path` establece la ruta de `translate` e identifica que deseamos usar la versión 3 de la API.

Los `params` se utilizan para establecer los idiomas de salida. En este ejemplo vamos a traducir de inglés a italiano y alemán: `it` y `de`.

NOTE

Para más información sobre los puntos de conexión, las rutas y los parámetros de la solicitud, consulte [Translator Text API 3.0: Traducción](#).

```
path = '/translate?api-version=3.0'
params = '&to=de&to=it'
constructed_url = endpoint + path + params
```


Incorporación de encabezados

La manera más fácil de autenticar una solicitud es pasar la clave de suscripción como un encabezado `Ocp-Apim-Subscription-Key`, que es el que se usa en este ejemplo. O bien, puede intercambiar la clave de suscripción para un token de acceso y pasar este token como un encabezado `Authorization` para validar la solicitud. Para más información, consulte [Autenticación](#).

Copie este fragmento de código en el proyecto:

```
headers = {
    'Ocp-Apim-Subscription-Key': subscription_key,
    'Content-type': 'application/json',
    'X-ClientTraceId': str(uuid.uuid4())
}
```

Si usa una suscripción a varios servicios de Cognitive Services, también debe incluir `Ocp-Apim-Subscription-Region` en los parámetros de la solicitud. [Más información sobre la autenticación con la suscripción a varios servicios](#).

Creación de una solicitud para traducir texto

Defina la cadena (o cadenas) que desea traducir:

```
body = [{
    'text': 'Hello World!'
}]
```

A continuación, vamos a crear una solicitud POST mediante el módulo `requests`, que usa tres argumentos: la dirección URL concatenada, los encabezados de solicitud y el cuerpo de solicitud:

```
request = requests.post(constructed_url, headers=headers, json=body)
response = request.json()
```

Impresión de la respuesta

El último paso es imprimir los resultados. Este fragmento de código adorna los resultados mediante una clasificación de las claves, el establecimiento de una sangría y la declaración de separadores de elementos y de claves.

```
print(json.dumps(response, sort_keys=True, indent=4,
                  ensure_ascii=False, separators=(',', ' ')))
```

Colocación de todo junto

Eso es todo, ha creado un sencillo programa que llama a Translator Text API y devuelve una respuesta JSON. Ahora es el momento de ejecutar el programa:

```
python translate-text.py
```

Si desea comparar su código con el nuestro, el ejemplo completo está disponible en [GitHub](#).

Respuesta de muestra

```
[
  {
    "detectedLanguage": {
      "language": "en",
      "score": 1.0
    },
    "translations": [
      {
        "text": "Hallo Welt!",
        "to": "de"
      },
      {
        "text": "Salve, mondo!",
        "to": "it"
      }
    ]
  }
]
```

Limpieza de recursos

Si ha codificado de forma rígida la clave de suscripción en el programa, asegúrese de quitarla cuando haya terminado con esta guía de inicio rápido.

Pasos siguientes

Eche un vistazo a la referencia de API para comprender todo lo que puede hacer con Translator Text API.

[Referencia de API](#)

Requisitos previos

Esta guía de inicio rápido requiere:

- [Node 8.12.x o posterior](#)
- Una suscripción a Azure: [cree una cuenta gratuita](#).

Instalación

Creación de un recurso de Translator Text

Los servicios de Azure Cognitive Services se representan por medio de recursos de Azure a los que se suscribe. Cree un recurso para Translator Text mediante [Azure Portal](#) o la [CLI de Azure](#) en la máquina local. También puede:

- Obtener una [clave de prueba](#) válida durante siete días de forma gratuita. Después de registrarse, estará disponible en el sitio web de Azure.
- Vea un recurso existente en [Azure Portal](#).

Después de obtener una clave del recurso o la suscripción de evaluación, cree dos [variables de entorno](#):

- `TRANSLATOR_TEXT_SUBSCRIPTION_KEY`: la clave de suscripción del recurso de Translator Text.
- `TRANSLATOR_TEXT_ENDPOINT`: el punto de conexión global para Translator Text. Mediante `https://api.cognitive.microsofttranslator.com/`.

Creación de un proyecto e importación de los módulos necesarios

Cree un proyecto con su editor o IDE favoritos, o bien una carpeta con un archivo llamado `translate-text.js` en el escritorio. A continuación, copie este fragmento de código en el proyecto o archivo:

```
const request = require('request');
const uuidv4 = require('uuid/v4');
```

NOTE

Si no ha usado estos módulos deberá instalarlos antes de ejecutar el programa. Para instalar estos paquetes, ejecute:

```
npm install request uuidv4 .
```

Estos módulos son necesarios para construir la solicitud HTTP y crear un identificador único para el encabezado `'X-ClientTraceId'`.

Establecimiento de la clave de suscripción y el punto de conexión

En este ejemplo se intenta leer la clave de suscripción y el punto de conexión de Translator Text desde estas variables de entorno: `TRANSLATOR_TEXT_SUBSCRIPTION_KEY` y `TRANSLATOR_TEXT_ENDPOINT`. Si no está familiarizado con las variables de entorno, puede establecer `subscriptionKey` y `endpoint` como cadenas y convertir en comentario las instrucciones condicionales.

Copie este código en el proyecto:

```
var key_var = 'TRANSLATOR_TEXT_SUBSCRIPTION_KEY';
if (!process.env[key_var]) {
    throw new Error('Please set/export the following environment variable: ' + key_var);
}
var subscriptionKey = process.env[key_var];
var endpoint_var = 'TRANSLATOR_TEXT_ENDPOINT';
if (!process.env[endpoint_var]) {
    throw new Error('Please set/export the following environment variable: ' + endpoint_var);
}
var endpoint = process.env[endpoint_var];
```

Configuración de la solicitud

El método `request()`, disponible mediante el módulo de solicitud, nos permite pasar el método HTTP, la dirección URL, los parámetros de la solicitud, los encabezados y el cuerpo de JSON como un objeto `options`. En este fragmento de código, vamos a configurar la solicitud:

NOTE

Para más información sobre los puntos de conexión, las rutas y los parámetros de la solicitud, consulte [Translator Text API 3.0: Traducción](#).

```
let options = {
  method: 'POST',
  baseUrl: endpoint,
  url: 'translate',
  qs: {
    'api-version': '3.0',
    'to': ['de', 'it']
  },
  headers: {
    'Ocp-Apim-Subscription-Key': subscriptionKey,
    'Content-type': 'application/json',
    'X-ClientTraceId': uuidv4().toString()
  },
  body: [{
    'text': 'Hello World!'
  }],
  json: true,
};
```

La manera más fácil de autenticar una solicitud es pasar la clave de suscripción como un encabezado `Ocp-Apim-Subscription-Key`, que es el que se usa en este ejemplo. O bien, puede intercambiar la clave de suscripción para un token de acceso y pasar este token como un encabezado `Authorization` para validar la solicitud.

Si usa una suscripción a varios servicios de Cognitive Services, también debe incluir `Ocp-Apim-Subscription-Region` en los encabezados de la solicitud.

Para más información, consulte [Autenticación](#).

Realización de solicitud e impresión de la respuesta

A continuación, vamos a crear una solicitud mediante el método `request()`. Toma el objeto `options` que se creó en la sección anterior como el primer argumento y, a continuación, imprime la respuesta JSON embellecida.

```
request(options, function(err, res, body){
  console.log(JSON.stringify(body, null, 4));
});
```

NOTE

En este ejemplo, vamos a definir la solicitud HTTP en el objeto `options`. Sin embargo, el módulo de solicitud también admite métodos de conveniencia como `.post` y `.get`. Para más información, consulte [métodos de conveniencia](#).

Colocación de todo junto

Eso es todo, ha creado un sencillo programa que llama a Translator Text API y devuelve una respuesta JSON. Ahora es el momento de ejecutar el programa:

```
node translate-text.js
```

Si desea comparar su código con el nuestro, el ejemplo completo está disponible en [GitHub](#).

Respuesta de muestra

```
[
  {
    "detectedLanguage": {
      "language": "en",
      "score": 1.0
    },
    "translations": [
      {
        "text": "Hallo Welt!",
        "to": "de"
      },
      {
        "text": "Salve, mondo!",
        "to": "it"
      }
    ]
  }
]
```

Limpieza de recursos

Si ha codificado de forma rígida la clave de suscripción en el programa, asegúrese de quitarla cuando haya terminado con esta guía de inicio rápido.

Pasos siguientes

Eche un vistazo a la referencia de API para comprender todo lo que puede hacer con Translator Text API.

[Referencia de API](#)

Requisitos previos

Esta guía de inicio rápido requiere:

- [Go](#)
- Una suscripción a Azure: [cree una cuenta gratuita](#).

Instalación

Creación de un recurso de Translator Text

Los servicios de Azure Cognitive Services se representan por medio de recursos de Azure a los que se suscribe. Cree un recurso para Translator Text mediante [Azure Portal](#) o la [CLI de Azure](#) en la máquina local. También puede:

- Obtener una [clave de prueba](#) válida durante siete días de forma gratuita. Después de registrarse, estará disponible en el sitio web de Azure.
- Vea un recurso existente en [Azure Portal](#).

Después de obtener una clave del recurso o la suscripción de evaluación, cree dos [variables de entorno](#):

- `TRANSLATOR_TEXT_SUBSCRIPTION_KEY`: la clave de suscripción del recurso de Translator Text.
- `TRANSLATOR_TEXT_ENDPOINT`: el punto de conexión global para Translator Text. Mediante `https://api.cognitive.microsofttranslator.com/`.

Creación de un proyecto e importación de los módulos necesarios

Cree un proyecto de Go con su IDE o editor favorito. A continuación, copie este fragmento de código en un archivo llamado `translate-text.go`.

```
package main

import (
    "bytes"
    "encoding/json"
    "fmt"
    "log"
    "net/http"
    "net/url"
    "os"
)
```

Creación de la función main

En este ejemplo se intenta leer la clave de suscripción y el punto de conexión de Translator Text desde estas variables de entorno: `TRANSLATOR_TEXT_SUBSCRIPTION_KEY` y `TRANSLATOR_TEXT_ENDPOINT`. Si no está familiarizado con las variables de entorno, puede establecer `subscriptionKey` y `endpoint` como cadenas y convertir en comentario las instrucciones condicionales.

Copie este código en el proyecto:

```
func main() {
    /*
     * Read your subscription key from an env variable.
     * Please note: You can replace this code block with
     * var subscriptionKey = "YOUR_SUBSCRIPTION_KEY" if you don't
     * want to use env variables. If so, be sure to delete the "os" import.
     */
    if "" == os.Getenv("TRANSLATOR_TEXT_SUBSCRIPTION_KEY") {
        log.Fatal("Please set/export the environment variable TRANSLATOR_TEXT_SUBSCRIPTION_KEY.")
    }
    subscriptionKey := os.Getenv("TRANSLATOR_TEXT_SUBSCRIPTION_KEY")
    if "" == os.Getenv("TRANSLATOR_TEXT_ENDPOINT") {
        log.Fatal("Please set/export the environment variable TRANSLATOR_TEXT_ENDPOINT.")
    }
    endpoint := os.Getenv("TRANSLATOR_TEXT_ENDPOINT")
    uri := endpoint + "/translate?api-version=3.0"
    /*
     * This calls our breakSentence function, which we'll
     * create in the next section. It takes a single argument,
     * the subscription key.
     */
    translate(subscriptionKey, uri)
}
```

Creación de una función para traducir texto

Vamos a crear una función para traducir texto. Esta función tendrá un solo argumento, su clave de suscripción de Translator Text.

```
func translate(subscriptionKey string, uri string) {
    /*
     * In the next few sections, we'll add code to this
     * function to make a request and handle the response.
     */
}
```

A continuación, vamos a construir la dirección URL. La dirección URL se crea mediante los métodos `Parse()` y `Query()`. Observará que se han agregado los parámetros con el método `Add()`. En este ejemplo va a traducir de

inglés a alemán e italiano: `de` y `it`.

Copie este código en la función `translate`.

```
// Build the request URL. See: https://golang.org/pkg/net/url/#example_URL_Parse
u, _ := url.Parse(uri)
q := u.Query()
q.Add("to", "de")
q.Add("to", "it")
u.RawQuery = q.Encode()
```

NOTE

Para más información sobre los puntos de conexión, las rutas y los parámetros de la solicitud, consulte [Translator Text API 3.0: Traducción](#).

Creación de una estructura para el cuerpo de la solicitud

A continuación, cree una estructura anónima para el cuerpo de la solicitud y codifíquelo como JSON con

`json.Marshal()`. Agregue este código a la función `translate`.

```
// Create an anonymous struct for your request body and encode it to JSON
body := []struct {
    Text string
}{
    {Text: "Hello, world!"},
}
b, _ := json.Marshal(body)
```

Compilar la solicitud

Ahora que se ha codificado el cuerpo de la solicitud como JSON, puede crear la solicitud POST y llamar a Translator Text API.

```
// Build the HTTP POST request
req, err := http.NewRequest("POST", u.String(), bytes.NewBuffer(b))
if err != nil {
    log.Fatal(err)
}
// Add required headers to the request
req.Header.Add("Ocp-Apim-Subscription-Key", subscriptionKey)
req.Header.Add("Content-Type", "application/json")

// Call the Translator Text API
res, err := http.DefaultClient.Do(req)
if err != nil {
    log.Fatal(err)
}
```

Si usa una suscripción a varios servicios de Cognitive Services, también debe incluir `Ocp-Apim-Subscription-Region` en los parámetros de la solicitud. [Más información sobre la autenticación con la suscripción a varios servicios](#).

Control e impresión de la respuesta

Agregue este código a la función `translate` para decodificar la respuesta JSON y luego dé formato al resultado e imprímalo.

```
// Decode the JSON response
var result interface{}
if err := json.NewDecoder(res.Body).Decode(&result); err != nil {
    log.Fatal(err)
}
// Format and print the response to terminal
prettyJSON, _ := json.MarshalIndent(result, "", " ")
fmt.Printf("%s\n", prettyJSON)
```

Colocación de todo junto

Eso es todo, ha creado un sencillo programa que llama a Translator Text API y devuelve una respuesta JSON. Ahora es el momento de ejecutar el programa:

```
go run translate-text.go
```

Si desea comparar su código con el nuestro, el ejemplo completo está disponible en [GitHub](#).

Respuesta de muestra

Se devuelve una respuesta correcta en JSON, tal como se muestra en el siguiente ejemplo:

```
[
  {
    "detectedLanguage": {
      "language": "en",
      "score": 1.0
    },
    "translations": [
      {
        "text": "Hallo Welt!",
        "to": "de"
      },
      {
        "text": "Salve, mondo!",
        "to": "it"
      }
    ]
  }
]
```

Pasos siguientes

Eche un vistazo a la referencia de API para comprender todo lo que puede hacer con Translator Text API.

[Referencia de API](#)

Consulte también

- [Transliterar texto](#)
- [Identificar el idioma de entrada](#)
- [Obtener traducciones alternativas](#)
- [Obtener una lista de idiomas admitidos](#)
- [Determinar la longitud de las oraciones de una entrada](#)

Inicio rápido: Uso de Translator Text API para transliterar texto

13/01/2020 • 36 minutes to read • [Edit Online](#)

En este tutorial aprenderá a transliterar (convertir) texto de un script a otro mediante la API REST Translator Text. En el ejemplo que se proporciona, se transcribe el japonés para que use el alfabeto latino.

En esta guía de inicio rápido, se requiere una [cuenta de Azure Cognitive Services](#) con un recurso de Translator Text. Si no tiene una cuenta, puede usar la [evaluación gratuita](#) para obtener una clave de suscripción.

Requisitos previos

Esta guía de inicio rápido requiere:

- C# 7.1 o posterior
- [SDK de .NET](#)
- [Paquete NuGet de Json.NET](#)
- [Visual Studio](#), [Visual Studio Code](#) o su editor favorito de código
- Una suscripción a Azure: [cree una cuenta gratuita](#).

Instalación

Creación de un recurso de Translator Text

Los servicios de Azure Cognitive Services se representan por medio de recursos de Azure a los que se suscribe. Cree un recurso para Translator Text mediante [Azure Portal](#) o la [CLI de Azure](#) en la máquina local. También puede:

- Obtener una [clave de prueba](#) válida durante siete días de forma gratuita. Después de registrarse, estará disponible en el sitio web de Azure.
- Vea un recurso existente en [Azure Portal](#).

Después de obtener una clave del recurso o la suscripción de evaluación, cree dos [variables de entorno](#):

- `TRANSLATOR_TEXT_SUBSCRIPTION_KEY`: la clave de suscripción del recurso de Translator Text.
- `TRANSLATOR_TEXT_ENDPOINT`: el punto de conexión global para Translator Text. Mediante `https://api.cognitive.microsofttranslator.com/`.

Creación de un proyecto de .NET Core

Abra un nuevo símbolo del sistema (o una sesión de terminal) y ejecute estos comandos:

```
dotnet new console -o transliterate-sample
cd transliterate-sample
```

El primer comando hace dos cosas. Crea una nueva aplicación de consola .NET y crea un directorio denominado `transliterate-sample`. El segundo comando cambia al directorio del proyecto.

A continuación, deberá instalar Json.Net. En el directorio del proyecto, ejecute:

```
dotnet add package Newtonsoft.Json --version 11.0.2
```

Selección de la versión del lenguaje C#

Esta guía de inicio rápido requiere C# 7.1 o posterior. Hay algunas maneras de cambiar la versión de C# del proyecto. En esta guía le mostraremos cómo ajustar el archivo `transliterate-sample.csproj`. Para todas las opciones disponibles, como el cambio de idioma en Visual Studio, consulte [Selección de la versión del lenguaje C#](#).

Abra el proyecto y, después, abra `transliterate-sample.csproj`. Asegúrese de que `LangVersion` está establecido en 7.1 o posterior. Si no hay un grupo de propiedades para la versión del lenguaje, añada estas líneas:

```
<PropertyGroup>
  <LangVersion>7.1</LangVersion>
</PropertyGroup>
```

Incorporación de los espacios de nombres necesarios al proyecto

El comando `dotnet new console` que ejecutó anteriormente creó un proyecto que incluía `Program.cs`. En este archivo es donde deberá colocar el código de la aplicación. Abra `Program.cs` y reemplace las instrucciones `using` existentes. Estas instrucciones garantizan que tiene acceso a todos los tipos necesarios para compilar y ejecutar la aplicación de ejemplo.

```
using System;
using System.Net.Http;
using System.Text;
using System.Threading.Tasks;
// Install Newtonsoft.Json with NuGet
using Newtonsoft.Json;
```

Creación de clases para la respuesta JSON

A continuación, vamos a crear una clase que se utiliza al deserializar la respuesta JSON devuelta por Translator Text API.

```
/// <summary>
/// The C# classes that represents the JSON returned by the Translator Text API.
/// </summary>
public class TransliterationResult
{
    public string Text { get; set; }
    public string Script { get; set; }
}
```

Obtención de información de la suscripción a partir de las variables de entorno

Agregue las líneas siguientes a la clase `Program`. Estas líneas leen la clave de suscripción y el punto de conexión de las variables de entorno y devuelven un error si se produce algún problema.

```
private const string key_var = "TRANSLATOR_TEXT_SUBSCRIPTION_KEY";
private static readonly string subscriptionKey = Environment.GetEnvironmentVariable(key_var);

private const string endpoint_var = "TRANSLATOR_TEXT_ENDPOINT";
private static readonly string endpoint = Environment.GetEnvironmentVariable(endpoint_var);

static Program()
{
    if (null == subscriptionKey)
    {
        throw new Exception("Please set/export the environment variable: " + key_var);
    }
    if (null == endpoint)
    {
        throw new Exception("Please set/export the environment variable: " + endpoint_var);
    }
}
// The code in the next section goes here.
```

Creación de una función para transliterar texto

Dentro de la clase `Program`, cree una función asíncrona denominada `TransliterateTextRequest()`. Esta función toma cuatro argumentos: `subscriptionKey`, `endpoint`, `route` y `inputText`.

```
static public async Task TransliterateTextRequest(string subscriptionKey, string endpoint, string route,
string inputText)
{
    /*
    * The code for your call to the translation service will be added to this
    * function in the next few sections.
    */
}
```

Serialización de la solicitud de traducción

A continuación, se debe crear y serializar el objeto JSON que incluye el texto que desea traducir. Tenga en cuenta que puede pasar más de un objeto en `body`.

```
object[] body = new object[] { new { Text = inputText } };
var requestBody = JsonConvert.SerializeObject(body);
```

Creación de una instancia de cliente y realización de una solicitud

Estas líneas crean instancias de `HttpClient` y `HttpRequestMessage`:

```
using (var client = new HttpClient())
using (var request = new HttpRequestMessage())
{
    // In the next few sections you'll add code to construct the request.
}
```

Construcción de la solicitud e impresión de la respuesta

Dentro de `HttpRequestMessage`:

- Declarará el método HTTP

- Construirá el URI de solicitud
- Insertará el cuerpo de la solicitud (el objeto JSON serializado)
- Agregará los encabezados necesarios
- Realizará una solicitud asincrónica
- Impresión de la respuesta

Agregue este código a `HttpRequestMessage`:

```
// Build the request.
// Set the method to Post.
request.Method = HttpMethod.Post;
// Construct the URI and add headers.
request.RequestUri = new Uri(endpoint + route);
request.Content = new StringContent(requestBody, Encoding.UTF8, "application/json");
request.Headers.Add("Ocp-Apim-Subscription-Key", subscriptionKey);

// Send the request and get response.
HttpResponseMessage response = await client.SendAsync(request).ConfigureAwait(false);
// Read response as a string.
string result = await response.Content.ReadAsStringAsync();
// Deserialize the response using the classes created earlier.
TransliterationResult[] deserializedOutput = JsonConvert.DeserializeObject<TransliterationResult[]>(result);
// Iterate over the deserialized results.
foreach (TransliterationResult o in deserializedOutput)
{
    Console.WriteLine("Transliterated to {0} script: {1}", o.Script, o.Text);
}
```

Si usa una suscripción a varios servicios de Cognitive Services, también debe incluir

`Ocp-Apim-Subscription-Region` en los parámetros de la solicitud. [Más información sobre la autenticación con la suscripción a varios servicios.](#)

Colocación de todo junto

El último paso consiste en llamar a `TransliterateTextRequest()` en la función `Main`. En este ejemplo, vamos a transcribir de japonés al alfabeto latino. Busque `static void Main(string[] args)` y reemplácelo por este código:

```
static async Task Main(string[] args)
{
    // This is our main function.
    // Output languages are defined in the route.
    // For a complete list of options, see API reference.
    // https://docs.microsoft.com/azure/cognitive-services/translator/reference/v3-0-transliterate
    string route = "/transliterate?api-version=3.0&language=ja&fromScript=jpan&toScript=latn";
    string textToTransliterate = @"こんにちは";
    await TransliterateTextRequest(subscriptionKey, endpoint, route, textToTransliterate);
    Console.WriteLine("Press any key to continue.");
    Console.ReadKey();
}
```

Observará que en `Main`, está declarando `subscriptionKey`, `endpoint` y `route`, y el script para transcribir `textToTransliterate`.

Ejecutar la aplicación de ejemplo

Eso es todo, ya está listo para ejecutar la aplicación de ejemplo. Desde la línea de comandos (o sesión de terminal), vaya al directorio del proyecto y ejecute:

```
dotnet run
```

Respuesta de muestra

Después de ejecutar el ejemplo, debería ver lo siguiente impreso en el terminal:

```
Transliterated to latn script: Kon\'nichiwa
```

Este mensaje se crea a partir del JSON sin formato, y tendrá un aspecto similar al siguiente:

```
[
  {
    "script": "latn",
    "text": "konnichiwa"
  }
]
```

Limpieza de recursos

Asegúrese de quitar cualquier información confidencial del código fuente de la aplicación de ejemplo como, por ejemplo, las claves de suscripción.

Pasos siguientes

Eche un vistazo a la referencia de API para comprender todo lo que puede hacer con Translator Text API.

[Referencia de API](#)

Requisitos previos

Esta guía de inicio rápido requiere:

- [JDK 7 o posterior](#)
- [Gradle](#)
- Una suscripción a Azure: [cree una cuenta gratuita](#).

Instalación

Creación de un recurso de Translator Text

Los servicios de Azure Cognitive Services se representan por medio de recursos de Azure a los que se suscribe. Cree un recurso para Translator Text mediante [Azure Portal](#) o la [CLI de Azure](#) en la máquina local. También puede:

- Obtener una [clave de prueba](#) válida durante siete días de forma gratuita. Después de registrarse, estará disponible en el sitio web de Azure.
- Vea un recurso existente en [Azure Portal](#).

Después de obtener una clave del recurso o la suscripción de evaluación, cree dos [variables de entorno](#):

- `TRANSLATOR_TEXT_SUBSCRIPTION_KEY`: la clave de suscripción del recurso de Translator Text.
- `TRANSLATOR_TEXT_ENDPOINT`: el punto de conexión global para Translator Text. Mediante `https://api.cognitive.microsofttranslator.com/`.

Inicialización de un proyecto con Gradle

Comencemos por crear un directorio de trabajo para este proyecto. Desde la línea de comandos (o terminal), ejecute este comando:

```
mkdir transliterate-sample
cd transliterate-sample
```

A continuación, va a inicializar un proyecto de Gradle. Este comando creará archivos de compilación esenciales para Gradle, el más importante, el `build.gradle.kts`, que se utiliza en tiempo de ejecución para crear y configurar la aplicación. Ejecute este comando desde el directorio de trabajo:

```
gradle init --type basic
```

Cuando se le solicite que elija un **DSL**, seleccione **Kotlin**.

Configuración del archivo de compilación

Localice `build.gradle.kts` y ábralo con el IDE o editor de texto favorito. A continuación, se copia en esta configuración de compilación:

```
plugins {
    java
    application
}
application {
    mainClassName = "Transliterate"
}
repositories {
    mavenCentral()
}
dependencies {
    compile("com.squareup.okhttp:okhttp:2.5.0")
    compile("com.google.code.gson:gson:2.8.5")
}
```

Tenga en cuenta que este ejemplo tiene dependencias en OkHttp para las solicitudes HTTP, y Gson para administrar y analizar JSON. Si desea obtener más información acerca de las configuraciones de compilación, consulte [Creating New Gradle Builds](#) (Creación de nuevas compilaciones de Gradle).

Creación de un archivo Java

Vamos a crear una carpeta para la aplicación de ejemplo. En el directorio de trabajo, ejecute:

```
mkdir -p src\main\java
```

A continuación, en esta carpeta, cree un archivo denominado `Transliterate.java`.

Importación de bibliotecas necesarias

Abra `Transliterate.java` y agregue las siguientes instrucciones de importación:

```
import java.io.*;
import java.net.*;
import java.util.*;
import com.google.gson.*;
import com.squareup.okhttp.*;
```

Definición de variables

En primer lugar, deberá crear una clase pública para el proyecto:

```
public class Transliterate {
    // All project code goes here...
}
```

Agregue estas líneas a la clase `Transliterate`. Primero, se lee la clave de suscripción y el punto de conexión de las variables de entorno. A continuación, observe que junto con la `api-version`, se han anexado dos parámetros adicionales a `url`. Estos parámetros se utilizan para establecer el idioma de entrada y los scripts para la transliteración. En este ejemplo, se ha establecido en japonés (`jpan`) y latín (`latn`).

```
private static String subscriptionKey = System.getenv("TRANSLATOR_TEXT_SUBSCRIPTION_KEY");
private static String endpoint = System.getenv("TRANSLATOR_TEXT_ENDPOINT");
String url = endpoint + "/transliterate?api-version=3.0&language=ja&fromScript=jpan&toScript=latn";
```

Si usa una suscripción a varios servicios de Cognitive Services, también debe incluir

`Ocp-Apim-Subscription-Region` en los parámetros de la solicitud. [Más información sobre la autenticación con la suscripción a varios servicios.](#)

Creación de un cliente y compilación de una solución

Agregue esta línea a la clase `Transliterate` para crear una instancia de `OkHttpClient`:

```
// Instantiates the OkHttpClient.
OkHttpClient client = new OkHttpClient();
```

A continuación, vamos a crear la solicitud POST. No dude en cambiar el texto para la transliteración.

```
// This function performs a POST request.
public String Post() throws IOException {
    MediaType mediaType = MediaType.parse("application/json");
    RequestBody body = RequestBody.create(mediaType,
        "[{\n\t\t\"Text\": \"こんにちは\"\n}]");
    Request request = new Request.Builder()
        .url(url).post(body)
        .addHeader("Ocp-Apim-Subscription-Key", subscriptionKey)
        .addHeader("Content-type", "application/json").build();
    Response response = client.newCall(request).execute();
    return response.body().string();
}
```

Creación de una función para analizar la respuesta

Esta sencilla función analiza y embellece la respuesta JSON del servicio Translator Text.

```
// This function prettifies the json response.
public static String prettify(String json_text) {
    JsonParser parser = new JsonParser();
    JsonElement json = parser.parse(json_text);
    Gson gson = new GsonBuilder().setPrettyPrinting().create();
    return gson.toJson(json);
}
```

Colocación de todo junto

El último paso consiste en realizar una solicitud y obtener una respuesta. Agregue estas líneas al proyecto:

```
public static void main(String[] args) {
    try {
        Transliterate transliterateRequest = new Transliterate();
        String response = transliterateRequest.Post();
        System.out.println(prettify(response));
    } catch (Exception e) {
        System.out.println(e);
    }
}
```

Ejecutar la aplicación de ejemplo

Eso es todo, ya está listo para ejecutar la aplicación de ejemplo. Desde la línea de comandos (o sesión de terminal), navegue hasta la raíz del directorio de trabajo y ejecute:

```
gradle build
```

Cuando la compilación se complete, ejecute lo siguiente:

```
gradle run
```

Respuesta de muestra

```
[
  {
    "text": "konnichiwa",
    "script": "latn"
  }
]
```

Pasos siguientes

Eche un vistazo a la referencia de API para comprender todo lo que puede hacer con Translator Text API.

[Referencia de API](#)

Requisitos previos

Esta guía de inicio rápido requiere:

- [Python 2.7.x o 3.x](#)

- Una suscripción a Azure: [cree una cuenta gratuita](#).

Instalación

Creación de un recurso de Translator Text

Los servicios de Azure Cognitive Services se representan por medio de recursos de Azure a los que se suscribe. Cree un recurso para Translator Text mediante [Azure Portal](#) o la [CLI de Azure](#) en la máquina local. También puede:

- Obtener una [clave de prueba](#) válida durante siete días de forma gratuita. Después de registrarse, estará disponible en el sitio web de Azure.
- Vea un recurso existente en [Azure Portal](#).

Después de obtener una clave del recurso o la suscripción de evaluación, cree dos [variables de entorno](#):

- `TRANSLATOR_TEXT_SUBSCRIPTION_KEY`: la clave de suscripción del recurso de Translator Text.
- `TRANSLATOR_TEXT_ENDPOINT`: el punto de conexión global para Translator Text. Mediante `https://api.cognitive.microsofttranslator.com/`.

Creación de un proyecto e importación de los módulos necesarios

Cree un proyecto con su editor o IDE favoritos, o bien una carpeta con un archivo llamado `transliterate-text.py` en el escritorio. A continuación, copie este fragmento de código en el proyecto o archivo:

```
# -*- coding: utf-8 -*-  
import os, requests, uuid, json
```

NOTE

Si no ha usado estos módulos deberá instalarlos antes de ejecutar el programa. Para instalar estos paquetes, ejecute:

```
pip install requests uuid
```

El primer comentario le indica al intérprete de Python que debe usar la codificación UTF-8. Después, se importan los módulos necesarios para leer la clave de suscripción desde una variable de entorno, construir la solicitud HTTP, crear un identificador único y controlar la respuesta JSON que devuelve Translator Text API.

Establecimiento de la clave de suscripción, el punto de conexión y la ruta de acceso

En este ejemplo se intenta leer la clave de suscripción y el punto de conexión de Translator Text desde las variables de entorno `TRANSLATOR_TEXT_KEY` y `TRANSLATOR_TEXT_ENDPOINT`. Si no está familiarizado con las variables de entorno, puede establecer `subscription_key` y `endpoint` como cadenas y convertir en comentario las instrucciones condicionales.

Copie este código en el proyecto:

```
key_var_name = 'TRANSLATOR_TEXT_SUBSCRIPTION_KEY'
if not key_var_name in os.environ:
    raise Exception('Please set/export the environment variable: {}'.format(key_var_name))
subscription_key = os.environ[key_var_name]

endpoint_var_name = 'TRANSLATOR_TEXT_ENDPOINT'
if not endpoint_var_name in os.environ:
    raise Exception('Please set/export the environment variable: {}'.format(endpoint_var_name))
endpoint = os.environ[endpoint_var_name]
```

El punto de conexión global de Translator Text se establece como `endpoint`. `path` establece la ruta de `transliterate` e identifica que deseamos usar la versión 3 de la API.

`params` se utiliza para establecer el idioma de entrada y los scripts de entrada y salida. En este ejemplo, vamos a transcribir de japonés al alfabeto latino.

NOTE

Para más información sobre los puntos de conexión, las rutas y los parámetros de la solicitud, consulte [Translator Text API 3.0: transliteración](#).

```
path = '/transliterate?api-version=3.0'
params = '&language=ja&fromScript=jpan&toScript=latn'
constructed_url = endpoint + path + params
```

Incorporación de encabezados

La manera más fácil de autenticar una solicitud es pasar la clave de suscripción como un encabezado `Ocp-Apim-Subscription-Key`, que es el que se usa en este ejemplo. O bien, puede intercambiar la clave de suscripción para un token de acceso y pasar este token como un encabezado `Authorization` para validar la solicitud. Para más información, consulte [Autenticación](#).

Copie este fragmento de código en el proyecto:

```
headers = {
    'Ocp-Apim-Subscription-Key': subscription_key,
    'Content-type': 'application/json',
    'X-ClientTraceId': str(uuid.uuid4())
}
```

Si usa una suscripción a varios servicios de Cognitive Services, también debe incluir

`Ocp-Apim-Subscription-Region` en los parámetros de la solicitud. [Más información sobre la autenticación con la suscripción a varios servicios](#).

Creación de una solicitud para transcribir texto

Defina la cadena (o cadenas) que desea transcribir:

```
# Transliterate "good afternoon" from source Japanese.
# Note: You can pass more than one object in body.
body = [{
    'text': 'こんにちは'
}]
```

A continuación, vamos a crear una solicitud POST mediante el módulo `requests`, que usa tres argumentos: la dirección URL concatenada, los encabezados de solicitud y el cuerpo de solicitud:

```
request = requests.post(constructed_url, headers=headers, json=body)
response = request.json()
```

Impresión de la respuesta

El último paso es imprimir los resultados. Este fragmento de código adorna los resultados mediante una clasificación de las claves, el establecimiento de una sangría y la declaración de separadores de elementos y de claves.

```
print(json.dumps(response, sort_keys=True, indent=4,
                  ensure_ascii=False, separators=(',', ' ')))
```

Colocación de todo junto

Eso es todo, ha creado un sencillo programa que llama a Translator Text API y devuelve una respuesta JSON. Ahora es el momento de ejecutar el programa:

```
python transliterate-text.py
```

Si desea comparar su código con el nuestro, el ejemplo completo está disponible en [GitHub](#).

Respuesta de muestra

```
[
  {
    "script": "latn",
    "text": "konnichiwa"
  }
]
```

Limpieza de recursos

Si ha codificado de forma rígida la clave de suscripción en el programa, asegúrese de quitarla cuando haya terminado con esta guía de inicio rápido.

Pasos siguientes

Eche un vistazo a la referencia de API para comprender todo lo que puede hacer con Translator Text API.

[Referencia de API](#)

Requisitos previos

Esta guía de inicio rápido requiere:

- [Node 8.12.x o posterior](#)
- Una suscripción a Azure: [cree una cuenta gratuita](#).

Instalación

Creación de un recurso de Translator Text

Los servicios de Azure Cognitive Services se representan por medio de recursos de Azure a los que se suscribe. Cree un recurso para Translator Text mediante [Azure Portal](#) o la [CLI de Azure](#) en la máquina local. También puede:

- Obtener una [clave de prueba](#) válida durante siete días de forma gratuita. Después de registrarse, estará disponible en el sitio web de Azure.
- Vea un recurso existente en [Azure Portal](#).

Después de obtener una clave del recurso o la suscripción de evaluación, cree dos [variables de entorno](#):

- `TRANSLATOR_TEXT_SUBSCRIPTION_KEY`: la clave de suscripción del recurso de Translator Text.
- `TRANSLATOR_TEXT_ENDPOINT`: el punto de conexión global para Translator Text. Mediante `https://api.cognitive.microsofttranslator.com/`.

Creación de un proyecto e importación de los módulos necesarios

Cree un proyecto con su editor o IDE favoritos, o bien una carpeta con un archivo llamado `translate-text.js` en el escritorio. A continuación, copie este fragmento de código en el proyecto o archivo:

```
const request = require('request');
const uuidv4 = require('uuid/v4');
```

NOTE

Si no ha usado estos módulos deberá instalarlos antes de ejecutar el programa. Para instalar estos paquetes, ejecute:

```
npm install request uuidv4
```

Estos módulos son necesarios para construir la solicitud HTTP y crear un identificador único para el encabezado `'X-ClientTraceId'`.

Establecimiento de la clave de suscripción y el punto de conexión

En este ejemplo se intenta leer la clave de suscripción y el punto de conexión de Translator Text desde estas variables de entorno: `TRANSLATOR_TEXT_SUBSCRIPTION_KEY` y `TRANSLATOR_TEXT_ENDPOINT`. Si no está familiarizado con las variables de entorno, puede establecer `subscriptionKey` y `endpoint` como cadenas y convertir en comentario las instrucciones condicionales.

Copie este código en el proyecto:

```
var key_var = 'TRANSLATOR_TEXT_SUBSCRIPTION_KEY';
if (!process.env[key_var]) {
  throw new Error('Please set/export the following environment variable: ' + key_var);
}
var subscriptionKey = process.env[key_var];
var endpoint_var = 'TRANSLATOR_TEXT_ENDPOINT';
if (!process.env[endpoint_var]) {
  throw new Error('Please set/export the following environment variable: ' + endpoint_var);
}
var endpoint = process.env[endpoint_var];
```

Configuración de la solicitud

El método `request()`, disponible mediante el módulo de solicitud, nos permite pasar el método HTTP, la dirección URL, los parámetros de la solicitud, los encabezados y el cuerpo de JSON como un objeto `options`. En este fragmento de código, vamos a configurar la solicitud:

NOTE

Para más información sobre los puntos de conexión, las rutas y los parámetros de la solicitud, consulte [Translator Text API 3.0: transliteración](#).

```
let options = {
  method: 'POST',
  baseUrl: endpoint,
  url: 'transliterate',
  qs: {
    'api-version': '3.0',
    'language': 'ja',
    'fromScript': 'jpan',
    'toScript': 'latn'
  },
  headers: {
    'Ocp-Apim-Subscription-Key': subscriptionKey,
    'Content-type': 'application/json',
    'X-ClientTraceId': uuidv4().toString()
  },
  body: [{
    'text': 'こんにちは'
  }],
  json: true,
};
```

La manera más fácil de autenticar una solicitud es pasar la clave de suscripción como un encabezado `Ocp-Apim-Subscription-Key`, que es el que se usa en este ejemplo. O bien, puede intercambiar la clave de suscripción para un token de acceso y pasar este token como un encabezado `Authorization` para validar la solicitud.

Si usa una suscripción a varios servicios de Cognitive Services, también debe incluir `Ocp-Apim-Subscription-Region` en los encabezados de la solicitud.

Para más información, consulte [Autenticación](#).

Realización de solicitud e impresión de la respuesta

A continuación, vamos a crear una solicitud mediante el método `request()`. Toma el objeto `options` que se creó en la sección anterior como el primer argumento y, a continuación, imprime la respuesta JSON embellecida.

```
request(options, function(err, res, body){
  console.log(JSON.stringify(body, null, 4));
});
```

NOTE

En este ejemplo, vamos a definir la solicitud HTTP en el objeto `options`. Sin embargo, el módulo de solicitud también admite métodos de conveniencia como `.post` y `.get`. Para más información, consulte [métodos de conveniencia](#).

Colocación de todo junto

Eso es todo, ha creado un sencillo programa que llama a Translator Text API y devuelve una respuesta JSON. Ahora es el momento de ejecutar el programa:

```
node transliterate-text.js
```

Si desea comparar su código con el nuestro, el ejemplo completo está disponible en [GitHub](#).

Respuesta de muestra

```
[
  {
    "script": "latn",
    "text": "konnichiwa"
  }
]
```

Limpieza de recursos

Si ha codificado de forma rígida la clave de suscripción en el programa, asegúrese de quitarla cuando haya terminado con esta guía de inicio rápido.

Pasos siguientes

Eche un vistazo a la referencia de API para comprender todo lo que puede hacer con Translator Text API.

[Referencia de API](#)

Requisitos previos

Esta guía de inicio rápido requiere:

- [Go](#)
- Una suscripción a Azure: [cree una cuenta gratuita](#).

Instalación

Creación de un recurso de Translator Text

Los servicios de Azure Cognitive Services se representan por medio de recursos de Azure a los que se suscribe. Cree un recurso para Translator Text mediante [Azure Portal](#) o la [CLI de Azure](#) en la máquina local. También puede:

- Obtener una [clave de prueba](#) válida durante siete días de forma gratuita. Después de registrarse, estará disponible en el sitio web de Azure.
- Vea un recurso existente en [Azure Portal](#).

Después de obtener una clave del recurso o la suscripción de evaluación, cree dos [variables de entorno](#):

- `TRANSLATOR_TEXT_SUBSCRIPTION_KEY`: la clave de suscripción del recurso de Translator Text.
- `TRANSLATOR_TEXT_ENDPOINT`: el punto de conexión global para Translator Text. Mediante `https://api.cognitive.microsofttranslator.com/`.

Creación de un proyecto e importación de los módulos necesarios

Cree un proyecto de Go con su IDE o editor favorito. A continuación, copie este fragmento de código en un archivo llamado `transliterate-text.go`.

```
package main

import (
    "bytes"
    "encoding/json"
    "fmt"
    "log"
    "net/http"
    "net/url"
    "os"
)
```

Creación de la función main

En este ejemplo se intenta leer la clave de suscripción y el punto de conexión de Translator Text desde estas variables de entorno: `TRANSLATOR_TEXT_SUBSCRIPTION_KEY` y `TRANSLATOR_TEXT_ENDPOINT`. Si no está familiarizado con las variables de entorno, puede establecer `subscriptionKey` y `endpoint` como cadenas y convertir en comentario las instrucciones condicionales.

Copie este código en el proyecto:

```
func main() {
    /*
     * Read your subscription key from an env variable.
     * Please note: You can replace this code block with
     * var subscriptionKey = "YOUR_SUBSCRIPTION_KEY" if you don't
     * want to use env variables. If so, be sure to delete the "os" import.
     */
    if "" == os.Getenv("TRANSLATOR_TEXT_SUBSCRIPTION_KEY") {
        log.Fatal("Please set/export the environment variable TRANSLATOR_TEXT_SUBSCRIPTION_KEY.")
    }
    subscriptionKey := os.Getenv("TRANSLATOR_TEXT_SUBSCRIPTION_KEY")
    if "" == os.Getenv("TRANSLATOR_TEXT_ENDPOINT") {
        log.Fatal("Please set/export the environment variable TRANSLATOR_TEXT_ENDPOINT.")
    }
    endpoint := os.Getenv("TRANSLATOR_TEXT_ENDPOINT")
    uri := endpoint + "/transliterate?api-version=3.0"
    /*
     * This calls our breakSentence function, which we'll
     * create in the next section. It takes a single argument,
     * the subscription key.
     */
    transliterate(subscriptionKey, uri)
}
```

Creación de una función para transliterar texto

Vamos a crear una función para transliterar texto. Esta función tendrá un solo argumento, su clave de suscripción de Translator Text.

```
func transliterate(subscriptionKey string, uri string) {
    /*
     * In the next few sections, we'll add code to this
     * function to make a request and handle the response.
     */
}
```

A continuación, vamos a construir la dirección URL. La dirección URL se crea mediante los métodos `Parse()` y `Query()`. Observará que se han agregado los parámetros con el método `Add()`. En este ejemplo, vamos a

transcribir de japonés al alfabeto latino.

Copie este código en la función `transliterate`.

```
// Build the request URL. See: https://golang.org/pkg/net/url/#example_URL_Parse
u, _ := url.Parse(uri)
q := u.Query()
q.Add("language", "ja")
q.Add("fromScript", "jpan")
q.Add("toScript", "latn")
u.RawQuery = q.Encode()
```

NOTE

Para más información sobre los puntos de conexión, las rutas y los parámetros de la solicitud, consulte [Translator Text API 3.0: transliteración](#).

Creación de una estructura para el cuerpo de la solicitud

A continuación, cree una estructura anónima para el cuerpo de la solicitud y codifíquelo como JSON con

`json.Marshal()`. Agregue este código a la función `transliterate`.

```
// Create an anonymous struct for your request body and encode it to JSON
body := []struct {
    Text string
}{
    {Text: "こんにちは"},
}
b, _ := json.Marshal(body)
```

Compilar la solicitud

Ahora que se ha codificado el cuerpo de la solicitud como JSON, puede crear la solicitud POST y llamar a Translator Text API.

```
// Build the HTTP POST request
req, err := http.NewRequest("POST", u.String(), bytes.NewBuffer(b))
if err != nil {
    log.Fatal(err)
}
// Add required headers to the request
req.Header.Add("Ocp-Apim-Subscription-Key", subscriptionKey)
req.Header.Add("Content-Type", "application/json")

// Call the Translator Text API
res, err := http.DefaultClient.Do(req)
if err != nil {
    log.Fatal(err)
}
```

Si usa una suscripción a varios servicios de Cognitive Services, también debe incluir

`Ocp-Apim-Subscription-Region` en los parámetros de la solicitud. [Más información sobre la autenticación con la suscripción a varios servicios](#).

Control e impresión de la respuesta

Agregue este código a la función `transliterate` para descodificar la respuesta JSON y luego dé formato al resultado e imprímalo.

```
// Decode the JSON response
var result interface{}
if err := json.NewDecoder(res.Body).Decode(&result); err != nil {
    log.Fatal(err)
}
// Format and print the response to terminal
prettyJSON, _ := json.MarshalIndent(result, "", " ")
fmt.Printf("%s\n", prettyJSON)
```

Colocación de todo junto

Eso es todo, ha creado un sencillo programa que llama a Translator Text API y devuelve una respuesta JSON. Ahora es el momento de ejecutar el programa:

```
go run transliterate-text.go
```

Si desea comparar su código con el nuestro, el ejemplo completo está disponible en [GitHub](#).

Respuesta de muestra

```
[
  {
    "script": "latn",
    "text": "konnichiwa"
  }
]
```

Pasos siguientes

Eche un vistazo a la referencia de API para comprender todo lo que puede hacer con Translator Text API.

[Referencia de API](#)

Consulte también

- [Traducir texto](#)
- [Identificar el idioma de entrada](#)
- [Obtener traducciones alternativas](#)
- [Obtener una lista de idiomas admitidos](#)
- [Determinar la longitud de las oraciones de una entrada](#)

Inicio rápido: Uso de la API Translator Text para detectar el idioma del texto

13/01/2020 • 36 minutes to read • [Edit Online](#)

En este inicio rápido, obtendrá información sobre cómo detectar el idioma del texto proporcionado mediante la API REST Translator Text.

En esta guía de inicio rápido, se requiere una [cuenta de Azure Cognitive Services](#) con un recurso de Translator Text. Si no tiene una cuenta, puede usar la [evaluación gratuita](#) para obtener una clave de suscripción.

Requisitos previos

Esta guía de inicio rápido requiere:

- C# 7.1 o posterior
- [SDK de .NET](#)
- [Paquete NuGet de Json.NET](#)
- [Visual Studio](#), [Visual Studio Code](#) o su editor favorito de código
- Una suscripción a Azure: [cree una cuenta gratuita](#).

Instalación

Creación de un recurso de Translator Text

Los servicios de Azure Cognitive Services se representan por medio de recursos de Azure a los que se suscribe. Cree un recurso para Translator Text mediante [Azure Portal](#) o la [CLI de Azure](#) en la máquina local. También puede:

- Obtener una [clave de prueba](#) válida durante siete días de forma gratuita. Después de registrarse, estará disponible en el sitio web de Azure.
- Vea un recurso existente en [Azure Portal](#).

Después de obtener una clave del recurso o la suscripción de evaluación, cree dos [variables de entorno](#):

- `TRANSLATOR_TEXT_SUBSCRIPTION_KEY`: la clave de suscripción del recurso de Translator Text.
- `TRANSLATOR_TEXT_ENDPOINT`: el punto de conexión global para Translator Text. Mediante `https://api.cognitive.microsofttranslator.com/`.

Creación de un proyecto de .NET Core

Abra un nuevo símbolo del sistema (o una sesión de terminal) y ejecute estos comandos:

```
dotnet new console -o detect-sample
cd detect-sample
```

El primer comando hace dos cosas. Crea una nueva aplicación de consola .NET y crea un directorio denominado `detect-sample`. El segundo comando cambia al directorio del proyecto.

A continuación, deberá instalar Json.Net. En el directorio del proyecto, ejecute:

```
dotnet add package Newtonsoft.Json --version 11.0.2
```

Selección de la versión del lenguaje C#

Esta guía de inicio rápido requiere C# 7.1 o posterior. Hay algunas maneras de cambiar la versión de C# del proyecto. En esta guía le mostraremos cómo ajustar el archivo `detect-sample.csproj`. Para todas las opciones disponibles, como el cambio de idioma en Visual Studio, consulte [Selección de la versión del lenguaje C#](#).

Abra el proyecto y, después, abra `detect-sample.csproj`. Asegúrese de que `LangVersion` está establecido en 7.1 o posterior. Si no hay un grupo de propiedades para la versión del lenguaje, añada estas líneas:

```
<PropertyGroup>
  <LangVersion>7.1</LangVersion>
</PropertyGroup>
```

Incorporación de los espacios de nombres necesarios al proyecto

El comando `dotnet new console` que ejecutó anteriormente creó un proyecto que incluía `Program.cs`. En este archivo es donde deberá colocar el código de la aplicación. Abra `Program.cs` y reemplace las instrucciones `using` existentes. Estas instrucciones garantizan que tiene acceso a todos los tipos necesarios para compilar y ejecutar la aplicación de ejemplo.

```
using System;
using System.Net.Http;
using System.Text;
using System.Threading.Tasks;
// Install Newtonsoft.Json with NuGet
using Newtonsoft.Json;
```

Creación de clases para la respuesta JSON

A continuación, vamos a crear una clase que se utiliza al deserializar la respuesta JSON devuelta por Translator Text API.

```
/// <summary>
/// The C# classes that represents the JSON returned by the Translator Text API.
/// </summary>
public class DetectResult
{
    public string Language { get; set; }
    public float Score { get; set; }
    public bool IsTranslationSupported { get; set; }
    public bool IsTransliterationSupported { get; set; }
    public AltTranslations[] Alternatives { get; set; }
}
public class AltTranslations
{
    public string Language { get; set; }
    public float Score { get; set; }
    public bool IsTranslationSupported { get; set; }
    public bool IsTransliterationSupported { get; set; }
}
```

Obtención de información de la suscripción a partir de las variables de

entorno

Agregue las líneas siguientes a la clase `Program`. Estas líneas leen la clave de suscripción y el punto de conexión de las variables de entorno y devuelven un error si se produce algún problema.

```
private const string key_var = "TRANSLATOR_TEXT_SUBSCRIPTION_KEY";
private static readonly string subscriptionKey = Environment.GetEnvironmentVariable(key_var);

private const string endpoint_var = "TRANSLATOR_TEXT_ENDPOINT";
private static readonly string endpoint = Environment.GetEnvironmentVariable(endpoint_var);

static Program()
{
    if (null == subscriptionKey)
    {
        throw new Exception("Please set/export the environment variable: " + key_var);
    }
    if (null == endpoint)
    {
        throw new Exception("Please set/export the environment variable: " + endpoint_var);
    }
}
// The code in the next section goes here.
```

Creación de una función para detectar el idioma del texto de origen

Dentro de la clase `Program`, cree una función denominada `DetectTextRequest()`. Esta clase encapsula el código que se usa para llamar al recurso Detect e imprime el resultado en la consola.

```
static public async Task DetectTextRequest(string subscriptionKey, string endpoint, string route, string
inputText)
{
    /*
     * The code for your call to the translation service will be added to this
     * function in the next few sections.
     */
}
```

Serialización de la solicitud de detección

A continuación, se debe crear y serializar el objeto JSON que incluye el texto en el que desea detectar el idioma.

```
System.Object[] body = new System.Object[] { new { Text = inputText } };
var requestBody = JsonConvert.SerializeObject(body);
```

Creación de una instancia de cliente y realización de una solicitud

Estas líneas crean instancias de `HttpClient` y `HttpRequestMessage`:

```
using (var client = new HttpClient())
using (var request = new HttpRequestMessage())
{
    // In the next few sections you'll add code to construct the request.
}
```

Construcción de la solicitud e impresión de la respuesta

Dentro de `HttpRequestMessage`:

- Declarará el método HTTP
- Construirá el URI de solicitud
- Insertará el cuerpo de la solicitud (el objeto JSON serializado)
- Agregará los encabezados necesarios
- Realizará una solicitud asíncrona
- Impresión de la respuesta

Agregue este código a `HttpRequestMessage`:

```
// Build the request.
request.Method = HttpMethod.Post;
// Construct the URI and add headers.
request.RequestUri = new Uri(endpoint + route);
request.Content = new StringContent(requestBody, Encoding.UTF8, "application/json");
request.Headers.Add("Ocp-Apim-Subscription-Key", subscriptionKey);

// Send the request and get response.
HttpResponseMessage response = await client.SendAsync(request).ConfigureAwait(false);
// Read response as a string.
string result = await response.Content.ReadAsStringAsync();
// Deserialize the response using the classes created earlier.
DetectResult[] deserializedOutput = JsonConvert.DeserializeObject<DetectResult[]>(result);
// Iterate over the deserialized response.
foreach (DetectResult o in deserializedOutput)
{
    Console.WriteLine("The detected language is '{0}'. Confidence is: {1}.\nTranslation supported: {2}.\nTransliteration supported: {3}.\n",
        o.Language, o.Score, o.IsTranslationSupported, o.IsTransliterationSupported);
    // Create a counter
    int counter = 0;
    // Iterate over alternate translations.
    foreach (AltTranslations a in o.Alternatives)
    {
        counter++;
        Console.WriteLine("Alternative {0}", counter);
        Console.WriteLine("The detected language is '{0}'. Confidence is: {1}.\nTranslation supported: {2}.\nTransliteration supported: {3}.\n",
            a.Language, a.Score, a.IsTranslationSupported, a.IsTransliterationSupported);
    }
}
```

Si usa una suscripción a varios servicios de Cognitive Services, también debe incluir

`Ocp-Apim-Subscription-Region` en los parámetros de la solicitud. [Más información sobre la autenticación con la suscripción a varios servicios.](#)

Colocación de todo junto

El último paso consiste en llamar a `DetectTextRequest()` en la función `Main`. Busque `static void Main(string[] args)` y reemplázelo por este código:

```
static async Task Main(string[] args)
{
    // This is our main function.
    // Output languages are defined in the route.
    // For a complete list of options, see API reference.
    string route = "/detect?api-version=3.0";
    string detectSentenceText = @"How are you doing today? The weather is pretty pleasant. Have you been to
the movies lately?";
    await DetectTextRequest(subscriptionKey, endpoint, route, detectSentenceText);
    Console.WriteLine("Press any key to continue.");
    Console.ReadKey();
}
```

Ejecutar la aplicación de ejemplo

Eso es todo, ya está listo para ejecutar la aplicación de ejemplo. Desde la línea de comandos (o sesión de terminal), vaya al directorio del proyecto y ejecute:

```
dotnet run
```

Respuesta de muestra

Después de ejecutar el ejemplo, debería ver lo siguiente impreso en el terminal:

NOTE

Busque la abreviatura del país o región en esta [lista de idiomas](#).

```
The detected language is 'en'. Confidence is: 1.
Translation supported: True.
Transliteration supported: False.
```

Alternative 1

```
The detected language is 'fil'. Confidence is: 0.82.
Translation supported: True.
Transliteration supported: False.
```

Alternative 2

```
The detected language is 'ro'. Confidence is: 1.
Translation supported: True.
Transliteration supported: False.
```

Este mensaje se crea a partir del JSON sin formato, y tendrá un aspecto similar al siguiente:

```
[
  {
    "language": "en",
    "score": 1.0,
    "isTranslationSupported": true,
    "isTransliterationSupported": false,
    "alternatives": [
      {
        "language": "fil",
        "score": 0.82,
        "isTranslationSupported": true,
        "isTransliterationSupported": false
      },
      {
        "language": "ro",
        "score": 1.0,
        "isTranslationSupported": true,
        "isTransliterationSupported": false
      }
    ]
  }
]
```

Limpieza de recursos

Asegúrese de quitar cualquier información confidencial del código fuente de la aplicación de ejemplo como, por ejemplo, las claves de suscripción.

Pasos siguientes

Eche un vistazo a la referencia de API para comprender todo lo que puede hacer con Translator Text API.

[Referencia de API](#)

Requisitos previos

Esta guía de inicio rápido requiere:

- [JDK 7 o posterior](#)
- [Gradle](#)
- Una suscripción a Azure: [cree una cuenta gratuita](#).

Instalación

Creación de un recurso de Translator Text

Los servicios de Azure Cognitive Services se representan por medio de recursos de Azure a los que se suscribe. Cree un recurso para Translator Text mediante [Azure Portal](#) o la [CLI de Azure](#) en la máquina local. También puede:

- Obtener una [clave de prueba](#) válida durante siete días de forma gratuita. Después de registrarse, estará disponible en el sitio web de Azure.
- Vea un recurso existente en [Azure Portal](#).

Después de obtener una clave del recurso o la suscripción de evaluación, cree dos [variables de entorno](#):

- `TRANSLATOR_TEXT_SUBSCRIPTION_KEY`: la clave de suscripción del recurso de Translator Text.
- `TRANSLATOR_TEXT_ENDPOINT`: el punto de conexión global para Translator Text. Mediante `https://api.cognitive.microsofttranslator.com/`.

Inicialización de un proyecto con Gradle

Comencemos por crear un directorio de trabajo para este proyecto. Desde la línea de comandos (o terminal), ejecute este comando:

```
mkdir detect-sample
cd detect-sample
```

A continuación, va a inicializar un proyecto de Gradle. Este comando creará archivos de compilación esenciales para Gradle, el más importante, el `build.gradle.kts`, que se utiliza en tiempo de ejecución para crear y configurar la aplicación. Ejecute este comando desde el directorio de trabajo:

```
gradle init --type basic
```

Cuando se le solicite que elija un **DSL**, seleccione **Kotlin**.

Configuración del archivo de compilación

Localice `build.gradle.kts` y ábralo con el IDE o editor de texto favorito. A continuación, se copia en esta configuración de compilación:

```
plugins {
    java
    application
}
application {
    mainClassName = "Detect"
}
repositories {
    mavenCentral()
}
dependencies {
    compile("com.squareup.okhttp:okhttp:2.5.0")
    compile("com.google.code.gson:gson:2.8.5")
}
```

Tenga en cuenta que este ejemplo tiene dependencias en OkHttp para las solicitudes HTTP, y Gson para administrar y analizar JSON. Si desea obtener más información acerca de las configuraciones de compilación, consulte [Creating New Gradle Builds](#) (Creación de nuevas compilaciones de Gradle).

Creación de un archivo Java

Vamos a crear una carpeta para la aplicación de ejemplo. En el directorio de trabajo, ejecute:

```
mkdir -p src/main/java
```

A continuación, en esta carpeta, cree un archivo denominado `Detect.java`.

Importación de bibliotecas necesarias

Abra `Detect.java` y agregue las siguientes instrucciones de importación:


```
import java.io.*;
import java.net.*;
import java.util.*;
import com.google.gson.*;
import com.squareup.okhttp.*;
```

Definición de variables

En primer lugar, deberá crear una clase pública para el proyecto:

```
public class Detect {
    // All project code goes here...
}
```

Agregue estas líneas a la clase `Detect`. Primero, se lee la clave de suscripción y el punto de conexión de las variables de entorno:

```
private static String subscriptionKey = System.getenv("TRANSLATOR_TEXT_SUBSCRIPTION_KEY");
private static String endpoint = System.getenv("TRANSLATOR_TEXT_ENDPOINT");
String url = endpoint + "/detect?api-version=3.0";
```

Si usa una suscripción a varios servicios de Cognitive Services, también debe incluir

`Ocp-Apim-Subscription-Region` en los parámetros de la solicitud. [Más información sobre la autenticación con la suscripción a varios servicios.](#)

Creación de un cliente y compilación de una solución

Agregue esta línea a la clase `Detect` para crear una instancia de `OkHttpClient`:

```
// Instantiates the OkHttpClient.
OkHttpClient client = new OkHttpClient();
```

A continuación, vamos a crear la solicitud POST. Si lo desea, cambie el texto para la detección de idioma.

```
// This function performs a POST request.
public String Post() throws IOException {
    MediaType mediaType = MediaType.parse("application/json");
    RequestBody body = RequestBody.create(mediaType,
        "[{\n\t\"Text\": \"Salve mondo!\n}]");
    Request request = new Request.Builder()
        .url(url).post(body)
        .addHeader("Ocp-Apim-Subscription-Key", subscriptionKey)
        .addHeader("Content-type", "application/json").build();
    Response response = client.newCall(request).execute();
    return response.body().string();
}
```

Creación de una función para analizar la respuesta

Esta sencilla función analiza y embellece la respuesta JSON del servicio Translator Text.

```
// This function prettifies the json response.
public static String prettify(String json_text) {
    JsonParser parser = new JsonParser();
    JsonElement json = parser.parse(json_text);
    Gson gson = new GsonBuilder().setPrettyPrinting().create();
    return gson.toJson(json);
}
```

Colocación de todo junto

El último paso consiste en realizar una solicitud y obtener una respuesta. Agregue estas líneas al proyecto:

```
public static void main(String[] args) {
    try {
        Detect detectRequest = new Detect();
        String response = detectRequest.Post();
        System.out.println(prettify(response));
    } catch (Exception e) {
        System.out.println(e);
    }
}
```

Ejecutar la aplicación de ejemplo

Eso es todo, ya está listo para ejecutar la aplicación de ejemplo. Desde la línea de comandos (o sesión de terminal), navegue hasta la raíz del directorio de trabajo y ejecute:

```
gradle build
```

Cuando la compilación se complete, ejecute lo siguiente:

```
gradle run
```

Respuesta de muestra

Después de ejecutar el ejemplo, debería ver lo siguiente impreso en el terminal:

NOTE

Busque la abreviatura del país o región en esta [lista de idiomas](#).

```
[
  {
    "language": "it",
    "score": 1.0,
    "isTranslationSupported": true,
    "isTransliterationSupported": false,
    "alternatives": [
      {
        "language": "pt",
        "score": 1.0,
        "isTranslationSupported": true,
        "isTransliterationSupported": false
      },
      {
        "language": "en",
        "score": 1.0,
        "isTranslationSupported": true,
        "isTransliterationSupported": false
      }
    ]
  }
]
```

Pasos siguientes

Eche un vistazo a la referencia de API para comprender todo lo que puede hacer con Translator Text API.

[Referencia de API](#)

Requisitos previos

Esta guía de inicio rápido requiere:

- [Python 2.7.x o 3.x](#)
- Una suscripción a Azure: [cree una cuenta gratuita](#).

Instalación

Creación de un recurso de Translator Text

Los servicios de Azure Cognitive Services se representan por medio de recursos de Azure a los que se suscribe.

Cree un recurso para Translator Text mediante [Azure Portal](#) o la [CLI de Azure](#) en la máquina local. También puede:

- Obtener una [clave de prueba](#) válida durante siete días de forma gratuita. Después de registrarse, estará disponible en el sitio web de Azure.
- Vea un recurso existente en [Azure Portal](#).

Después de obtener una clave del recurso o la suscripción de evaluación, cree dos [variables de entorno](#):

- `TRANSLATOR_TEXT_SUBSCRIPTION_KEY`: la clave de suscripción del recurso de Translator Text.
- `TRANSLATOR_TEXT_ENDPOINT`: el punto de conexión global para Translator Text. Mediante `https://api.cognitive.microsofttranslator.com/`.

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 `detect.py`.

```
# -*- coding: utf-8 -*-
import os, requests, uuid, json
```

NOTE

Si no ha usado estos módulos deberá instalarlos antes de ejecutar el programa. Para instalar estos paquetes, ejecute:

```
pip install requests uuid .
```

El primer comentario le indica al intérprete de Python que debe usar la codificación UTF-8. Después, se importan los módulos necesarios para leer la clave de suscripción desde una variable de entorno, construir la solicitud HTTP, crear un identificador único y controlar la respuesta JSON que devuelve Translator Text API.

Establecimiento de la clave de suscripción, el punto de conexión y la ruta de acceso

En este ejemplo se intenta leer la clave de suscripción y el punto de conexión de Translator Text desde las variables de entorno `TRANSLATOR_TEXT_KEY` y `TRANSLATOR_TEXT_ENDPOINT`. Si no está familiarizado con las variables de entorno, puede establecer `subscription_key` y `endpoint` como cadenas y convertir en comentario las instrucciones condicionales.

Copie este código en el proyecto:

```
key_var_name = 'TRANSLATOR_TEXT_SUBSCRIPTION_KEY'
if not key_var_name in os.environ:
    raise Exception('Please set/export the environment variable: {}'.format(key_var_name))
subscription_key = os.environ[key_var_name]

endpoint_var_name = 'TRANSLATOR_TEXT_ENDPOINT'
if not endpoint_var_name in os.environ:
    raise Exception('Please set/export the environment variable: {}'.format(endpoint_var_name))
endpoint = os.environ[endpoint_var_name]
```

El punto de conexión global de Translator Text se establece como `endpoint`. `path` establece la ruta de `detect` e identifica que deseamos usar la versión 3 de la API.

NOTE

Para más información sobre los puntos de conexión, las rutas y los parámetros de la solicitud, consulte [Translator Text API 3.0: Detección](#).

```
path = '/detect?api-version=3.0'
constructed_url = endpoint + path
```

Incorporación de encabezados

La manera más fácil de autenticar una solicitud es pasar la clave de suscripción como un encabezado `Ocp-Apim-Subscription-Key`, que es el que se usa en este ejemplo. O bien, puede intercambiar la clave de suscripción para un token de acceso y pasar este token como un encabezado `Authorization` para validar la solicitud. Para más información, consulte [Autenticación](#).

Copie este fragmento de código en el proyecto:

```
headers = {
    'Ocp-Apim-Subscription-Key': subscription_key,
    'Content-type': 'application/json',
    'X-ClientTraceId': str(uuid.uuid4())
}
```

Si usa una suscripción a varios servicios de Cognitive Services, también debe incluir

`Ocp-Apim-Subscription-Region` en los parámetros de la solicitud. [Más información sobre la autenticación con la suscripción a varios servicios.](#)

Creación de una solicitud para detectar el idioma del texto

Defina la cadena (o cadenas) para las que desea detectar el idioma:

```
# You can pass more than one object in body.
body = [{
    'text': 'Salve, mondo!'
}]
```

A continuación, vamos a crear una solicitud POST mediante el módulo `requests`, que usa tres argumentos: la dirección URL concatenada, los encabezados de solicitud y el cuerpo de solicitud:

```
request = requests.post(constructed_url, headers=headers, json=body)
response = request.json()
```

Impresión de la respuesta

El último paso es imprimir los resultados. Este fragmento de código adorna los resultados mediante una clasificación de las claves, el establecimiento de una sangría y la declaración de separadores de elementos y de claves.

```
print(json.dumps(response, sort_keys=True, indent=4,
                  ensure_ascii=False, separators=(',', ' ')))
```

Colocación de todo junto

Eso es todo, ha creado un sencillo programa que llama a Translator Text API y devuelve una respuesta JSON. Ahora es el momento de ejecutar el programa:

```
python detect.py
```

Si desea comparar su código con el nuestro, el ejemplo completo está disponible en [GitHub](#).

Respuesta de muestra

Después de ejecutar el ejemplo, debería ver lo siguiente impreso en el terminal:

NOTE

Busque la abreviatura del país o región en esta [lista de idiomas](#).

```
[
  {
    "alternatives": [
      {
        "isTranslationSupported": true,
        "isTransliterationSupported": false,
        "language": "pt",
        "score": 1.0
      },
      {
        "isTranslationSupported": true,
        "isTransliterationSupported": false,
        "language": "en",
        "score": 1.0
      }
    ],
    "isTranslationSupported": true,
    "isTransliterationSupported": false,
    "language": "it",
    "score": 1.0
  }
]
```

Limpieza de recursos

Si ha codificado de forma rígida la clave de suscripción en el programa, asegúrese de quitarla cuando haya terminado con esta guía de inicio rápido.

Pasos siguientes

Eche un vistazo a la referencia de API para comprender todo lo que puede hacer con Translator Text API.

[Referencia de API](#)

Requisitos previos

Esta guía de inicio rápido requiere:

- [Node 8.12.x o posterior](#)
- Una suscripción a Azure: [cree una cuenta gratuita](#).

Instalación

Creación de un recurso de Translator Text

Los servicios de Azure Cognitive Services se representan por medio de recursos de Azure a los que se suscribe.

Cree un recurso para Translator Text mediante [Azure Portal](#) o la [CLI de Azure](#) en la máquina local. También puede:

- Obtener una [clave de prueba](#) válida durante siete días de forma gratuita. Después de registrarse, estará disponible en el sitio web de Azure.
- Vea un recurso existente en [Azure Portal](#).

Después de obtener una clave del recurso o la suscripción de evaluación, cree dos [variables de entorno](#):

- `TRANSLATOR_TEXT_SUBSCRIPTION_KEY`: la clave de suscripción del recurso de Translator Text.
- `TRANSLATOR_TEXT_ENDPOINT`: el punto de conexión global para Translator Text. Mediante `https://api.cognitive.microsofttranslator.com/`.

Creación de un proyecto e importación de los módulos necesarios

Cree un proyecto con su IDE o editor favorito. A continuación, copie este fragmento de código en un archivo llamado `detect.js`.

```
const request = require('request');
const uuidv4 = require('uuid/v4');
```

NOTE

Si no ha usado estos módulos deberá instalarlos antes de ejecutar el programa. Para instalar estos paquetes, ejecute:

```
npm install request uuidv4.
```

Estos módulos son necesarios para construir la solicitud HTTP y crear un identificador único para el encabezado `'X-ClientTraceId'`.

Establecimiento de la clave de suscripción y el punto de conexión

En este ejemplo se intenta leer la clave de suscripción y el punto de conexión de Translator Text desde estas variables de entorno: `TRANSLATOR_TEXT_SUBSCRIPTION_KEY` y `TRANSLATOR_TEXT_ENDPOINT`. Si no está familiarizado con las variables de entorno, puede establecer `subscriptionKey` y `endpoint` como cadenas y convertir en comentario las instrucciones condicionales.

Copie este código en el proyecto:

```
var key_var = 'TRANSLATOR_TEXT_SUBSCRIPTION_KEY';
if (!process.env[key_var]) {
    throw new Error('Please set/export the following environment variable: ' + key_var);
}
var subscriptionKey = process.env[key_var];
var endpoint_var = 'TRANSLATOR_TEXT_ENDPOINT';
if (!process.env[endpoint_var]) {
    throw new Error('Please set/export the following environment variable: ' + endpoint_var);
}
var endpoint = process.env[endpoint_var];
```

Configuración de la solicitud

El método `request()`, disponible mediante el módulo de solicitud, nos permite pasar el método HTTP, la dirección URL, los parámetros de la solicitud, los encabezados y el cuerpo de JSON como un objeto `options`. En este fragmento de código, vamos a configurar la solicitud:

```
let options = {
  method: 'POST',
  baseUrl: endpoint,
  url: 'detect',
  qs: {
    'api-version': '3.0',
  },
  headers: {
    'Ocp-Apim-Subscription-Key': subscriptionKey,
    'Content-type': 'application/json',
    'X-ClientTraceId': uuidv4().toString()
  },
  body: [{
    'text': 'Salve, mondo!'
  }],
  json: true,
};
```

La manera más fácil de autenticar una solicitud es pasar la clave de suscripción como un encabezado `Ocp-Apim-Subscription-Key`, que es el que se usa en este ejemplo. O bien, puede intercambiar la clave de suscripción para un token de acceso y pasar este token como un encabezado `Authorization` para validar la solicitud.

Si usa una suscripción a varios servicios de Cognitive Services, también debe incluir `Ocp-Apim-Subscription-Region` en los encabezados de la solicitud.

Para más información, consulte [Autenticación](#).

Realización de solicitud e impresión de la respuesta

A continuación, vamos a crear una solicitud mediante el método `request()`. Toma el objeto `options` que se creó en la sección anterior como el primer argumento y, a continuación, imprime la respuesta JSON embellecida.

```
request(options, function(err, res, body){
  console.log(JSON.stringify(body, null, 4));
});
```

NOTE

En este ejemplo, vamos a definir la solicitud HTTP en el objeto `options`. Sin embargo, el módulo de solicitud también admite métodos de conveniencia como `.post` y `.get`. Para más información, consulte [métodos de conveniencia](#).

Colocación de todo junto

Eso es todo, ha creado un sencillo programa que llama a Translator Text API y devuelve una respuesta JSON. Ahora es el momento de ejecutar el programa:

```
node detect.js
```

Respuesta de muestra

Después de ejecutar el ejemplo, debería ver lo siguiente impreso en el terminal:

NOTE

Busque la abreviatura del país o región en esta [lista de idiomas](#).

```
[
  {
    "alternatives": [
      {
        "isTranslationSupported": true,
        "isTransliterationSupported": false,
        "language": "pt",
        "score": 1.0
      },
      {
        "isTranslationSupported": true,
        "isTransliterationSupported": false,
        "language": "en",
        "score": 1.0
      }
    ],
    "isTranslationSupported": true,
    "isTransliterationSupported": false,
    "language": "it",
    "score": 1.0
  }
]
```

Limpieza de recursos

Si ha codificado de forma rígida la clave de suscripción en el programa, asegúrese de quitarla cuando haya terminado con esta guía de inicio rápido.

Pasos siguientes

Eche un vistazo a la referencia de API para comprender todo lo que puede hacer con Translator Text API.

[Referencia de API](#)

Requisitos previos

Esta guía de inicio rápido requiere:

- [Go](#)
- Una suscripción a Azure: [cree una cuenta gratuita](#).

Instalación

Creación de un recurso de Translator Text

Los servicios de Azure Cognitive Services se representan por medio de recursos de Azure a los que se suscribe. Cree un recurso para Translator Text mediante [Azure Portal](#) o la [CLI de Azure](#) en la máquina local. También puede:

- Obtener una [clave de prueba](#) válida durante siete días de forma gratuita. Después de registrarse, estará disponible en el sitio web de Azure.
- Vea un recurso existente en [Azure Portal](#).

Después de obtener una clave del recurso o la suscripción de evaluación, cree dos [variables de entorno](#):

- `TRANSLATOR_TEXT_SUBSCRIPTION_KEY` : la clave de suscripción del recurso de Translator Text.
- `TRANSLATOR_TEXT_ENDPOINT` : el punto de conexión global para Translator Text. Mediante `https://api.cognitive.microsofttranslator.com/`.

Creación de un proyecto e importación de los módulos necesarios

Cree un proyecto de Go con su IDE o editor favorito. A continuación, copie este fragmento de código en un archivo llamado `detect-language.go`.

```
package main

import (
    "bytes"
    "encoding/json"
    "fmt"
    "log"
    "net/http"
    "net/url"
    "os"
)
```

Creación de la función main

En este ejemplo se intenta leer la clave de suscripción y el punto de conexión de Translator Text desde estas variables de entorno: `TRANSLATOR_TEXT_SUBSCRIPTION_KEY` y `TRANSLATOR_TEXT_ENDPOINT`. Si no está familiarizado con las variables de entorno, puede establecer `subscriptionKey` y `endpoint` como cadenas y convertir en comentario las instrucciones condicionales.

Copie este código en el proyecto:

```
func main() {
    /*
     * Read your subscription key from an env variable.
     * Please note: You can replace this code block with
     * var subscriptionKey = "YOUR_SUBSCRIPTION_KEY" if you don't
     * want to use env variables. If so, be sure to delete the "os" import.
     */
    if "" == os.Getenv("TRANSLATOR_TEXT_SUBSCRIPTION_KEY") {
        log.Fatal("Please set/export the environment variable TRANSLATOR_TEXT_SUBSCRIPTION_KEY.")
    }
    subscriptionKey := os.Getenv("TRANSLATOR_TEXT_SUBSCRIPTION_KEY")
    if "" == os.Getenv("TRANSLATOR_TEXT_ENDPOINT") {
        log.Fatal("Please set/export the environment variable TRANSLATOR_TEXT_ENDPOINT.")
    }
    endpoint := os.Getenv("TRANSLATOR_TEXT_ENDPOINT")
    uri := endpoint + "/detect?api-version=3.0"
    /*
     * This calls our breakSentence function, which we'll
     * create in the next section. It takes a single argument,
     * the subscription key.
     */
    detect(subscriptionKey, uri)
}
```

Creación de una función para detectar el idioma del texto

Vamos a crear una función para detectar el idioma del texto. Esta función tendrá un solo argumento, su clave de suscripción de Translator Text.

```
func detect(subscriptionKey string, uri string) {
    /*
     * In the next few sections, we'll add code to this
     * function to make a request and handle the response.
     */
}
```

A continuación, vamos a construir la dirección URL. La dirección URL se crea mediante los métodos `Parse()` y `Query()`.

Copie este código en la función `detect`.

```
// Build the request URL. See: https://golang.org/pkg/net/url/#example_URL_Parse
u, _ := url.Parse(uri)
q := u.Query()
u.RawQuery = q.Encode()
```

NOTE

Para más información sobre los puntos de conexión, las rutas y los parámetros de la solicitud, consulte [Translator Text API 3.0: Detección](#).

Creación de una estructura para el cuerpo de la solicitud

A continuación, cree una estructura anónima para el cuerpo de la solicitud y codifíquelo como JSON con `json.Marshal()`. Agregue este código a la función `detect`.

```
// Create an anonymous struct for your request body and encode it to JSON
body := []struct {
    Text string
}{
    {Text: "Salve, Mondo!"},
}
b, _ := json.Marshal(body)
```

Compilar la solicitud

Ahora que se ha codificado el cuerpo de la solicitud como JSON, puede crear la solicitud POST y llamar a Translator Text API.

```
// Build the HTTP POST request
req, err := http.NewRequest("POST", u.String(), bytes.NewBuffer(b))
if err != nil {
    log.Fatal(err)
}
// Add required headers to the request
req.Header.Add("Ocp-Apim-Subscription-Key", subscriptionKey)
req.Header.Add("Content-Type", "application/json")

// Call the Translator Text API
res, err := http.DefaultClient.Do(req)
if err != nil {
    log.Fatal(err)
}
```

Si usa una suscripción a varios servicios de Cognitive Services, también debe incluir

Ocp-Apim-Subscription-Region en los parámetros de la solicitud. [Más información sobre la autenticación con la suscripción a varios servicios.](#)

Control e impresión de la respuesta

Agregue este código a la función `detect` para descodificar la respuesta JSON y luego dé formato al resultado e imprímalo.

```
// Decode the JSON response
var result interface{}
if err := json.NewDecoder(res.Body).Decode(&result); err != nil {
    log.Fatal(err)
}
// Format and print the response to terminal
prettyJSON, _ := json.MarshalIndent(result, "", " ")
fmt.Printf("%s\n", prettyJSON)
```

Colocación de todo junto

Eso es todo, ha creado un sencillo programa que llama a Translator Text API y devuelve una respuesta JSON. Ahora es el momento de ejecutar el programa:

```
go run detect-language.go
```

Si desea comparar su código con el nuestro, el ejemplo completo está disponible en [GitHub](#).

Respuesta de muestra

Después de ejecutar el ejemplo, debería ver lo siguiente impreso en el terminal:

NOTE

Busque la abreviatura del país o región en esta [lista de idiomas](#).

```
[
  {
    "alternatives": [
      {
        "isTranslationSupported": true,
        "isTransliterationSupported": false,
        "language": "pt",
        "score": 1
      },
      {
        "isTranslationSupported": true,
        "isTransliterationSupported": false,
        "language": "en",
        "score": 1
      }
    ],
    "isTranslationSupported": true,
    "isTransliterationSupported": false,
    "language": "it",
    "score": 1
  }
]
```

Pasos siguientes

Eche un vistazo a la referencia de API para comprender todo lo que puede hacer con Translator Text API.

[Referencia de API](#)

Consulte también

- [Traducir texto](#)
- [Transliterar texto](#)
- [Obtener traducciones alternativas](#)
- [Obtener una lista de idiomas admitidos](#)
- [Determinar la longitud de las oraciones de una entrada](#)

Inicio rápido: Búsqueda de palabras con un diccionario bilingüe

13/01/2020 • 35 minutes to read • [Edit Online](#)

En este inicio rápido aprenderá a obtener posibles traducciones alternativas de un término y se proporcionan ejemplos de uso mediante la API Translator Text.

En esta guía de inicio rápido, se requiere una [cuenta de Azure Cognitive Services](#) con un recurso de Translator Text. Si no tiene una cuenta, puede usar la [evaluación gratuita](#) para obtener una clave de suscripción.

Requisitos previos

Esta guía de inicio rápido requiere:

- C# 7.1 o posterior
- [SDK de .NET](#)
- [Paquete NuGet de Json.NET](#)
- [Visual Studio](#), [Visual Studio Code](#) o su editor favorito de código
- Una suscripción a Azure: [cree una cuenta gratuita](#).

Instalación

Creación de un recurso de Translator Text

Los servicios de Azure Cognitive Services se representan por medio de recursos de Azure a los que se suscribe. Cree un recurso para Translator Text mediante [Azure Portal](#) o la [CLI de Azure](#) en la máquina local. También puede:

- Obtener una [clave de prueba](#) válida durante siete días de forma gratuita. Después de registrarse, estará disponible en el sitio web de Azure.
- Vea un recurso existente en [Azure Portal](#).

Después de obtener una clave del recurso o la suscripción de evaluación, cree dos [variables de entorno](#):

- `TRANSLATOR_TEXT_SUBSCRIPTION_KEY`: la clave de suscripción del recurso de Translator Text.
- `TRANSLATOR_TEXT_ENDPOINT`: el punto de conexión global para Translator Text. Mediante `https://api.cognitive.microsofttranslator.com/`.

Creación de un proyecto de .NET Core

Abra un nuevo símbolo del sistema (o una sesión de terminal) y ejecute estos comandos:

```
dotnet new console -o alternate-sample
cd alternate-sample
```

El primer comando hace dos cosas. Crea una nueva aplicación de consola .NET y crea un directorio denominado `alternate-sample`. El segundo comando cambia al directorio del proyecto.

A continuación, deberá instalar Json.Net. En el directorio del proyecto, ejecute:

```
dotnet add package Newtonsoft.Json --version 11.0.2
```

Incorporación de los espacios de nombres necesarios al proyecto

El comando `dotnet new console` que ejecutó anteriormente creó un proyecto que incluía `Program.cs`. En este archivo es donde deberá colocar el código de la aplicación. Abra `Program.cs` y reemplace las instrucciones `using` existentes. Estas instrucciones garantizan que tiene acceso a todos los tipos necesarios para compilar y ejecutar la aplicación de ejemplo.

```
using System;
using System.Net.Http;
using System.Text;
using Newtonsoft.Json;
```

Obtención de información de la suscripción a partir de las variables de entorno

Agregue las líneas siguientes a la clase `Program`. Estas líneas leen la clave de suscripción y el punto de conexión de las variables de entorno y devuelven un error si se produce algún problema.

```
private const string key_var = "TRANSLATOR_TEXT_SUBSCRIPTION_KEY";
private static readonly string subscriptionKey = Environment.GetEnvironmentVariable(key_var);

private const string endpoint_var = "TRANSLATOR_TEXT_ENDPOINT";
private static readonly string endpoint = Environment.GetEnvironmentVariable(endpoint_var);

static Program()
{
    if (null == subscriptionKey)
    {
        throw new Exception("Please set/export the environment variable: " + key_var);
    }
    if (null == endpoint)
    {
        throw new Exception("Please set/export the environment variable: " + endpoint_var);
    }
}
// The code in the next section goes here.
```

Crear una función para obtener posibles traducciones alternativas

Dentro de la clase `Program`, cree una función denominada `AltTranslation`. Esta clase encapsula el código que se usa para llamar al recurso Dictionary e imprime el resultado en la consola.

```
static void AltTranslation()
{
    /*
     * The code for your call to the translation service will be added to this
     * function in the next few sections.
     */
}
```

Construcción del identificador URI

Agregue estas líneas a la función `AltTranslation`. Observe que junto con la `api-version`, se han declarado dos parámetros adicionales. Estos parámetros se usan para establecer la entrada y la salida de la traducción. En este ejemplo, se trata de inglés (`en`) y español (`es`).

```
string route = "/dictionary/lookup?api-version=3.0";
static string params_ = "from=en&to=es";
static string uri = endpoint + path + params_;
```

A continuación, se debe crear y serializar el objeto JSON que incluye el texto que desea traducir. Tenga en cuenta que puede pasar más de un objeto en la matriz `body`.

```
System.Object[] body = new System.Object[] { new { Text = @"Elephants" } };
var requestBody = JsonConvert.SerializeObject(body);
```

Creación de una instancia de cliente y realización de una solicitud

Estas líneas crean instancias de `HttpClient` y `HttpRequestMessage`:

```
using (var client = new HttpClient())
using (var request = new HttpRequestMessage())
{
    // In the next few sections you'll add code to construct the request.
}
```

Construcción de la solicitud e impresión de la respuesta

Dentro de `HttpRequestMessage`:

- Declarará el método HTTP
- Construirá el URI de solicitud
- Insertará el cuerpo de la solicitud (el objeto JSON serializado)
- Agregará los encabezados necesarios
- Realizará una solicitud asíncrona
- Impresión de la respuesta

Agregue este código a `HttpRequestMessage`:


```
// Set the method to POST
request.Method = HttpMethod.Post;

// Construct the full URI
request.RequestUri = new Uri(uri);

// Add the serialized JSON object to your request
request.Content = new StringContent(requestBody, Encoding.UTF8, "application/json");

// Add the authorization header
request.Headers.Add("Ocp-Apim-Subscription-Key", subscriptionKey);

// Send request, get response
var response = client.SendAsync(request).Result;
var jsonResponse = response.Content.ReadAsStringAsync().Result;

// Print the response
Console.WriteLine(PrettyPrint(jsonResponse));
Console.WriteLine("Press any key to continue.");
```

Agregue `PrettyPrint` para aplicar formato a la respuesta JSON:

```
static string PrettyPrint(string s)
{
    return JsonConvert.SerializeObject(JsonConvert.DeserializeObject(s), Formatting.Indented);
}
```

Si usa una suscripción a varios servicios de Cognitive Services, también debe incluir

`Ocp-Apim-Subscription-Region` en los parámetros de la solicitud. [Más información sobre la autenticación con la suscripción a varios servicios.](#)

Colocación de todo junto

El último paso consiste en llamar a `AltTranslation()` en la función `Main`. Busque

`static void Main(string[] args)` y agregue estas líneas:

```
AltTranslation();
Console.WriteLine("Press any key to continue.");
Console.ReadKey();
```

Ejecutar la aplicación de ejemplo

Eso es todo, ya está listo para ejecutar la aplicación de ejemplo. Desde la línea de comandos (o sesión de terminal), vaya al directorio del proyecto y ejecute:

```
dotnet run
```

Respuesta de muestra

```
[
  {
    "displaySource": "elephants",
    "normalizedSource": "elephants",
    "translations": [
      {
        "backTranslations": [
          {
            "displayText": "elephants",
            "frequencyCount": 1207,
            "normalizedText": "elephants",
            "numExamples": 5
          }
        ],
        "confidence": 1.0,
        "displayTarget": "elefantes",
        "normalizedTarget": "elefantes",
        "posTag": "NOUN",
        "prefixWord": ""
      }
    ]
  }
]
```

Limpieza de recursos

Asegúrese de quitar cualquier información confidencial del código fuente de la aplicación de ejemplo como, por ejemplo, las claves de suscripción.

Pasos siguientes

Eche un vistazo a la referencia de API para comprender todo lo que puede hacer con Translator Text API.

[Referencia de API](#)

Requisitos previos

Esta guía de inicio rápido requiere:

- [JDK 7 o posterior](#)
- [Gradle](#)
- Una suscripción a Azure: [cree una cuenta gratuita](#).

Instalación

Creación de un recurso de Translator Text

Los servicios de Azure Cognitive Services se representan por medio de recursos de Azure a los que se suscribe. Cree un recurso para Translator Text mediante [Azure Portal](#) o la [CLI de Azure](#) en la máquina local. También puede:

- Obtener una [clave de prueba](#) válida durante siete días de forma gratuita. Después de registrarse, estará disponible en el sitio web de Azure.
- Vea un recurso existente en [Azure Portal](#).

Después de obtener una clave del recurso o la suscripción de evaluación, cree dos [variables de entorno](#):

- `TRANSLATOR_TEXT_SUBSCRIPTION_KEY`: la clave de suscripción del recurso de Translator Text.
- `TRANSLATOR_TEXT_ENDPOINT`: el punto de conexión global para Translator Text. Mediante `https://api.cognitive.microsofttranslator.com/`.

Inicialización de un proyecto con Gradle

Comencemos por crear un directorio de trabajo para este proyecto. Desde la línea de comandos (o terminal), ejecute este comando:

```
mkdir alt-translation-sample
cd alt-translation-sample
```

A continuación, va a inicializar un proyecto de Gradle. Este comando creará archivos de compilación esenciales para Gradle, el más importante, el `build.gradle.kts`, que se utiliza en tiempo de ejecución para crear y configurar la aplicación. Ejecute este comando desde el directorio de trabajo:

```
gradle init --type basic
```

Cuando se le solicite que elija un **DSL**, seleccione **Kotlin**.

Configuración del archivo de compilación

Localice `build.gradle.kts` y ábralo con el IDE o editor de texto favorito. A continuación, se copia en esta configuración de compilación:

```
plugins {
    java
    application
}
application {
    mainClassName = "AltTranslation"
}
repositories {
    mavenCentral()
}
dependencies {
    compile("com.squareup.okhttp:okhttp:2.5.0")
    compile("com.google.code.gson:gson:2.8.5")
}
```

Tenga en cuenta que este ejemplo tiene dependencias en OkHttp para las solicitudes HTTP, y Gson para administrar y analizar JSON. Si desea obtener más información acerca de las configuraciones de compilación, consulte [Creating New Gradle Builds](#) (Creación de nuevas compilaciones de Gradle).

Creación de un archivo Java

Vamos a crear una carpeta para la aplicación de ejemplo. En el directorio de trabajo, ejecute:

```
mkdir -p src\main\java
```

A continuación, en esta carpeta, cree un archivo denominado `AltTranslation.java`.

Importación de bibliotecas necesarias

Abra `AltTranslation.java` y agregue las siguientes instrucciones de importación:

```
import java.io.*;
import java.net.*;
import java.util.*;
import com.google.gson.*;
import com.squareup.okhttp.*;
```

Definición de variables

En primer lugar, deberá crear una clase pública para el proyecto:

```
public class AltTranslation {
    // All project code goes here...
}
```

Agregue estas líneas a la clase `AltTranslation`. Primero, se lee la clave de suscripción y el punto de conexión de las variables de entorno. A continuación, observe que junto con la `api-version`, se han anexado dos parámetros adicionales a `url`. Estos parámetros se usan para establecer la entrada y la salida de la traducción. En este ejemplo, se trata de inglés (`en`) y español (`es`).

```
private static String subscriptionKey = System.getenv("TRANSLATOR_TEXT_SUBSCRIPTION_KEY");
private static String endpoint = System.getenv("TRANSLATOR_TEXT_ENDPOINT");
String url = endpoint + "/dictionary/lookup?api-version=3.0&from=en&to=es";
```

Si usa una suscripción a varios servicios de Cognitive Services, también debe incluir

`Ocp-Apim-Subscription-Region` en los parámetros de la solicitud. [Más información sobre la autenticación con la suscripción a varios servicios.](#)

Creación de un cliente y compilación de una solución

Agregue esta línea a la clase `AltTranslation` para crear una instancia de `OkHttpClient`:

```
// Instantiates the OkHttpClient.
OkHttpClient client = new OkHttpClient();
```

A continuación, vamos a crear la solicitud POST. No dude en cambiar el texto para la traducción.

```
// This function performs a POST request.
public String Post() throws IOException {
    MediaType mediaType = MediaType.parse("application/json");
    RequestBody body = RequestBody.create(mediaType,
        "[{\n\t\"Text\": \"Pineapples\"\n}]");
    Request request = new Request.Builder()
        .url(url).post(body)
        .addHeader("Ocp-Apim-Subscription-Key", subscriptionKey)
        .addHeader("Content-type", "application/json").build();
    Response response = client.newCall(request).execute();
    return response.body().string();
}
```

Creación de una función para analizar la respuesta

Esta sencilla función analiza y embellece la respuesta JSON del servicio Translator Text.

```
// This function prettifies the json response.
public static String prettify(String json_text) {
    JsonParser parser = new JsonParser();
    JsonElement json = parser.parse(json_text);
    Gson gson = new GsonBuilder().setPrettyPrinting().create();
    return gson.toJson(json);
}
```

Colocación de todo junto

El último paso consiste en realizar una solicitud y obtener una respuesta. Agregue estas líneas al proyecto:

```
public static void main(String[] args) {
    try {
        AltTranslation altTranslationRequest = new AltTranslation();
        String response = altTranslationRequest.Post();
        System.out.println(prettify(response));
    } catch (Exception e) {
        System.out.println(e);
    }
}
```

Ejecutar la aplicación de ejemplo

Eso es todo, ya está listo para ejecutar la aplicación de ejemplo. Desde la línea de comandos (o sesión de terminal), navegue hasta la raíz del directorio de trabajo y ejecute:

```
gradle build
```

Cuando la compilación se complete, ejecute lo siguiente:

```
gradle run
```

Respuesta de muestra

```
[
  {
    "normalizedSource": "pineapples",
    "displaySource": "pineapples",
    "translations": [
      {
        "normalizedTarget": "piñas",
        "displayTarget": "piñas",
        "posTag": "NOUN",
        "confidence": 0.7016,
        "prefixWord": "",
        "backTranslations": [
          {
            "normalizedText": "pineapples",
            "displayText": "pineapples",
            "numExamples": 5,
            "frequencyCount": 158
          },
          {
            "normalizedText": "cones",
            "displayText": "cones",
            "numExamples": 5,
            "frequencyCount": 13
          },
          {
            "normalizedText": "piña",
            "displayText": "piña",
            "numExamples": 3,
            "frequencyCount": 5
          },
          {
            "normalizedText": "ganks",
            "displayText": "ganks",
            "numExamples": 2,
            "frequencyCount": 3
          }
        ]
      }
    ],
  },
  {
    "normalizedTarget": "ananás",
    "displayTarget": "ananás",
    "posTag": "NOUN",
    "confidence": 0.2984,
    "prefixWord": "",
    "backTranslations": [
      {
        "normalizedText": "pineapples",
        "displayText": "pineapples",
        "numExamples": 2,
        "frequencyCount": 16
      }
    ]
  }
]
}
```

Pasos siguientes

Eche un vistazo a la referencia de API para comprender todo lo que puede hacer con Translator Text API.

[Referencia de API](#)

Requisitos previos

Esta guía de inicio rápido requiere:

- [Python 2.7.x o 3.x](#)
- Una suscripción a Azure: [cree una cuenta gratuita](#).

Instalación

Creación de un recurso de Translator Text

Los servicios de Azure Cognitive Services se representan por medio de recursos de Azure a los que se suscribe. Cree un recurso para Translator Text mediante [Azure Portal](#) o la [CLI de Azure](#) en la máquina local. También puede:

- Obtener una [clave de prueba](#) válida durante siete días de forma gratuita. Después de registrarse, estará disponible en el sitio web de Azure.
- Vea un recurso existente en [Azure Portal](#).

Después de obtener una clave del recurso o la suscripción de evaluación, cree dos [variables de entorno](#):

- `TRANSLATOR_TEXT_SUBSCRIPTION_KEY`: la clave de suscripción del recurso de Translator Text.
- `TRANSLATOR_TEXT_ENDPOINT`: el punto de conexión global para Translator Text. Mediante `https://api.cognitive.microsofttranslator.com/`.

Creación de un proyecto e importación de los módulos necesarios

Cree un proyecto de Python con su editor o IDE favoritos, o bien cree una carpeta en el escritorio. Copie este fragmento de código en un archivo llamado `dictionary-lookup.py` en el proyecto o carpeta.

```
# -*- coding: utf-8 -*-  
import os, requests, uuid, json
```

NOTE

Si no ha usado estos módulos deberá instalarlos antes de ejecutar el programa. Para instalar estos paquetes, ejecute:

```
pip install requests uuid
```

El primer comentario le indica al intérprete de Python que debe usar la codificación UTF-8. Después, se importan los módulos necesarios para leer la clave de suscripción desde una variable de entorno, construir la solicitud HTTP, crear un identificador único y controlar la respuesta JSON que devuelve Translator Text API.

Establecimiento de la clave de suscripción, el punto de conexión y la ruta de acceso

En este ejemplo se intenta leer la clave de suscripción y el punto de conexión de Translator Text desde las variables de entorno `TRANSLATOR_TEXT_KEY` y `TRANSLATOR_TEXT_ENDPOINT`. Si no está familiarizado con las variables de entorno, puede establecer `subscription_key` y `endpoint` como cadenas y convertir en comentario las instrucciones condicionales.

Copie este código en el proyecto:

```
key_var_name = 'TRANSLATOR_TEXT_SUBSCRIPTION_KEY'
if not key_var_name in os.environ:
    raise Exception('Please set/export the environment variable: {}'.format(key_var_name))
subscription_key = os.environ[key_var_name]

endpoint_var_name = 'TRANSLATOR_TEXT_ENDPOINT'
if not endpoint_var_name in os.environ:
    raise Exception('Please set/export the environment variable: {}'.format(endpoint_var_name))
endpoint = os.environ[endpoint_var_name]
```

El punto de conexión global de Translator Text se establece como `endpoint`. `path` establece la ruta de `dictionary/lookup` e identifica que deseamos usar la versión 3 de la API.

`params` se utiliza para establecer los idiomas de entrada y salida. En este ejemplo, se van a usar inglés y español: `en` y `es`.

NOTE

Para más información sobre los puntos de conexión, las rutas y los parámetros de la solicitud, consulte [Translator Text API 3.0: Búsqueda de diccionario](#).

```
path = '/dictionary/lookup?api-version=3.0'
params = '&from=en&to=es'
constructed_url = endpoint + path + params
```

Incorporación de encabezados

La manera más fácil de autenticar una solicitud es pasar la clave de suscripción como un encabezado `Ocp-Apim-Subscription-Key`, que es el que se usa en este ejemplo. O bien, puede intercambiar la clave de suscripción para un token de acceso y pasar este token como un encabezado `Authorization` para validar la solicitud. Para más información, consulte [Autenticación](#).

Copie este fragmento de código en el proyecto:

```
headers = {
    'Ocp-Apim-Subscription-Key': subscription_key,
    'Content-type': 'application/json',
    'X-ClientTraceId': str(uuid.uuid4())
}
```

Si usa una suscripción a varios servicios de Cognitive Services, también debe incluir

`Ocp-Apim-Subscription-Region` en los parámetros de la solicitud. [Más información sobre la autenticación con la suscripción a varios servicios](#).

Creación de una solicitud para buscar traducciones alternativas

Defina la cadena (o cadenas) para las que desea encontrar traducciones:

```
# You can pass more than one object in body.
body = [{
    'text': 'Elephants'
}]
```

A continuación, vamos a crear una solicitud POST mediante el módulo `requests`, que usa tres argumentos: la

dirección URL concatenada, los encabezados de solicitud y el cuerpo de solicitud:

```
request = requests.post(constructed_url, headers=headers, json=body)
response = request.json()
```

Impresión de la respuesta

El último paso es imprimir los resultados. Este fragmento de código adorna los resultados mediante una clasificación de las claves, el establecimiento de una sangría y la declaración de separadores de elementos y de claves.

```
print(json.dumps(response, sort_keys=True, indent=4,
                  ensure_ascii=False, separators=(',', ' ')))
```

Colocación de todo junto

Eso es todo, ha creado un sencillo programa que llama a Translator Text API y devuelve una respuesta JSON. Ahora es el momento de ejecutar el programa:

```
python alt-translations.py
```

Si desea comparar su código con el nuestro, el ejemplo completo está disponible en [GitHub](#).

Respuesta de muestra

```
[
  {
    "displaySource": "elephants",
    "normalizedSource": "elephants",
    "translations": [
      {
        "backTranslations": [
          {
            "displayText": "elephants",
            "frequencyCount": 1207,
            "normalizedText": "elephants",
            "numExamples": 5
          }
        ],
        "confidence": 1.0,
        "displayTarget": "elefantes",
        "normalizedTarget": "elefantes",
        "posTag": "NOUN",
        "prefixWord": ""
      }
    ]
  }
]
```

Limpieza de recursos

Si ha codificado de forma rígida la clave de suscripción en el programa, asegúrese de quitarla cuando haya terminado con esta guía de inicio rápido.

Pasos siguientes

Eche un vistazo a la referencia de API para comprender todo lo que puede hacer con Translator Text API.

[Referencia de API](#)

Requisitos previos

Esta guía de inicio rápido requiere:

- [Node 8.12.x o posterior](#)
- Una suscripción a Azure: [cree una cuenta gratuita](#).

Instalación

Creación de un recurso de Translator Text

Los servicios de Azure Cognitive Services se representan por medio de recursos de Azure a los que se suscribe. Cree un recurso para Translator Text mediante [Azure Portal](#) o la [CLI de Azure](#) en la máquina local. También puede:

- Obtener una [clave de prueba](#) válida durante siete días de forma gratuita. Después de registrarse, estará disponible en el sitio web de Azure.
- Vea un recurso existente en [Azure Portal](#).

Después de obtener una clave del recurso o la suscripción de evaluación, cree dos [variables de entorno](#):

- `TRANSLATOR_TEXT_SUBSCRIPTION_KEY`: la clave de suscripción del recurso de Translator Text.
- `TRANSLATOR_TEXT_ENDPOINT`: el punto de conexión global para Translator Text. Mediante `https://api.cognitive.microsofttranslator.com/`.

Creación de un proyecto e importación de los módulos necesarios

Cree un proyecto con su editor o IDE favoritos, o cree una carpeta en el escritorio. Copie este fragmento de código en un archivo llamado `alt-translations.js` en el proyecto o carpeta.

```
const request = require('request');
const uuidv4 = require('uuid/v4');
```

NOTE

Si no ha usado estos módulos deberá instalarlos antes de ejecutar el programa. Para instalar estos paquetes, ejecute:

```
npm install request uuidv4
```

Estos módulos son necesarios para construir la solicitud HTTP y crear un identificador único para el encabezado

```
'X-ClientTraceId'.
```

Establecimiento de la clave de suscripción y el punto de conexión

En este ejemplo se intenta leer la clave de suscripción y el punto de conexión de Translator Text desde estas variables de entorno: `TRANSLATOR_TEXT_SUBSCRIPTION_KEY` y `TRANSLATOR_TEXT_ENDPOINT`. Si no está familiarizado con las variables de entorno, puede establecer `subscriptionKey` y `endpoint` como cadenas y convertir en comentario las instrucciones condicionales.

Copie este código en el proyecto:

```
var key_var = 'TRANSLATOR_TEXT_SUBSCRIPTION_KEY';
if (!process.env[key_var]) {
    throw new Error('Please set/export the following environment variable: ' + key_var);
}
var subscriptionKey = process.env[key_var];
var endpoint_var = 'TRANSLATOR_TEXT_ENDPOINT';
if (!process.env[endpoint_var]) {
    throw new Error('Please set/export the following environment variable: ' + endpoint_var);
}
var endpoint = process.env[endpoint_var];
```

Configuración de la solicitud

El método `request()`, disponible mediante el módulo de solicitud, nos permite pasar el método HTTP, la dirección URL, los parámetros de la solicitud, los encabezados y el cuerpo de JSON como un objeto `options`. En este fragmento de código, vamos a configurar la solicitud:

NOTE

Para más información sobre los puntos de conexión, las rutas y los parámetros de la solicitud, consulte [Translator Text API 3.0: Búsqueda de diccionario](#).

```
let options = {
  method: 'POST',
  baseUrl: endpoint,
  url: 'dictionary/lookup',
  qs: {
    'api-version': '3.0',
    'from': 'en',
    'to': 'es'
  },
  headers: {
    'Ocp-Apim-Subscription-Key': subscriptionKey,
    'Content-type': 'application/json',
    'X-ClientTraceId': uuidv4().toString()
  },
  body: [{
    'text': 'Elephants'
  }],
  json: true,
};
```

La manera más fácil de autenticar una solicitud es pasar la clave de suscripción como un encabezado `Ocp-Apim-Subscription-Key`, que es el que se usa en este ejemplo. O bien, puede intercambiar la clave de suscripción para un token de acceso y pasar este token como un encabezado `Authorization` para validar la solicitud.

Si usa una suscripción a varios servicios de Cognitive Services, también debe incluir `Ocp-Apim-Subscription-Region` en los encabezados de la solicitud.

Para más información, consulte [Autenticación](#).

Realización de solicitud e impresión de la respuesta

A continuación, vamos a crear una solicitud mediante el método `request()`. Toma el objeto `options` que se creó en la sección anterior como el primer argumento y, a continuación, imprime la respuesta JSON embellecida.

```
request(options, function(err, res, body){
    console.log(JSON.stringify(body, null, 4));
});
```

NOTE

En este ejemplo, vamos a definir la solicitud HTTP en el objeto `options`. Sin embargo, el módulo de solicitud también admite métodos de conveniencia como `.post` y `.get`. Para más información, consulte [métodos de conveniencia](#).

Colocación de todo junto

Eso es todo, ha creado un sencillo programa que llama a Translator Text API y devuelve una respuesta JSON. Ahora es el momento de ejecutar el programa:

```
node alt-translations.js
```

Si desea comparar su código con el nuestro, el ejemplo completo está disponible en [GitHub](#).

Respuesta de muestra

```
[
  {
    "displaySource": "elephants",
    "normalizedSource": "elephants",
    "translations": [
      {
        "backTranslations": [
          {
            "displayText": "elephants",
            "frequencyCount": 1207,
            "normalizedText": "elephants",
            "numExamples": 5
          }
        ],
        "confidence": 1.0,
        "displayTarget": "elefantes",
        "normalizedTarget": "elefantes",
        "posTag": "NOUN",
        "prefixWord": ""
      }
    ]
  }
]
```

Limpieza de recursos

Si ha codificado de forma rígida la clave de suscripción en el programa, asegúrese de quitarla cuando haya terminado con esta guía de inicio rápido.

Pasos siguientes

Eche un vistazo a la referencia de API para comprender todo lo que puede hacer con Translator Text API.

[Referencia de API](#)

Requisitos previos

Esta guía de inicio rápido requiere:

- [Go](#)
- Una suscripción a Azure: [cree una cuenta gratuita](#).

Instalación

Creación de un recurso de Translator Text

Los servicios de Azure Cognitive Services se representan por medio de recursos de Azure a los que se suscribe. Cree un recurso para Translator Text mediante [Azure Portal](#) o la [CLI de Azure](#) en la máquina local. También puede:

- Obtener una [clave de prueba](#) válida durante siete días de forma gratuita. Después de registrarse, estará disponible en el sitio web de Azure.
- Vea un recurso existente en [Azure Portal](#).

Después de obtener una clave del recurso o la suscripción de evaluación, cree dos [variables de entorno](#):

- `TRANSLATOR_TEXT_SUBSCRIPTION_KEY`: la clave de suscripción del recurso de Translator Text.
- `TRANSLATOR_TEXT_ENDPOINT`: el punto de conexión global para Translator Text. Mediante `https://api.cognitive.microsofttranslator.com/`.

Creación de un proyecto e importación de los módulos necesarios

Cree un proyecto de Go con su IDE o editor favoritos, o bien en una nueva carpeta en el escritorio. A continuación, copie este fragmento de código en un archivo llamado `dictionaryLookup.go` en el proyecto o carpeta.

```
package main

import (
    "bytes"
    "encoding/json"
    "fmt"
    "log"
    "net/http"
    "net/url"
    "os"
)
```

Creación de la función main

En este ejemplo se intenta leer la clave de suscripción y el punto de conexión de Translator Text desde estas variables de entorno: `TRANSLATOR_TEXT_SUBSCRIPTION_KEY` y `TRANSLATOR_TEXT_ENDPOINT`. Si no está familiarizado con las variables de entorno, puede establecer `subscriptionKey` y `endpoint` como cadenas y convertir en comentario las instrucciones condicionales.

Copie este código en el proyecto:

```
func main() {
    /*
     * Read your subscription key from an env variable.
     * Please note: You can replace this code block with
     * var subscriptionKey = "YOUR_SUBSCRIPTION_KEY" if you don't
     * want to use env variables. If so, be sure to delete the "os" import.
     */
    if "" == os.Getenv("TRANSLATOR_TEXT_SUBSCRIPTION_KEY") {
        log.Fatal("Please set/export the environment variable TRANSLATOR_TEXT_SUBSCRIPTION_KEY.")
    }
    subscriptionKey := os.Getenv("TRANSLATOR_TEXT_SUBSCRIPTION_KEY")
    if "" == os.Getenv("TRANSLATOR_TEXT_ENDPOINT") {
        log.Fatal("Please set/export the environment variable TRANSLATOR_TEXT_ENDPOINT.")
    }
    endpoint := os.Getenv("TRANSLATOR_TEXT_ENDPOINT")
    uri := endpoint + "/dictionary/lookup?api-version=3.0"
    /*
     * This calls our breakSentence function, which we'll
     * create in the next section. It takes a single argument,
     * the subscription key.
     */
    dictionaryLookup(subscriptionKey, uri)
}
```

Crear una función para obtener posibles traducciones alternativas

Vamos a crear una función para obtener posibles traducciones alternativas. Esta función tendrá un solo argumento, su clave de suscripción de Translator Text.

```
func dictionaryLookup(subscriptionKey string, uri string) {
    /*
     * In the next few sections, we'll add code to this
     * function to make a request and handle the response.
     */
}
```

A continuación, vamos a construir la dirección URL. La dirección URL se crea mediante los métodos `Parse()` y `Query()`. Observará que se han agregado los parámetros con el método `Add()`. En este ejemplo vamos a traducir de inglés a español.

Copie este código en la función `altTranslations`.

```
// Build the request URL. See: https://golang.org/pkg/net/url/#example_URL_Parse
u, _ := url.Parse(uri)
q := u.Query()
q.Add("from", "en")
q.Add("to", "es")
u.RawQuery = q.Encode()
```

NOTE

Para más información sobre los puntos de conexión, las rutas y los parámetros de la solicitud, consulte [Translator Text API 3.0: Búsqueda de diccionario](#).

Creación de una estructura para el cuerpo de la solicitud

A continuación, cree una estructura anónima para el cuerpo de la solicitud y codifíquelo como JSON con

`json.Marshal()` . Agregue este código a la función `altTranslations` .

```
// Create an anonymous struct for your request body and encode it to JSON
body := []struct {
    Text string
}{
    {Text: "Pineapples"},
}
b, _ := json.Marshal(body)
```

Compilar la solicitud

Ahora que se ha codificado el cuerpo de la solicitud como JSON, puede crear la solicitud POST y llamar a Translator Text API.

```
// Build the HTTP POST request
req, err := http.NewRequest("POST", u.String(), bytes.NewBuffer(b))
if err != nil {
    log.Fatal(err)
}
// Add required headers to the request
req.Header.Add("Ocp-Apim-Subscription-Key", subscriptionKey)
req.Header.Add("Content-Type", "application/json")

// Call the Translator Text API
res, err := http.DefaultClient.Do(req)
if err != nil {
    log.Fatal(err)
}
```

Si usa una suscripción a varios servicios de Cognitive Services, también debe incluir

`Ocp-Apim-Subscription-Region` en los parámetros de la solicitud. [Más información sobre la autenticación con la suscripción a varios servicios.](#)

Control e impresión de la respuesta

Agregue este código a la función `altTranslations` para decodificar la respuesta JSON y luego dé formato al resultado e imprímalo.

```
// Decode the JSON response
var result interface{}
if err := json.NewDecoder(res.Body).Decode(&result); err != nil {
    log.Fatal(err)
}
// Format and print the response to terminal
prettyJSON, _ := json.MarshalIndent(result, "", " ")
fmt.Printf("%s\n", prettyJSON)
```

Colocación de todo junto

Eso es todo, ha creado un sencillo programa que llama a Translator Text API y devuelve una respuesta JSON. Ahora es el momento de ejecutar el programa:

```
go run dictionaryLookup.go
```

Si desea comparar su código con el nuestro, el ejemplo completo está disponible en [GitHub](#).

Respuesta de muestra

```
[
  {
    "displaySource": "pineapples",
    "normalizedSource": "pineapples",
    "translations": [
      {
        "backTranslations": [
          {
            "displayText": "pineapples",
            "frequencyCount": 158,
            "normalizedText": "pineapples",
            "numExamples": 5
          },
          {
            "displayText": "cones",
            "frequencyCount": 13,
            "normalizedText": "cones",
            "numExamples": 5
          },
          {
            "displayText": "piña",
            "frequencyCount": 5,
            "normalizedText": "piña",
            "numExamples": 3
          },
          {
            "displayText": "ganks",
            "frequencyCount": 3,
            "normalizedText": "ganks",
            "numExamples": 2
          }
        ],
        "confidence": 0.7016,
        "displayTarget": "piñas",
        "normalizedTarget": "piñas",
        "posTag": "NOUN",
        "prefixWord": ""
      },
      {
        "backTranslations": [
          {
            "displayText": "pineapples",
            "frequencyCount": 16,
            "normalizedText": "pineapples",
            "numExamples": 2
          }
        ],
        "confidence": 0.2984,
        "displayTarget": "ananás",
        "normalizedTarget": "ananás",
        "posTag": "NOUN",
        "prefixWord": ""
      }
    ]
  }
]
```

Pasos siguientes

Eche un vistazo a la referencia de API para comprender todo lo que puede hacer con Translator Text API.

[Referencia de API](#)

Consulte también

- [Traducir texto](#)
- [Transliterar texto](#)
- [Identificar el idioma de entrada](#)
- [Obtener una lista de idiomas admitidos](#)
- [Determinar la longitud de las oraciones de una entrada](#)

Inicio rápido: Uso de la API Translator Text para obtener una lista de los idiomas admitidos

13/01/2020 • 33 minutes to read • [Edit Online](#)

En esta guía de inicio rápido se obtiene una lista de los idiomas admitidos para la traducción, la transliteración y la búsqueda en el diccionario mediante Translator Text API.

Requisitos previos

Esta guía de inicio rápido requiere:

- C# 7.1 o posterior
- [SDK de .NET](#)
- [Paquete NuGet de Json.NET](#)
- [Visual Studio](#), [Visual Studio Code](#) o su editor favorito de código
- Una suscripción a Azure: [cree una cuenta gratuita](#).

Instalación

Creación de un recurso de Translator Text

Los servicios de Azure Cognitive Services se representan por medio de recursos de Azure a los que se suscribe. Cree un recurso para Translator Text mediante [Azure Portal](#) o la [CLI de Azure](#) en la máquina local. También puede:

- Obtener una [clave de prueba](#) válida durante siete días de forma gratuita. Después de registrarse, estará disponible en el sitio web de Azure.
- Vea un recurso existente en [Azure Portal](#).

Después de obtener una clave del recurso o la suscripción de evaluación, cree dos [variables de entorno](#):

- `TRANSLATOR_TEXT_SUBSCRIPTION_KEY`: la clave de suscripción del recurso de Translator Text.
- `TRANSLATOR_TEXT_ENDPOINT`: el punto de conexión global para Translator Text. Mediante `https://api.cognitive.microsofttranslator.com/`.

Creación de un proyecto de .NET Core

Abra un nuevo símbolo del sistema (o una sesión de terminal) y ejecute estos comandos:

```
dotnet new console -o languages-sample
cd languages-sample
```

El primer comando hace dos cosas. Crea una nueva aplicación de consola .NET y crea un directorio denominado `languages-sample`. El segundo comando cambia al directorio del proyecto.

A continuación, deberá instalar Json.Net. En el directorio del proyecto, ejecute:

```
dotnet add package Newtonsoft.Json --version 11.0.2
```

Incorporación de los espacios de nombres necesarios al proyecto

El comando `dotnet new console` que ejecutó anteriormente creó un proyecto que incluía `Program.cs`. En este archivo es donde deberá colocar el código de la aplicación. Abra `Program.cs` y reemplace las instrucciones using existentes. Estas instrucciones garantizan que tiene acceso a todos los tipos necesarios para compilar y ejecutar la aplicación de ejemplo.

```
using System;
using System.Net.Http;
using System.Text;
using Newtonsoft.Json;
```

Obtención de información a partir de una variable de entorno

Agregue las líneas siguientes a la clase `Program`. Estas líneas leen la clave de suscripción y el punto de conexión de las variables de entorno y devuelven un error si se produce algún problema.

```
private const string endpoint_var = "TRANSLATOR_TEXT_ENDPOINT";
private static readonly string endpoint = Environment.GetEnvironmentVariable(endpoint_var);

static Program()
{
    if (null == endpoint)
    {
        throw new Exception("Please set/export the environment variable: " + endpoint_var);
    }
}
```

Creación de una función para obtener una lista de idiomas

Dentro de la clase `Program`, cree una función denominada `GetLanguages`. Esta clase encapsula el código que se usa para llamar a los recursos de idiomas e imprime el resultado en la consola.

```
static void GetLanguages()
{
    /*
     * The code for your call to the translation service will be added to this
     * function in the next few sections.
     */
}
```

Establecimiento de la ruta

Agregue estas líneas a la función `GetLanguages`.

```
string route = "/languages?api-version=3.0";
```

Creación de una instancia de cliente y realización de una solicitud

Estas líneas crean instancias de `HttpClient` y `HttpRequestMessage`:

```
using (var client = new HttpClient())
using (var request = new HttpRequestMessage())
{
    // In the next few sections you'll add code to construct the request.
}
```

Construcción de la solicitud e impresión de la respuesta

Dentro de `HttpRequestMessage`:

- Declarará el método HTTP
- Construirá el URI de solicitud
- Agregará los encabezados necesarios
- Realizará una solicitud asíncrona
- Impresión de la respuesta

Agregue este código a `HttpRequestMessage`:

```
// Set the method to GET
request.Method = HttpMethod.Get;
// Construct the full URI
request.RequestUri = new Uri(endpoint + route);
// Send request, get response
var response = client.SendAsync(request).Result;
var jsonResponse = response.Content.ReadAsStringAsync().Result;
// Pretty print the response
Console.WriteLine(PrettyPrint(jsonResponse));
Console.WriteLine("Press any key to continue.");
```

Si usa una suscripción a varios servicios de Cognitive Services, también debe incluir

`Ocp-Apim-Subscription-Region` en los parámetros de la solicitud. [Más información sobre la autenticación con la suscripción a varios servicios.](#)

Para imprimir la respuesta con "Impresión con sangría" (formato de la respuesta), agregue esta función a la clase Program:

```
static string PrettyPrint(string s)
{
    return JsonConvert.SerializeObject(JsonConvert.DeserializeObject(s), Formatting.Indented);
}
```

Colocación de todo junto

El último paso consiste en llamar a `GetLanguages()` en la función `Main`. Busque `static void Main(string[] args)` y agregue estas líneas:

```
GetLanguages();
Console.ReadLine();
```

Ejecutar la aplicación de ejemplo

Eso es todo, ya está listo para ejecutar la aplicación de ejemplo. Desde la línea de comandos (o sesión de terminal), vaya al directorio del proyecto y ejecute:

```
dotnet run
```

Respuesta de muestra

Busque la abreviatura del país o región en esta [lista de idiomas](#).

```
{
  "translation": {
    "af": {
      "name": "Afrikaans",
      "nativeName": "Afrikaans",
      "dir": "ltr"
    },
    "ar": {
      "name": "Arabic",
      "nativeName": "العربية",
      "dir": "rtl"
    },
    ...
  },
  "transliteration": {
    "ar": {
      "name": "Arabic",
      "nativeName": "العربية",
      "scripts": [
        {
          "code": "Arab",
          "name": "Arabic",
          "nativeName": "العربية",
          "dir": "rtl",
          "toScripts": [
            {
              "code": "Latn",
              "name": "Latin",
              "nativeName": "اللاتينية",
              "dir": "ltr"
            }
          ]
        }
      ]
    },
    {
      "code": "Latn",
      "name": "Latin",
      "nativeName": "اللاتينية",
      "dir": "ltr",
      "toScripts": [
        {
          "code": "Arab",
          "name": "Arabic",
          "nativeName": "العربية",
          "dir": "rtl"
        }
      ]
    }
  ],
  ...
},
"dictionary": {
  "af": {
    "name": "Afrikaans",
    "nativeName": "Afrikaans",
    "dir": "ltr",
    "translations": [
      {
        "name": "English",
```

```

        "nativeName": "English",
        "dir": "ltr",
        "code": "en"
    }
]
},
"ar": {
    "name": "Arabic",
    "nativeName": "العربية",
    "dir": "rtl",
    "translations": [
        {
            "name": "English",
            "nativeName": "English",
            "dir": "ltr",
            "code": "en"
        }
    ]
}
},
...
}
}

```

Limpieza de recursos

Asegúrese de quitar cualquier información confidencial del código fuente de la aplicación de ejemplo como, por ejemplo, las claves de suscripción.

Pasos siguientes

Eche un vistazo a la referencia de API para comprender todo lo que puede hacer con Translator Text API.

[Referencia de API](#)

Requisitos previos

Esta guía de inicio rápido requiere:

- [JDK 7 o posterior](#)
- [Gradle](#)
- Una suscripción a Azure: [cree una cuenta gratuita](#).

Instalación

Creación de un recurso de Translator Text

Los servicios de Azure Cognitive Services se representan por medio de recursos de Azure a los que se suscribe. Cree un recurso para Translator Text mediante [Azure Portal](#) o la [CLI de Azure](#) en la máquina local. También puede:

- Obtener una [clave de prueba](#) válida durante siete días de forma gratuita. Después de registrarse, estará disponible en el sitio web de Azure.
- Vea un recurso existente en [Azure Portal](#).

Después de obtener una clave del recurso o la suscripción de evaluación, cree dos [variables de entorno](#):

- `TRANSLATOR_TEXT_SUBSCRIPTION_KEY`: la clave de suscripción del recurso de Translator Text.
- `TRANSLATOR_TEXT_ENDPOINT`: el punto de conexión global para Translator Text. Mediante `https://api.cognitive.microsofttranslator.com/`.

Inicialización de un proyecto con Gradle

Comencemos por crear un directorio de trabajo para este proyecto. Desde la línea de comandos (o terminal), ejecute este comando:

```
mkdir get-languages-sample
cd get-languages-sample
```

A continuación, va a inicializar un proyecto de Gradle. Este comando creará archivos de compilación esenciales para Gradle, el más importante, el `build.gradle.kts`, que se utiliza en tiempo de ejecución para crear y configurar la aplicación. Ejecute este comando desde el directorio de trabajo:

```
gradle init --type basic
```

Cuando se le solicite que elija un **DSL**, seleccione **Kotlin**.

Configuración del archivo de compilación

Localice `build.gradle.kts` y ábralo con el IDE o editor de texto favorito. A continuación, se copia en esta configuración de compilación:

```
plugins {
    java
    application
}
application {
    mainClassName = "GetLanguages"
}
repositories {
    mavenCentral()
}
dependencies {
    compile("com.squareup.okhttp:okhttp:2.5.0")
    compile("com.google.code.gson:gson:2.8.5")
}
```

Tenga en cuenta que este ejemplo tiene dependencias en OkHttp para las solicitudes HTTP, y Gson para administrar y analizar JSON. Si desea obtener más información acerca de las configuraciones de compilación, consulte [Creating New Gradle Builds](#) (Creación de nuevas compilaciones de Gradle).

Creación de un archivo Java

Vamos a crear una carpeta para la aplicación de ejemplo. En el directorio de trabajo, ejecute:

```
mkdir -p src/main/java
```

A continuación, en esta carpeta, cree un archivo denominado `GetLanguages.java`.

Importación de bibliotecas necesarias

Abra `GetLanguages.java` y agregue las siguientes instrucciones de importación:

```
import java.io.*;
import java.net.*;
import java.util.*;
import com.google.gson.*;
import com.squareup.okhttp.*;
```

Definición de variables

En primer lugar, deberá crear una clase pública para el proyecto:

```
public class GetLanguages {
    // All project code goes here...
}
```

Agregue estas líneas a la clase `GetLanguages`. Primero, se lee la clave de suscripción y el punto de conexión de las variables de entorno:

```
private static String subscriptionKey = System.getenv("TRANSLATOR_TEXT_SUBSCRIPTION_KEY");
private static String endpoint = System.getenv("TRANSLATOR_TEXT_ENDPOINT");
String url = endpoint + "/languages?api-version=3.0";
```

Si usa una suscripción a varios servicios de Cognitive Services, también debe incluir

`Ocp-Apim-Subscription-Region` en los parámetros de la solicitud. [Más información sobre la autenticación con la suscripción a varios servicios.](#)

Creación de un cliente y compilación de una solución

Agregue esta línea a la clase `GetLanguages` para crear una instancia de `OkHttpClient`:

```
// Instantiates the OkHttpClient.
OkHttpClient client = new OkHttpClient();
```

A continuación, vamos a crear la solicitud `GET`.

```
// This function performs a GET request.
public String Get() throws IOException {
    Request request = new Request.Builder()
        .url(url).get()
        .addHeader("Content-type", "application/json").build();
    Response response = client.newCall(request).execute();
    return response.body().string();
}
```

Creación de una función para analizar la respuesta

Esta sencilla función analiza y embellece la respuesta JSON del servicio Translator Text.


```
// This function prettifies the json response.
public static String prettify(String json_text) {
    JsonParser parser = new JsonParser();
    JsonElement json = parser.parse(json_text);
    Gson gson = new GsonBuilder().setPrettyPrinting().create();
    return gson.toJson(json);
}
```

Colocación de todo junto

El último paso consiste en realizar una solicitud y obtener una respuesta. Agregue estas líneas al proyecto:

```
public static void main(String[] args) {
    try {
        GetLanguages getLanguagesRequest = new GetLanguages();
        String response = getLanguagesRequest.Get();
        System.out.println(prettify(response));
    } catch (Exception e) {
        System.out.println(e);
    }
}
```

Ejecutar la aplicación de ejemplo

Eso es todo, ya está listo para ejecutar la aplicación de ejemplo. Desde la línea de comandos (o sesión de terminal), navegue hasta la raíz del directorio de trabajo y ejecute:

```
gradle build
```

Cuando la compilación se complete, ejecute lo siguiente:

```
gradle run
```

Respuesta de muestra

Busque la abreviatura del país o región en esta [lista de idiomas](#).

Se devuelve una respuesta correcta en JSON, tal como se muestra en el siguiente ejemplo:

```
{
  "translation": {
    "af": {
      "name": "Afrikaans",
      "nativeName": "Afrikaans",
      "dir": "ltr"
    },
    "ar": {
      "name": "Arabic",
      "nativeName": "العربية",
      "dir": "rtl"
    },
    ...
  },
  "transliteration": {
    "ar": {
      "name": "Arabic",
      "nativeName": "العربية",
```

```

"scripts": [
  {
    "code": "Arab",
    "name": "Arabic",
    "nativeName": "العربية",
    "dir": "rtl",
    "toScripts": [
      {
        "code": "Latn",
        "name": "Latin",
        "nativeName": "اللاتينية",
        "dir": "ltr"
      }
    ]
  },
  {
    "code": "Latn",
    "name": "Latin",
    "nativeName": "اللاتينية",
    "dir": "ltr",
    "toScripts": [
      {
        "code": "Arab",
        "name": "Arabic",
        "nativeName": "العربية",
        "dir": "rtl"
      }
    ]
  }
],
...
},
"dictionary": {
  "af": {
    "name": "Afrikaans",
    "nativeName": "Afrikaans",
    "dir": "ltr",
    "translations": [
      {
        "name": "English",
        "nativeName": "English",
        "dir": "ltr",
        "code": "en"
      }
    ]
  },
  "ar": {
    "name": "Arabic",
    "nativeName": "العربية",
    "dir": "rtl",
    "translations": [
      {
        "name": "English",
        "nativeName": "English",
        "dir": "ltr",
        "code": "en"
      }
    ]
  },
  ...
}
}

```

Pasos siguientes

Eche un vistazo a la [referencia de API](#) para comprender todo lo que puede hacer con Translator Text API.

Requisitos previos

Esta guía de inicio rápido requiere:

- [Python 2.7.x o 3.x](#)
- Una suscripción a Azure: [cree una cuenta gratuita](#).

Instalación

Creación de un recurso de Translator Text

Los servicios de Azure Cognitive Services se representan por medio de recursos de Azure a los que se suscribe. Cree un recurso para Translator Text mediante [Azure Portal](#) o la [CLI de Azure](#) en la máquina local. También puede:

- Obtener una [clave de prueba](#) válida durante siete días de forma gratuita. Después de registrarse, estará disponible en el sitio web de Azure.
- Vea un recurso existente en [Azure Portal](#).

Después de obtener una clave del recurso o la suscripción de evaluación, cree dos [variables de entorno](#):

- `TRANSLATOR_TEXT_SUBSCRIPTION_KEY`: la clave de suscripción del recurso de Translator Text.
- `TRANSLATOR_TEXT_ENDPOINT`: el punto de conexión global para Translator Text. Mediante `https://api.cognitive.microsofttranslator.com/`.

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 `get-languages.py`.

```
# -*- coding: utf-8 -*-
import os, requests, uuid, json
```

NOTE

Si no ha usado estos módulos deberá instalarlos antes de ejecutar el programa. Para instalar estos paquetes, ejecute:

```
pip install requests uuid
```

El primer comentario le indica al intérprete de Python que debe usar la codificación UTF-8. Después, se importan los módulos necesarios para leer la clave de suscripción desde una variable de entorno, construir la solicitud HTTP, crear un identificador único y controlar la respuesta JSON que devuelve Translator Text API.

Establecimiento del punto de conexión y la ruta de acceso

En este ejemplo se intenta leer el punto de conexión de Translator Text desde una variable de entorno:

`TRANSLATOR_TEXT_ENDPOINT`. Si no está familiarizado con las variables de entorno, puede establecer `endpoint` como una cadena y convertir en comentario la instrucción condicional.

```
endpoint_var_name = 'TRANSLATOR_TEXT_ENDPOINT'
if not endpoint_var_name in os.environ:
    raise Exception('Please set/export the environment variable: {}'.format(endpoint_var_name))
endpoint = os.environ[endpoint_var_name]
```

El punto de conexión global de Translator Text se establece como `endpoint` . `path` establece la ruta de `languages` e identifica que deseamos usar la versión 3 de la API.

NOTE

Para más información sobre los puntos de conexión, las rutas y los parámetros de la solicitud, consulte [Translator Text API 3.0: Idiomas](#).

```
path = '/languages?api-version=3.0'
constructed_url = endpoint + path
```

Incorporación de encabezados

La solicitud para obtener los idiomas compatibles no requiere autenticación. Establezca `Content-type` como `application/json` y agregue `X-ClientTraceId` para identificar de forma única la solicitud.

Copie este fragmento de código en el proyecto:

```
headers = {
    'Content-type': 'application/json',
    'X-ClientTraceId': str(uuid.uuid4())
}
```

Si usa una suscripción a varios servicios de Cognitive Services, también debe incluir

`Ocp-Apim-Subscription-Region` en los parámetros de la solicitud. [Más información sobre la autenticación con la suscripción a varios servicios](#).

Crear una solicitud para obtener una lista de idiomas admitidos

Vamos a crear una solicitud GET mediante el módulo `requests` , que usa DOS argumentos: la dirección URL concatenada y los encabezados de solicitud:

```
request = requests.get(constructed_url, headers=headers)
response = request.json()
```

Impresión de la respuesta

El último paso es imprimir los resultados. Este fragmento de código adorna los resultados mediante una clasificación de las claves, el establecimiento de una sangría y la declaración de separadores de elementos y de claves.

```
print(json.dumps(response, sort_keys=True, indent=4,
                  ensure_ascii=False, separators=(',', ' ')))
```

Colocación de todo junto

Eso es todo, ha creado un sencillo programa que llama a Translator Text API y devuelve una respuesta JSON. Ahora es el momento de ejecutar el programa:

```
python get-languages.py
```

Si desea comparar su código con el nuestro, el ejemplo completo está disponible en [GitHub](#).

Respuesta de muestra

Busque la abreviatura del país o región en esta [lista de idiomas](#).

Este ejemplo se ha truncado para mostrar un fragmento del resultado:

```
{
  "translation": {
    "af": {
      "name": "Afrikaans",
      "nativeName": "Afrikaans",
      "dir": "ltr"
    },
    "ar": {
      "name": "Arabic",
      "nativeName": "العربية",
      "dir": "rtl"
    },
    ...
  },
  "transliteration": {
    "ar": {
      "name": "Arabic",
      "nativeName": "العربية",
      "scripts": [
        {
          "code": "Arab",
          "name": "Arabic",
          "nativeName": "العربية",
          "dir": "rtl",
          "toScripts": [
            {
              "code": "Latn",
              "name": "Latin",
              "nativeName": "اللاتينية",
              "dir": "ltr"
            }
          ]
        }
      ],
      {
        "code": "Latn",
        "name": "Latin",
        "nativeName": "اللاتينية",
        "dir": "ltr",
        "toScripts": [
          {
            "code": "Arab",
            "name": "Arabic",
            "nativeName": "العربية",
            "dir": "rtl"
          }
        ]
      }
    ],
    ...
  },
  "dictionary": {
    "af": {
      "name": "Afrikaans",
      "nativeName": "Afrikaans",
      "dir": "ltr",
      "translations": [
        {
          "name": "English".
```

```

        "name": "English",
        "nativeName": "English",
        "dir": "ltr",
        "code": "en"
    }
]
},
"ar": {
    "name": "Arabic",
    "nativeName": "العربية",
    "dir": "rtl",
    "translations": [
        {
            "name": "English",
            "nativeName": "English",
            "dir": "ltr",
            "code": "en"
        }
    ]
}
...
}

```

Limpieza de recursos

Si ha codificado de forma rígida la clave de suscripción en el programa, asegúrese de quitarla cuando haya terminado con esta guía de inicio rápido.

Pasos siguientes

Eche un vistazo a la referencia de API para comprender todo lo que puede hacer con Translator Text API.

[Referencia de API](#)

Requisitos previos

Esta guía de inicio rápido requiere:

- [Node 8.12.x o posterior](#)
- Una suscripción a Azure: [cree una cuenta gratuita](#).

Instalación

Creación de un recurso de Translator Text

Los servicios de Azure Cognitive Services se representan por medio de recursos de Azure a los que se suscribe. Cree un recurso para Translator Text mediante [Azure Portal](#) o la [CLI de Azure](#) en la máquina local. También puede:

- Obtener una [clave de prueba](#) válida durante siete días de forma gratuita. Después de registrarse, estará disponible en el sitio web de Azure.
- Vea un recurso existente en [Azure Portal](#).

Después de obtener una clave del recurso o la suscripción de evaluación, cree dos [variables de entorno](#):

- `TRANSLATOR_TEXT_SUBSCRIPTION_KEY`: la clave de suscripción del recurso de Translator Text.
- `TRANSLATOR_TEXT_ENDPOINT`: el punto de conexión global para Translator Text. Mediante `https://api.cognitive.microsofttranslator.com/`.

Creación de un proyecto e importación de los módulos necesarios

Cree un proyecto con su IDE o editor favorito. A continuación, copie este fragmento de código en un archivo llamado `get-languages.js`.

```
const request = require('request');
const uuidv4 = require('uuid/v4');
```

NOTE

Si no ha usado estos módulos deberá instalarlos antes de ejecutar el programa. Para instalar estos paquetes, ejecute:

```
npm install request uuidv4.
```

Estos módulos son necesarios para construir la solicitud HTTP y crear un identificador único para el encabezado `'X-ClientTraceId'`.

Establecimiento del punto de conexión

En este ejemplo se intenta leer el punto de conexión de Translator Text desde una variable de entorno:

`TRANSLATOR_TEXT_ENDPOINT`. Si no está familiarizado con las variables de entorno, puede establecer `endpoint` como una cadena y convertir en comentario la instrucción condicional.

```
lorum ipsum
```

Configuración de la solicitud

El método `request()`, disponible mediante el módulo de solicitud, nos permite pasar el método HTTP, la dirección URL, los parámetros de la solicitud, los encabezados y el cuerpo de JSON como un objeto `options`. En este fragmento de código, vamos a configurar la solicitud:

NOTE

Para más información sobre los puntos de conexión, las rutas y los parámetros de la solicitud, consulte [Translator Text API 3.0: Idiomas](#).

```
let options = {
  method: 'GET',
  baseUrl: endpoint,
  url: 'languages',
  qs: {
    'api-version': '3.0',
  },
  headers: {
    'Content-type': 'application/json',
    'X-ClientTraceId': uuidv4().toString()
  },
  json: true,
};
```

Si usa una suscripción a varios servicios de Cognitive Services, también debe incluir

`Ocp-Apim-Subscription-Region` en los parámetros de la solicitud. [Más información sobre la autenticación con la suscripción a varios servicios](#).

Realización de solicitud e impresión de la respuesta

A continuación, vamos a crear una solicitud mediante el método `request()`. Toma el objeto `options` que se creó en la sección anterior como el primer argumento y, a continuación, imprime la respuesta JSON embellecida.

```
request(options, function(err, res, body){
  console.log(JSON.stringify(body, null, 4));
});
```

NOTE

En este ejemplo, vamos a definir la solicitud HTTP en el objeto `options`. Sin embargo, el módulo de solicitud también admite métodos de conveniencia como `.post` y `.get`. Para más información, consulte [métodos de conveniencia](#).

Colocación de todo junto

Eso es todo, ha creado un sencillo programa que llama a Translator Text API y devuelve una respuesta JSON. Ahora es el momento de ejecutar el programa:

```
node get-languages.js
```

Si desea comparar su código con el nuestro, el ejemplo completo está disponible en [GitHub](#).

Respuesta de muestra

Busque la abreviatura del país o región en esta [lista de idiomas](#).

Este ejemplo se ha truncado para mostrar un fragmento del resultado:

```
{
  "translation": {
    "af": {
      "name": "Afrikaans",
      "nativeName": "Afrikaans",
      "dir": "ltr"
    },
    "ar": {
      "name": "Arabic",
      "nativeName": "العربية",
      "dir": "rtl"
    },
    ...
  },
  "transliteration": {
    "ar": {
      "name": "Arabic",
      "nativeName": "العربية",
      "scripts": [
        {
          "code": "Arab",
          "name": "Arabic",
          "nativeName": "العربية",
          "dir": "rtl",
          "toScripts": [
            {
              "code": "Latn",
              "name": "Latin",
              "nativeName": "اللاتينية",
              "dir": "ltr"
            }
          ]
        }
      ]
    }
  }
}
```



```

    },
    {
      "code": "Latn",
      "name": "Latin",
      "nativeName": "اللاتينية",
      "dir": "ltr",
      "toScripts": [
        {
          "code": "Arab",
          "name": "Arabic",
          "nativeName": "العربية",
          "dir": "rtl"
        }
      ]
    }
  ]
},
...
},
"dictionary": {
  "af": {
    "name": "Afrikaans",
    "nativeName": "Afrikaans",
    "dir": "ltr",
    "translations": [
      {
        "name": "English",
        "nativeName": "English",
        "dir": "ltr",
        "code": "en"
      }
    ]
  },
  "ar": {
    "name": "Arabic",
    "nativeName": "العربية",
    "dir": "rtl",
    "translations": [
      {
        "name": "English",
        "nativeName": "English",
        "dir": "ltr",
        "code": "en"
      }
    ]
  },
  ...
}
}

```

Limpieza de recursos

Si ha codificado de forma rígida la clave de suscripción en el programa, asegúrese de quitarla cuando haya terminado con esta guía de inicio rápido.

Pasos siguientes

Eche un vistazo a la referencia de API para comprender todo lo que puede hacer con Translator Text API.

[Referencia de API](#)

Requisitos previos

Esta guía de inicio rápido requiere:

- [Go](#)
- Una suscripción a Azure: [cree una cuenta gratuita](#).

Instalación

Creación de un recurso de Translator Text

Los servicios de Azure Cognitive Services se representan por medio de recursos de Azure a los que se suscribe. Cree un recurso para Translator Text mediante [Azure Portal](#) o la [CLI de Azure](#) en la máquina local. También puede:

- Obtener una [clave de prueba](#) válida durante siete días de forma gratuita. Después de registrarse, estará disponible en el sitio web de Azure.
- Vea un recurso existente en [Azure Portal](#).

Después de obtener una clave del recurso o la suscripción de evaluación, cree dos [variables de entorno](#):

- `TRANSLATOR_TEXT_SUBSCRIPTION_KEY`: la clave de suscripción del recurso de Translator Text.
- `TRANSLATOR_TEXT_ENDPOINT`: el punto de conexión global para Translator Text. Mediante `https://api.cognitive.microsofttranslator.com/`.

Creación de un proyecto e importación de los módulos necesarios

Cree un proyecto de Go con su IDE o editor favoritos, o bien en una nueva carpeta en el escritorio. A continuación, copie este fragmento de código en un archivo llamado `get-languages.go` en el proyecto o carpeta.

```
package main

import (
    "encoding/json"
    "fmt"
    "log"
    "net/http"
    "net/url"
    "os"
)
```

Creación de la función main

Vamos a crear la función main de nuestra aplicación. Observará que es una sola línea de código. Eso se debe a que estamos creando una sola función para obtener e imprimir la lista de idiomas admitidos de Translator Text.

En este ejemplo se intenta leer el punto de conexión de Translator Text desde una variable de entorno:

`TRANSLATOR_TEXT_ENDPOINT`. Si no está familiarizado con las variables de entorno, puede establecer `endpoint` como una cadena y convertir en comentario la instrucción condicional.

Copie este código en el proyecto:

```
func main() {
    if "" == os.Getenv("TRANSLATOR_TEXT_ENDPOINT") {
        log.Fatal("Please set/export the environment variable TRANSLATOR_TEXT_ENDPOINT.")
    }
    endpoint := os.Getenv("TRANSLATOR_TEXT_ENDPOINT")
    uri := endpoint + "/languages?api-version=3.0"
    getLanguages(uri)
}
```

Creación de una función para obtener una lista de idiomas admitidos

Vamos a crear una función para obtener una lista de idiomas admitidos.

```
func getLanguages(uri string) {  
    /*  
     * In the next few sections, we'll add code to this  
     * function to make a request and handle the response.  
     */  
}
```

A continuación, vamos a construir la dirección URL. La dirección URL se crea mediante los métodos `Parse()` y `Query()`.

Copie este código en la función `getLanguages`.

```
// Build the request URL. See: https://golang.org/pkg/net/url/#example_URL_Parse  
u, _ := url.Parse(uri)  
q := u.Query()  
u.RawQuery = q.Encode()
```

NOTE

Para más información sobre los puntos de conexión, las rutas y los parámetros de la solicitud, consulte [Translator Text API 3.0: Idiomas](#).

Compilar la solicitud

Ahora que se ha codificado el cuerpo de la solicitud como JSON, puede crear la solicitud POST y llamar a Translator Text API.

```
// Build the HTTP GET request  
req, err := http.NewRequest("GET", u.String(), nil)  
if err != nil {  
    log.Fatal(err)  
}  
// Add required headers  
req.Header.Add("Content-Type", "application/json")  
  
// Call the Translator Text API  
res, err := http.DefaultClient.Do(req)  
if err != nil {  
    log.Fatal(err)  
}
```

Si usa una suscripción a varios servicios de Cognitive Services, también debe incluir

`Ocp-Apim-Subscription-Region` en los parámetros de la solicitud. [Más información sobre la autenticación con la suscripción a varios servicios](#).

Control e impresión de la respuesta

Agregue este código a la función `getLanguages` para decodificar la respuesta JSON y luego dé formato al resultado e imprímalo.

```
// Decode the JSON response
var result interface{}
if err := json.NewDecoder(res.Body).Decode(&result); err != nil {
    log.Fatal(err)
}
// Format and print the response to terminal
prettyJSON, _ := json.MarshalIndent(result, "", " ")
fmt.Printf("%s\n", prettyJSON)
```

Colocación de todo junto

Eso es todo, ha creado un sencillo programa que llama a Translator Text API y devuelve una respuesta JSON. Ahora es el momento de ejecutar el programa:

```
go run get-languages.go
```

Si desea comparar su código con el nuestro, el ejemplo completo está disponible en [GitHub](#).

Respuesta de muestra

Busque la abreviatura del país o región en esta [lista de idiomas](#).

Se devuelve una respuesta correcta en JSON, tal como se muestra en el siguiente ejemplo:

```
{
  "dictionary": {
    "af": {
      "name": "Afrikaans",
      "nativeName": "Afrikaans",
      "dir": "ltr",
      "translations": [
        {
          "name": "English",
          "nativeName": "English",
          "dir": "ltr",
          "code": "en"
        }
      ]
    },
    "ar": {
      "name": "Arabic",
      "nativeName": "العربية",
      "dir": "rtl",
      "translations": [
        {
          "name": "English",
          "nativeName": "English",
          "dir": "ltr",
          "code": "en"
        }
      ]
    },
    ...
  },
  "translation": {
    "af": {
      "name": "Afrikaans",
      "nativeName": "Afrikaans",
      "dir": "ltr"
    },
    "ar": {
      "name": "Arabic",
```

```

        "nativeName": "العربية",
        "dir": "rtl"
    },
    ...
  },
  "transliteration": {
    "ar": {
      "name": "Arabic",
      "nativeName": "العربية",
      "scripts": [
        {
          "code": "Arab",
          "name": "Arabic",
          "nativeName": "العربية",
          "dir": "rtl",
          "toScripts": [
            {
              "code": "Latn",
              "name": "Latin",
              "nativeName": "اللاتينية",
              "dir": "ltr"
            }
          ]
        }
      ]
    },
    {
      "code": "Latn",
      "name": "Latin",
      "nativeName": "اللاتينية",
      "dir": "ltr",
      "toScripts": [
        {
          "code": "Arab",
          "name": "Arabic",
          "nativeName": "العربية",
          "dir": "rtl"
        }
      ]
    }
  ]
},
...
}
}

```

Pasos siguientes

Eche un vistazo a la referencia de API para comprender todo lo que puede hacer con Translator Text API.

[Referencia de API](#)

Consulte también

- [Traducir texto](#)
- [Transliterar texto](#)
- [Identificar el idioma de entrada](#)
- [Obtener traducciones alternativas](#)
- [Determinar la longitud de las oraciones de una entrada](#)

Inicio rápido: Uso de la API Translator Text para determinar las longitudes de oración

13/01/2020 • 36 minutes to read • [Edit Online](#)

En este inicio rápido, aprenderá a determinar la longitud de las frases mediante la API Translator Text.

En esta guía de inicio rápido, se requiere una [cuenta de Azure Cognitive Services](#) con un recurso de Translator Text. Si no tiene una cuenta, puede usar la [evaluación gratuita](#) para obtener una clave de suscripción.

Requisitos previos

Esta guía de inicio rápido requiere:

- C# 7.1 o posterior
- [SDK de .NET](#)
- [Paquete NuGet de Json.NET](#)
- [Visual Studio](#), [Visual Studio Code](#) o su editor favorito de código
- Una suscripción a Azure: [cree una cuenta gratuita](#).

Instalación

Creación de un recurso de Translator Text

Los servicios de Azure Cognitive Services se representan por medio de recursos de Azure a los que se suscribe. Cree un recurso para Translator Text mediante [Azure Portal](#) o la [CLI de Azure](#) en la máquina local. También puede:

- Obtener una [clave de prueba](#) válida durante siete días de forma gratuita. Después de registrarse, estará disponible en el sitio web de Azure.
- Vea un recurso existente en [Azure Portal](#).

Después de obtener una clave del recurso o la suscripción de evaluación, cree dos [variables de entorno](#):

- `TRANSLATOR_TEXT_SUBSCRIPTION_KEY`: la clave de suscripción del recurso de Translator Text.
- `TRANSLATOR_TEXT_ENDPOINT`: el punto de conexión global para Translator Text. Mediante `https://api.cognitive.microsofttranslator.com/`.

Creación de un proyecto de .NET Core

Abra un nuevo símbolo del sistema (o una sesión de terminal) y ejecute estos comandos:

```
dotnet new console -o sentences-sample
cd sentences-sample
```

El primer comando hace dos cosas. Crea una nueva aplicación de consola .NET y crea un directorio denominado `sentences-sample`. El segundo comando cambia al directorio del proyecto.

A continuación, deberá instalar Json.Net. En el directorio del proyecto, ejecute:

```
dotnet add package Newtonsoft.Json --version 11.0.2
```

Selección de la versión del lenguaje C#

Esta guía de inicio rápido requiere C# 7.1 o posterior. Hay algunas maneras de cambiar la versión de C# del proyecto. En esta guía le mostraremos cómo ajustar el archivo `sentences-sample.csproj`. Para todas las opciones disponibles, como el cambio de idioma en Visual Studio, consulte [Selección de la versión del lenguaje C#](#).

Abra el proyecto y, después, abra `sentences-sample.csproj`. Asegúrese de que `LangVersion` está establecido en 7.1 o posterior. Si no hay un grupo de propiedades para la versión del lenguaje, añada estas líneas:

```
<PropertyGroup>
  <LangVersion>7.1</LangVersion>
</PropertyGroup>
```

Incorporación de los espacios de nombres necesarios al proyecto

El comando `dotnet new console` que ejecutó anteriormente creó un proyecto que incluía `Program.cs`. En este archivo es donde deberá colocar el código de la aplicación. Abra `Program.cs` y reemplace las instrucciones `using` existentes. Estas instrucciones garantizan que tiene acceso a todos los tipos necesarios para compilar y ejecutar la aplicación de ejemplo.

```
using System;
using System.Net.Http;
using System.Text;
using System.Threading.Tasks;
// Install Newtonsoft.Json with NuGet
using Newtonsoft.Json;
```

Creación de clases para la respuesta JSON

A continuación, vamos a crear una clase que se utiliza al deserializar la respuesta JSON devuelta por Translator Text API.

```
/// <summary>
/// The C# classes that represents the JSON returned by the Translator Text API.
/// </summary>
public class BreakSentenceResult
{
    public int[] SentLen { get; set; }
    public DetectedLanguage DetectedLanguage { get; set; }
}

public class DetectedLanguage
{
    public string Language { get; set; }
    public float Score { get; set; }
}
```

Obtención de información de la suscripción a partir de las variables de entorno

Agregue las líneas siguientes a la clase `Program`. Estas líneas leen la clave de suscripción y el punto de conexión de las variables de entorno y devuelven un error si se produce algún problema.

```
private const string key_var = "TRANSLATOR_TEXT_SUBSCRIPTION_KEY";
private static readonly string subscriptionKey = Environment.GetEnvironmentVariable(key_var);

private const string endpoint_var = "TRANSLATOR_TEXT_ENDPOINT";
private static readonly string endpoint = Environment.GetEnvironmentVariable(endpoint_var);

static Program()
{
    if (null == subscriptionKey)
    {
        throw new Exception("Please set/export the environment variable: " + key_var);
    }
    if (null == endpoint)
    {
        throw new Exception("Please set/export the environment variable: " + endpoint_var);
    }
}
// The code in the next section goes here.
```

Creación de una función para determinar la longitud de la frase

En la clase `Program`, cree una función denominada `BreakSentenceRequest()`. Esta función toma cuatro argumentos: `subscriptionKey`, `endpoint`, `route` y `inputText`.

```
static public async Task BreakSentenceRequest(string subscriptionKey, string endpoint, string route, string
inputText)
{
    /*
    * The code for your call to the translation service will be added to this
    * function in the next few sections.
    */
}
```

Serialización de la solicitud de interrupción de la frase

A continuación, deberá crear y serializar el objeto JSON que incluye el texto. Tenga en cuenta que puede pasar más de un objeto en la matriz `body`.

```
object[] body = new object[] { new { Text = inputText } };
var requestBody = JsonConvert.SerializeObject(body);
```

Creación de una instancia de cliente y realización de una solicitud

Estas líneas crean instancias de `HttpClient` y `HttpRequestMessage`:

```
using (var client = new HttpClient())
using (var request = new HttpRequestMessage())
{
    // In the next few sections you'll add code to construct the request.
}
```

Construcción de la solicitud e impresión de la respuesta

Dentro de `HttpRequestMessage`:

- Declarará el método HTTP

- Construirá el URI de solicitud
- Insertará el cuerpo de la solicitud (el objeto JSON serializado)
- Agregará los encabezados necesarios
- Realizará una solicitud asíncrona
- Impresión de la respuesta

Agregue este código a `HttpRequestMessage`:

```
// Build the request.
// Set the method to POST
request.Method = HttpMethod.Post;
// Construct the URI and add headers.
request.RequestUri = new Uri(endpoint + route);
request.Content = new StringContent(requestBody, Encoding.UTF8, "application/json");
request.Headers.Add("Ocp-Apim-Subscription-Key", subscriptionKey);

// Send the request and get response.
HttpResponseMessage response = await client.SendAsync(request).ConfigureAwait(false);
// Read response as a string.
string result = await response.Content.ReadAsStringAsync();
// Deserialize the response using the classes created earlier.
BreakSentenceResult[] deserializedOutput = JsonConvert.DeserializeObject<BreakSentenceResult[]>(result);
foreach (BreakSentenceResult o in deserializedOutput)
{
    Console.WriteLine("The detected language is '{0}'. Confidence is: {1}.", o.DetectedLanguage.Language,
o.DetectedLanguage.Score);
    Console.WriteLine("The first sentence length is: {0}", o.SentLen[0]);
}
```

Si usa una suscripción a varios servicios de Cognitive Services, también debe incluir

`Ocp-Apim-Subscription-Region` en los parámetros de la solicitud. [Más información sobre la autenticación con la suscripción a varios servicios.](#)

Colocación de todo junto

El último paso consiste en llamar a `BreakSentenceRequest()` en la función `Main`. Busque `static void Main(string[] args)` y reemplázelo por este código:

```
static async Task Main(string[] args)
{
    // This is our main function.
    // Output languages are defined in the route.
    // For a complete list of options, see API reference.
    string route = "/breaksentence?api-version=3.0";
    // Feel free to use any string.
    string breakSentenceText = @"How are you doing today? The weather is pretty pleasant. Have you been to
the movies lately?";
    await BreakSentenceRequest(subscriptionKey, endpoint, route, breakSentenceText);
    Console.WriteLine("Press any key to continue.");
    Console.ReadKey();
}
```

Observará que en `Main`, está declarando `subscriptionKey`, `endpoint` y `route`, y el texto para evaluar `breakSentenceText`.

Ejecutar la aplicación de ejemplo

Eso es todo, ya está listo para ejecutar la aplicación de ejemplo. Desde la línea de comandos (o sesión de terminal), vaya al directorio del proyecto y ejecute:

```
dotnet run
```

Respuesta de muestra

Después de ejecutar el ejemplo, debería ver lo siguiente impreso en el terminal:

```
The detected language is \'en\'. Confidence is: 1.  
The first sentence length is: 25
```

Este mensaje se crea a partir del JSON sin formato, y tendrá un aspecto similar al siguiente:

```
[  
  {  
    "detectedLanguage":  
    {  
      "language": "en",  
      "score": 1.0  
    },  
    "sentLen": [25, 32, 35]  
  }  
]
```

Limpieza de recursos

Asegúrese de quitar cualquier información confidencial del código fuente de la aplicación de ejemplo como, por ejemplo, las claves de suscripción.

Pasos siguientes

Eche un vistazo a la referencia de API para comprender todo lo que puede hacer con Translator Text API.

[Referencia de API](#)

Requisitos previos

Esta guía de inicio rápido requiere:

- [JDK 7 o posterior](#)
- [Gradle](#)
- Una suscripción a Azure: [cree una cuenta gratuita](#).

Instalación

Creación de un recurso de Translator Text

Los servicios de Azure Cognitive Services se representan por medio de recursos de Azure a los que se suscribe. Cree un recurso para Translator Text mediante [Azure Portal](#) o la [CLI de Azure](#) en la máquina local. También puede:

- Obtener una [clave de prueba](#) válida durante siete días de forma gratuita. Después de registrarse, estará disponible en el sitio web de Azure.
- Vea un recurso existente en [Azure Portal](#).

Después de obtener una clave del recurso o la suscripción de evaluación, cree dos [variables de entorno](#):

- `TRANSLATOR_TEXT_SUBSCRIPTION_KEY`: la clave de suscripción del recurso de Translator Text.

- `TRANSLATOR_TEXT_ENDPOINT`: el punto de conexión global para Translator Text. Mediante `https://api.cognitive.microsofttranslator.com/`.

Inicialización de un proyecto con Gradle

Comencemos por crear un directorio de trabajo para este proyecto. Desde la línea de comandos (o terminal), ejecute este comando:

```
mkdir length-sentence-sample
cd length-sentence-sample
```

A continuación, va a inicializar un proyecto de Gradle. Este comando creará archivos de compilación esenciales para Gradle, el más importante, el `build.gradle.kts`, que se utiliza en tiempo de ejecución para crear y configurar la aplicación. Ejecute este comando desde el directorio de trabajo:

```
gradle init --type basic
```

Cuando se le solicite que elija un **DSL**, seleccione **Kotlin**.

Configuración del archivo de compilación

Localice `build.gradle.kts` y ábralo con el IDE o editor de texto favorito. A continuación, se copia en esta configuración de compilación:

```
plugins {
    java
    application
}
application {
    mainClassName = "BreakSentence"
}
repositories {
    mavenCentral()
}
dependencies {
    compile("com.squareup.okhttp:okhttp:2.5.0")
    compile("com.google.code.gson:gson:2.8.5")
}
```

Tenga en cuenta que este ejemplo tiene dependencias en OkHttp para las solicitudes HTTP, y Gson para administrar y analizar JSON. Si desea obtener más información acerca de las configuraciones de compilación, consulte [Creating New Gradle Builds](#) (Creación de nuevas compilaciones de Gradle).

Creación de un archivo Java

Vamos a crear una carpeta para la aplicación de ejemplo. En el directorio de trabajo, ejecute:

```
mkdir -p src/main/java
```

A continuación, en esta carpeta, cree un archivo denominado `BreakSentence.java`.

Importación de bibliotecas necesarias

Abra `BreakSentence.java` y agregue las siguientes instrucciones de importación:

```
import java.io.*;
import java.net.*;
import java.util.*;
import com.google.gson.*;
import com.squareup.okhttp.*;
```

Definición de variables

En primer lugar, deberá crear una clase pública para el proyecto:

```
public class BreakSentence {
    // All project code goes here...
}
```

Agregue estas líneas a la clase `BreakSentence`. Primero, se lee la clave de suscripción y el punto de conexión de las variables de entorno. A continuación, observe que junto con la `api-version`, puede definir el idioma de entrada. En este ejemplo es el inglés.

```
private static String subscriptionKey = System.getenv("TRANSLATOR_TEXT_SUBSCRIPTION_KEY");
private static String endpoint = System.getenv("TRANSLATOR_TEXT_ENDPOINT");
String url = endpoint + "/breaksentence?api-version=3.0&language=en";
```

Si usa una suscripción a varios servicios de Cognitive Services, también debe incluir

`Ocp-Apim-Subscription-Region` en los parámetros de la solicitud. [Más información sobre la autenticación con la suscripción a varios servicios.](#)

Creación de un cliente y compilación de una solución

Agregue esta línea a la clase `BreakSentence` para crear una instancia de `OkHttpClient`:

```
// Instantiates the OkHttpClient.
OkHttpClient client = new OkHttpClient();
```

A continuación, vamos a crear la solicitud POST. Si lo desea, cambie el texto. El texto debe escaparse.

```
// This function performs a POST request.
public String Post() throws IOException {
    MediaType mediaType = MediaType.parse("application/json");
    RequestBody body = RequestBody.create(mediaType,
        "[{\n\t\"Text\": \"How are you? I am fine. What did you do today?\n}]]");
    Request request = new Request.Builder()
        .url(url).post(body)
        .addHeader("Ocp-Apim-Subscription-Key", subscriptionKey)
        .addHeader("Content-type", "application/json").build();
    Response response = client.newCall(request).execute();
    return response.body().string();
}
```

Creación de una función para analizar la respuesta

Esta sencilla función analiza y embellece la respuesta JSON del servicio Translator Text.

```
// This function prettifies the json response.
public static String prettify(String json_text) {
    JsonParser parser = new JsonParser();
    JsonElement json = parser.parse(json_text);
    Gson gson = new GsonBuilder().setPrettyPrinting().create();
    return gson.toJson(json);
}
```

Colocación de todo junto

El último paso consiste en realizar una solicitud y obtener una respuesta. Agregue estas líneas al proyecto:

```
public static void main(String[] args) {
    try {
        BreakSentence breakSentenceRequest = new BreakSentence();
        String response = BreakSentenceRequest.Post();
        System.out.println(prettify(response));
    } catch (Exception e) {
        System.out.println(e);
    }
}
```

Ejecutar la aplicación de ejemplo

Eso es todo, ya está listo para ejecutar la aplicación de ejemplo. Desde la línea de comandos (o sesión de terminal), navegue hasta la raíz del directorio de trabajo y ejecute:

```
gradle build
```

Cuando la compilación se complete, ejecute lo siguiente:

```
gradle run
```

Respuesta de muestra

Se devuelve una respuesta correcta en JSON, tal como se muestra en el siguiente ejemplo:

```
[
  {
    "detectedLanguage": {
      "language": "en",
      "score": 1.0
    },
    "sentLen": [
      13,
      11,
      22
    ]
  }
]
```

Pasos siguientes

Eche un vistazo a la referencia de API para comprender todo lo que puede hacer con Translator Text API.

Requisitos previos

Esta guía de inicio rápido requiere:

- [Python 2.7.x o 3.x](#)
- Una suscripción a Azure: [cree una cuenta gratuita](#).

Instalación

Creación de un recurso de Translator Text

Los servicios de Azure Cognitive Services se representan por medio de recursos de Azure a los que se suscribe. Cree un recurso para Translator Text mediante [Azure Portal](#) o la [CLI de Azure](#) en la máquina local. También puede:

- Obtener una [clave de prueba](#) válida durante siete días de forma gratuita. Después de registrarse, estará disponible en el sitio web de Azure.
- Vea un recurso existente en [Azure Portal](#).

Después de obtener una clave del recurso o la suscripción de evaluación, cree dos [variables de entorno](#):

- `TRANSLATOR_TEXT_SUBSCRIPTION_KEY`: la clave de suscripción del recurso de Translator Text.
- `TRANSLATOR_TEXT_ENDPOINT`: el punto de conexión global para Translator Text. Mediante `https://api.cognitive.microsofttranslator.com/`.

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 `sentence-length.py`.

```
# -*- coding: utf-8 -*-  
import os, requests, uuid, json
```

NOTE

Si no ha usado estos módulos deberá instalarlos antes de ejecutar el programa. Para instalar estos paquetes, ejecute:

```
pip install requests uuid .
```

El primer comentario le indica al intérprete de Python que debe usar la codificación UTF-8. Después, se importan los módulos necesarios para leer la clave de suscripción desde una variable de entorno, construir la solicitud HTTP, crear un identificador único y controlar la respuesta JSON que devuelve Translator Text API.

Establecimiento de la clave de suscripción, el punto de conexión y la ruta de acceso

En este ejemplo se intenta leer la clave de suscripción y el punto de conexión de Translator Text desde las variables de entorno `TRANSLATOR_TEXT_KEY` y `TRANSLATOR_TEXT_ENDPOINT`. Si no está familiarizado con las variables de entorno, puede establecer `subscription_key` y `endpoint` como cadenas y convertir en comentario las instrucciones condicionales.

Copie este código en el proyecto:

```
key_var_name = 'TRANSLATOR_TEXT_SUBSCRIPTION_KEY'
if not key_var_name in os.environ:
    raise Exception('Please set/export the environment variable: {}'.format(key_var_name))
subscription_key = os.environ[key_var_name]

endpoint_var_name = 'TRANSLATOR_TEXT_ENDPOINT'
if not endpoint_var_name in os.environ:
    raise Exception('Please set/export the environment variable: {}'.format(endpoint_var_name))
endpoint = os.environ[endpoint_var_name]
```

El punto de conexión global de Translator Text se establece como `endpoint`. `path` establece la ruta de `breaksentence` e identifica que deseamos usar la versión 3 de la API.

Los elementos `params` de este ejemplo se utilizan para establecer el idioma del texto proporcionado. Los elementos `params` no son necesarios para la ruta `breaksentence`. Si no se incluye en la solicitud, la API intentará detectar el idioma del texto proporcionado y proporcionará esta información junto con una puntuación de confianza en la respuesta.

NOTE

Para más información sobre los puntos de conexión, las rutas y los parámetros de la solicitud, consulte [Translator Text API 3.0: Idiomas](#).

```
path = '/breaksentence?api-version=3.0'
params = '&language=en'
constructed_url = endpoint + path + params
```

Incorporación de encabezados

La manera más fácil de autenticar una solicitud es pasar la clave de suscripción como un encabezado `Ocp-Apim-Subscription-Key`, que es el que se usa en este ejemplo. O bien, puede intercambiar la clave de suscripción para un token de acceso y pasar este token como un encabezado `Authorization` para validar la solicitud. Para más información, consulte [Autenticación](#).

Copie este fragmento de código en el proyecto:

```
headers = {
    'Ocp-Apim-Subscription-Key': subscription_key,
    'Content-type': 'application/json',
    'X-ClientTraceId': str(uuid.uuid4())
}
```

Si usa una suscripción a varios servicios de Cognitive Services, también debe incluir `Ocp-Apim-Subscription-Region` en los parámetros de la solicitud. [Más información sobre la autenticación con la suscripción a varios servicios](#).

Creación de una solicitud para determinar la longitud de la frase

Defina la frase (o frases) de las que desea determinar su longitud:

```
# You can pass more than one object in body.
body = [{
    'text': 'How are you? I am fine. What did you do today?'
}]
```

A continuación, vamos a crear una solicitud POST mediante el módulo `requests`, que usa tres argumentos: la dirección URL concatenada, los encabezados de solicitud y el cuerpo de solicitud:

```
request = requests.post(constructed_url, headers=headers, json=body)
response = request.json()
```

Impresión de la respuesta

El último paso es imprimir los resultados. Este fragmento de código adorna los resultados mediante una clasificación de las claves, el establecimiento de una sangría y la declaración de separadores de elementos y de claves.

```
print(json.dumps(response, sort_keys=True, indent=4,
                  ensure_ascii=False, separators=(',', ' ')))
```

Colocación de todo junto

Eso es todo, ha creado un sencillo programa que llama a Translator Text API y devuelve una respuesta JSON. Ahora es el momento de ejecutar el programa:

```
python sentence-length.py
```

Si desea comparar su código con el nuestro, el ejemplo completo está disponible en [GitHub](#).

Respuesta de muestra

```
[
  {
    "sentLen": [
      13,
      11,
      22
    ]
  }
]
```

Limpieza de recursos

Si ha codificado de forma rígida la clave de suscripción en el programa, asegúrese de quitarla cuando haya terminado con esta guía de inicio rápido.

Pasos siguientes

Eche un vistazo a la referencia de API para comprender todo lo que puede hacer con Translator Text API.

[Referencia de API](#)

Requisitos previos

Esta guía de inicio rápido requiere:

- [Node 8.12.x o posterior](#)
- Una suscripción a Azure: [cree una cuenta gratuita](#).

Instalación

Creación de un recurso de Translator Text

Los servicios de Azure Cognitive Services se representan por medio de recursos de Azure a los que se suscribe. Cree un recurso para Translator Text mediante [Azure Portal](#) o la [CLI de Azure](#) en la máquina local. También puede:

- Obtener una [clave de prueba](#) válida durante siete días de forma gratuita. Después de registrarse, estará disponible en el sitio web de Azure.
- Vea un recurso existente en [Azure Portal](#).

Después de obtener una clave del recurso o la suscripción de evaluación, cree dos [variables de entorno](#):

- `TRANSLATOR_TEXT_SUBSCRIPTION_KEY`: la clave de suscripción del recurso de Translator Text.
- `TRANSLATOR_TEXT_ENDPOINT`: el punto de conexión global para Translator Text. Mediante `https://api.cognitive.microsofttranslator.com/`.

Creación de un proyecto e importación de los módulos necesarios

Cree un proyecto con su IDE o editor favorito. A continuación, copie este fragmento de código en un archivo llamado `sentence-length.js`.

```
const request = require('request');
const uuidv4 = require('uuid/v4');
```

NOTE

Si no ha usado estos módulos deberá instalarlos antes de ejecutar el programa. Para instalar estos paquetes, ejecute:

```
npm install request uuidv4
```

Estos módulos son necesarios para construir la solicitud HTTP y crear un identificador único para el encabezado

```
'X-ClientTraceId'.
```

Establecimiento de la clave de suscripción y el punto de conexión

En este ejemplo se intenta leer la clave de suscripción y el punto de conexión de Translator Text desde estas variables de entorno: `TRANSLATOR_TEXT_SUBSCRIPTION_KEY` y `TRANSLATOR_TEXT_ENDPOINT`. Si no está familiarizado con las variables de entorno, puede establecer `subscriptionKey` y `endpoint` como cadenas y convertir en comentario las instrucciones condicionales.

Copie este código en el proyecto:

```
var key_var = 'TRANSLATOR_TEXT_SUBSCRIPTION_KEY';
if (!process.env[key_var]) {
    throw new Error('Please set/export the following environment variable: ' + key_var);
}
var subscriptionKey = process.env[key_var];
var endpoint_var = 'TRANSLATOR_TEXT_ENDPOINT';
if (!process.env[endpoint_var]) {
    throw new Error('Please set/export the following environment variable: ' + endpoint_var);
}
var endpoint = process.env[endpoint_var];
```

Configuración de la solicitud

El método `request()`, disponible mediante el módulo de solicitud, nos permite pasar el método HTTP, la dirección URL, los parámetros de la solicitud, los encabezados y el cuerpo de JSON como un objeto `options`. En este fragmento de código, vamos a configurar la solicitud:

NOTE

Para más información sobre los puntos de conexión, las rutas y los parámetros de la solicitud, consulte [Translator Text API 3.0: BreakSentence](#).

```
let options = {
  method: 'POST',
  baseUrl: endpoint,
  url: 'breaksentence',
  qs: {
    'api-version': '3.0',
  },
  headers: {
    'Ocp-Apim-Subscription-Key': subscriptionKey,
    'Content-type': 'application/json',
    'X-ClientTraceId': uuidv4().toString()
  },
  body: [{
    'text': 'How are you? I am fine. What did you do today?'
  }],
  json: true,
};
```

La manera más fácil de autenticar una solicitud es pasar la clave de suscripción como un encabezado `Ocp-Apim-Subscription-Key`, que es el que se usa en este ejemplo. O bien, puede intercambiar la clave de suscripción para un token de acceso y pasar este token como un encabezado `Authorization` para validar la solicitud.

Si usa una suscripción a varios servicios de Cognitive Services, también debe incluir `Ocp-Apim-Subscription-Region` en los encabezados de la solicitud.

Para más información, consulte [Autenticación](#).

Realización de solicitud e impresión de la respuesta

A continuación, vamos a crear una solicitud mediante el método `request()`. Toma el objeto `options` que se creó en la sección anterior como el primer argumento y, a continuación, imprime la respuesta JSON embellecida.

```
request(options, function(err, res, body){
    console.log(JSON.stringify(body, null, 4));
});
```

NOTE

En este ejemplo, vamos a definir la solicitud HTTP en el objeto `options`. Sin embargo, el módulo de solicitud también admite métodos de conveniencia como `.post` y `.get`. Para más información, consulte [métodos de conveniencia](#).

Colocación de todo junto

Eso es todo, ha creado un sencillo programa que llama a Translator Text API y devuelve una respuesta JSON. Ahora es el momento de ejecutar el programa:

```
node sentence-length.js
```

Si desea comparar su código con el nuestro, el ejemplo completo está disponible en [GitHub](#).

Respuesta de muestra

```
[
  {
    "sentLen": [
      13,
      11,
      22
    ]
  }
]
```

Limpieza de recursos

Si ha codificado de forma rígida la clave de suscripción en el programa, asegúrese de quitarla cuando haya terminado con esta guía de inicio rápido.

Pasos siguientes

Eche un vistazo a la referencia de API para comprender todo lo que puede hacer con Translator Text API.

[Referencia de API](#)

Requisitos previos

Esta guía de inicio rápido requiere:

- [Go](#)
- Una suscripción a Azure: [cree una cuenta gratuita](#).

Instalación

Creación de un recurso de Translator Text

Los servicios de Azure Cognitive Services se representan por medio de recursos de Azure a los que se suscribe.

Cree un recurso para Translator Text mediante [Azure Portal](#) o la [CLI de Azure](#) en la máquina local. También puede:

- Obtener una [clave de prueba](#) válida durante siete días de forma gratuita. Después de registrarse, estará disponible en el sitio web de Azure.
- Vea un recurso existente en [Azure Portal](#).

Después de obtener una clave del recurso o la suscripción de evaluación, cree dos [variables de entorno](#):

- `TRANSLATOR_TEXT_SUBSCRIPTION_KEY`: la clave de suscripción del recurso de Translator Text.
- `TRANSLATOR_TEXT_ENDPOINT`: el punto de conexión global para Translator Text. Mediante `https://api.cognitive.microsofttranslator.com/`.

Creación de un proyecto e importación de los módulos necesarios

Cree un proyecto de Go con su IDE o editor favorito. A continuación, copie este fragmento de código en un archivo llamado `sentence-length.go`.

```
package main

import (
    "bytes"
    "encoding/json"
    "fmt"
    "log"
    "net/http"
    "net/url"
    "os"
)
```

Creación de la función main

En este ejemplo se intenta leer la clave de suscripción y el punto de conexión de Translator Text desde estas variables de entorno: `TRANSLATOR_TEXT_SUBSCRIPTION_KEY` y `TRANSLATOR_TEXT_ENDPOINT`. Si no está familiarizado con las variables de entorno, puede establecer `subscriptionKey` y `endpoint` como cadenas y convertir en comentario las instrucciones condicionales.

Copie este código en el proyecto:

```
func main() {
    /*
     * Read your subscription key from an env variable.
     * Please note: You can replace this code block with
     * var subscriptionKey = "YOUR_SUBSCRIPTION_KEY" if you don't
     * want to use env variables. If so, be sure to delete the "os" import.
     */
    if "" == os.Getenv("TRANSLATOR_TEXT_SUBSCRIPTION_KEY") {
        log.Fatal("Please set/export the environment variable TRANSLATOR_TEXT_SUBSCRIPTION_KEY.")
    }
    subscriptionKey := os.Getenv("TRANSLATOR_TEXT_SUBSCRIPTION_KEY")
    if "" == os.Getenv("TRANSLATOR_TEXT_ENDPOINT") {
        log.Fatal("Please set/export the environment variable TRANSLATOR_TEXT_ENDPOINT.")
    }
    endpoint := os.Getenv("TRANSLATOR_TEXT_ENDPOINT")
    uri := endpoint + "/breaksentence?api-version=3.0"
    /*
     * This calls our breakSentence function, which we'll
     * create in the next section. It takes a single argument,
     * the subscription key.
     */
    breakSentence(subscriptionKey, uri)
}
```

Creación de una función para determinar la longitud de la frase

Vamos a crear una función para determinar la longitud de la oración. Esta función tendrá un solo argumento, su clave de suscripción de Translator Text.

```
func breakSentence(subscriptionKey string, uri string)
/*
 * In the next few sections, we'll add code to this
 * function to make a request and handle the response.
 */
}
```

A continuación, vamos a construir la dirección URL. La dirección URL se crea mediante los métodos `Parse()` y `Query()`. Observará que se han agregado los parámetros con el método `Add()`.

Copie este código en la función `breakSentence`.

```
// Build the request URL. See: https://golang.org/pkg/net/url/#example_URL_Parse
u, _ := url.Parse(uri)
q := u.Query()
q.Add("languages", "en")
u.RawQuery = q.Encode()
```

NOTE

Para más información sobre los puntos de conexión, las rutas y los parámetros de la solicitud, consulte [Translator Text API 3.0: BreakSentence](#).

Creación de una estructura para el cuerpo de la solicitud

A continuación, cree una estructura anónima para el cuerpo de la solicitud y codifíquelo como JSON con `json.Marshal()`. Agregue este código a la función `breakSentence`.

```
// Create an anonymous struct for your request body and encode it to JSON
body := []struct {
    Text string
}{
    {Text: "How are you? I am fine. What did you do today?"},
}
b, _ := json.Marshal(body)
```

Compilar la solicitud

Ahora que se ha codificado el cuerpo de la solicitud como JSON, puede crear la solicitud POST y llamar a Translator Text API.

```
// Build the HTTP POST request
req, err := http.NewRequest("POST", u.String(), bytes.NewBuffer(b))
if err != nil {
    log.Fatal(err)
}
// Add required headers to the request
req.Header.Add("Ocp-Apim-Subscription-Key", subscriptionKey)
req.Header.Add("Content-Type", "application/json")

// Call the Translator Text API
res, err := http.DefaultClient.Do(req)
if err != nil {
    log.Fatal(err)
}
```

Si usa una suscripción a varios servicios de Cognitive Services, también debe incluir

`Ocp-Apim-Subscription-Region` en los parámetros de la solicitud. [Más información sobre la autenticación con la suscripción a varios servicios.](#)

Control e impresión de la respuesta

Agregue este código a la función `breakSentence` para decodificar la respuesta JSON y luego dé formato al resultado e imprímalo.

```
// Decode the JSON response
var result interface{}
if err := json.NewDecoder(res.Body).Decode(&result); err != nil {
    log.Fatal(err)
}
// Format and print the response to terminal
prettyJSON, _ := json.MarshalIndent(result, "", " ")
fmt.Printf("%s\n", prettyJSON)
```

Colocación de todo junto

Eso es todo, ha creado un sencillo programa que llama a Translator Text API y devuelve una respuesta JSON. Ahora es el momento de ejecutar el programa:

```
go run sentence-length.go
```

Si desea comparar su código con el nuestro, el ejemplo completo está disponible en [GitHub](#).

Respuesta de muestra

```
[
  {
    "detectedLanguage": {
      "language": "en",
      "score": 1.0
    },
    "sentLen": [
      13,
      11,
      22
    ]
  }
]
```

Pasos siguientes

Eche un vistazo a la referencia de API para comprender todo lo que puede hacer con Translator Text API.

[Referencia de API](#)

Consulte también

- [Traducir texto](#)
- [Transliterar texto](#)
- [Identificar el idioma de entrada](#)
- [Obtener traducciones alternativas](#)
- [Obtener una lista de idiomas admitidos](#)

Tutorial: Creación de una aplicación de traducción con WPF

13/01/2020 • 29 minutes to read • [Edit Online](#)

En este tutorial, se va a crear una aplicación de [Windows Presentation Foundation \(WPF\)](#) que utiliza Azure Cognitive Services para la traducción de texto, la detección de idioma y la corrección ortográfica con una única clave de suscripción. En concreto, la aplicación llamará a las API de Translator Text y a las de [Bing Spell Check](#).

¿Qué es WPF? Es una plataforma de interfaz de usuario que crea aplicaciones de cliente de escritorio. La plataforma de desarrollo WPF admite un amplio conjunto de características de desarrollo de aplicaciones, incluidos un modelo de aplicación, recursos, controles, gráficos, diseño, enlace de datos, documentos y seguridad. Es un subconjunto de .NET Framework, por lo que si ya ha creado aplicaciones con .NET Framework mediante ASP.NET o Windows Forms, la experiencia de programación debería resultar familiar. WPF utiliza el lenguaje XAML para proporcionar un modelo declarativo para la programación de aplicaciones, que revisaremos en las próximas secciones.

En este tutorial, aprenderá a:

- Creación de un proyecto WPF en Visual Studio
- Adición de paquetes NuGet y ensamblados al proyecto
- Creación de la interfaz de usuario de la aplicación con XAML
- Uso de Translator Text API para obtener idiomas, traducir texto y detectar el idioma de origen
- Uso de Bing Spell Check API para validar la entrada y mejorar la precisión de la traducción
- Ejecución de la aplicación de WPF

Servicios Cognitive Services usados en este tutorial

En esta lista se incluyen los servicios Cognitive Services utilizados en este tutorial. Siga el vínculo para buscar la referencia de API de cada característica.

SERVICIO	CARACTERÍSTICA	DESCRIPCIÓN
Translator Text	Obtener idiomas	Recupera una lista completa de los idiomas admitidos para la traducción de texto.
Translator Text	Traducir	Traduce texto en más de 60 idiomas.
Translator Text	Detectar	Detecta el idioma del texto de entrada. Incluye la puntuación de confianza para la detección.
Bing Spell Check	Corrector ortográfico	Corrige los errores de ortografía para mejorar la precisión de la traducción.

Requisitos previos

Antes de continuar, necesitará lo siguiente:

- Una suscripción a Azure Cognitive Services. [Obtenga una clave de Cognitive Services](#).
- Un equipo Windows
- [Visual Studio 2019](#): Community o Enterprise

NOTE

Se recomienda crear la suscripción en la región Oeste de EE. UU. para este tutorial. De lo contrario, tendrá que cambiar los puntos de conexión y las regiones en el código mientras trabaja en este ejercicio.

Creación de una aplicación de WPF en Visual Studio

Lo primero que debemos hacer es configurar nuestro proyecto en Visual Studio.

1. Abra Visual Studio. Seleccione **Crear un proyecto**.
2. En **Crear un proyecto**, busque y seleccione **Aplicación de WPF (.NET Framework)** . Puede seleccionar C# en **Lenguaje** para restringir las opciones.
3. Seleccione **Siguiente** y, después, asigne un nombre al proyecto `MSTranslatorTextDemo` .
4. Establezca la versión de la plataforma en **.NET Framework 4.7.2 o posterior** y, después, haga clic en **Crear**.

Configure your new project

WPF App (.NET Framework) C# Windows Desktop

Project name
MSTranslatorTextDemo

Location
C:\Users\translatorprojects\Source\Repos

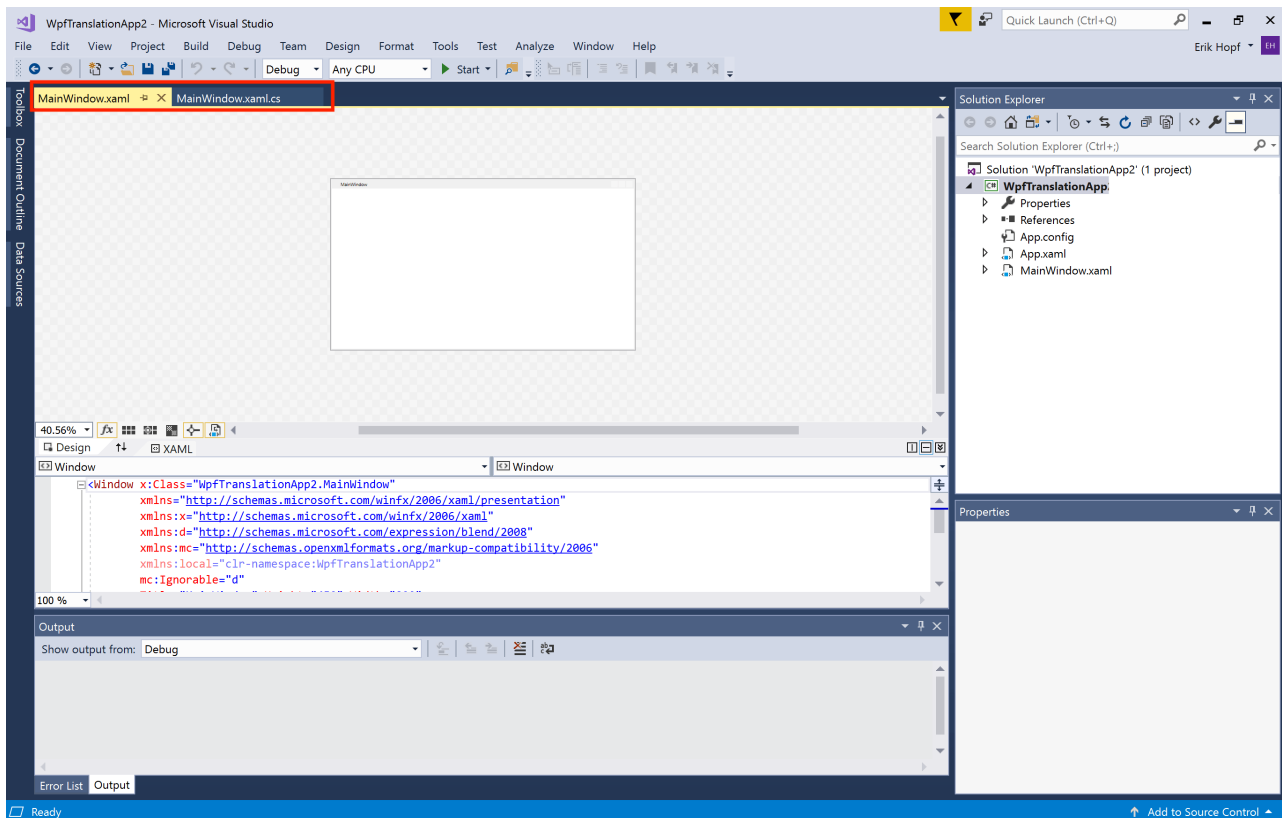
Solution name ⓘ
MSTranslatorTextDemo

☐ Place solution and project in the same directory

Framework
.NET Framework 4.7.2

Back Create

Se ha creado el proyecto. Observará que hay dos pestañas abiertas: `MainWindow.xaml` y `MainWindow.xaml.cs` . A lo largo de este tutorial, iremos agregando código a estos dos archivos. Modificaremos `MainWindow.xaml` para la interfaz de usuario de la aplicación. Modificaremos `MainWindow.xaml.cs` para las llamadas a Translator Text y Bing Spell Check.



En la siguiente sección vamos a agregar ensamblados y un paquete NuGet a nuestro proyecto para una funcionalidad adicional, como el análisis de JSON.

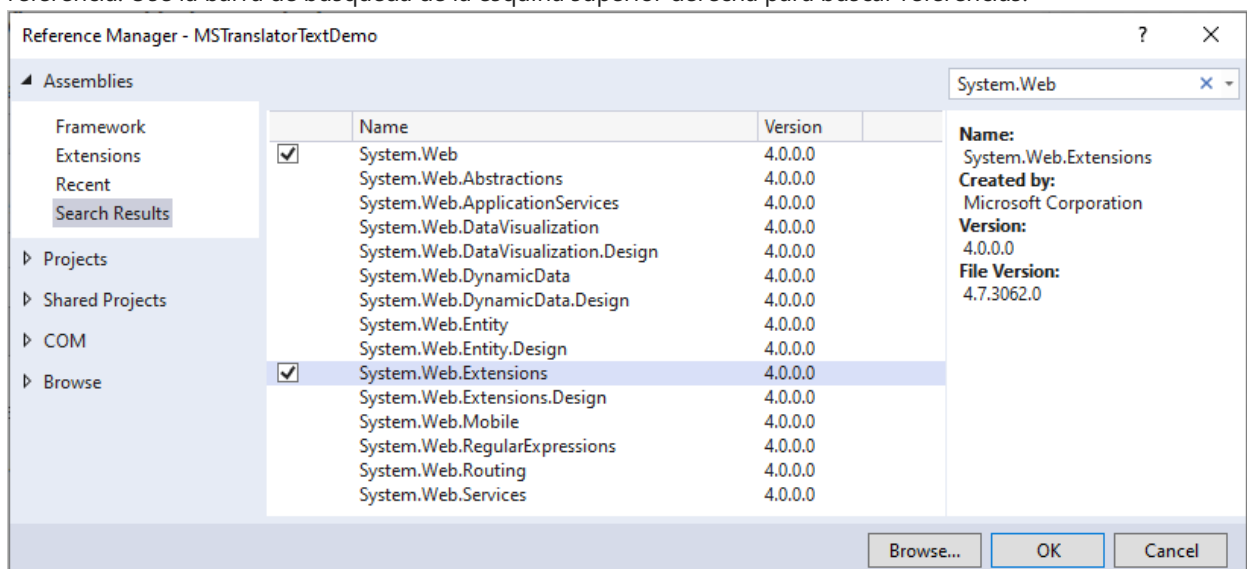
Adición de referencias y paquetes NuGet al proyecto

Nuestro proyecto requiere un grupo de ensamblados de .NET Framework y Newtonsoft.Json, que se van a instalar mediante el administrador de paquetes NuGet.

Adición de ensamblados de .NET Framework

Agreguemos ensamblados a nuestro proyecto para serializar y deserializar objetos, y para administrar solicitudes y respuestas HTTP.

1. Busque el proyecto en el Explorador de soluciones de Visual Studio. Haga clic con el botón derecho en el proyecto y seleccione **Agregar > Referencia**, que abrirá **Administrador de referencias**.
2. La pestaña **Ensamblados** muestra todos los ensamblados de .NET Framework que están disponibles como referencia. Use la barra de búsqueda de la esquina superior derecha para buscar referencias.



3. Seleccione las siguientes referencias para el proyecto:

- [System.Runtime.Serialization](#)
- [System.Web](#)
- System.Web.Extensions
- [System.Windows](#)

4. Después de agregar estas referencias al proyecto, haga clic en **Aceptar** para cerrar **Administrador de referencias**.

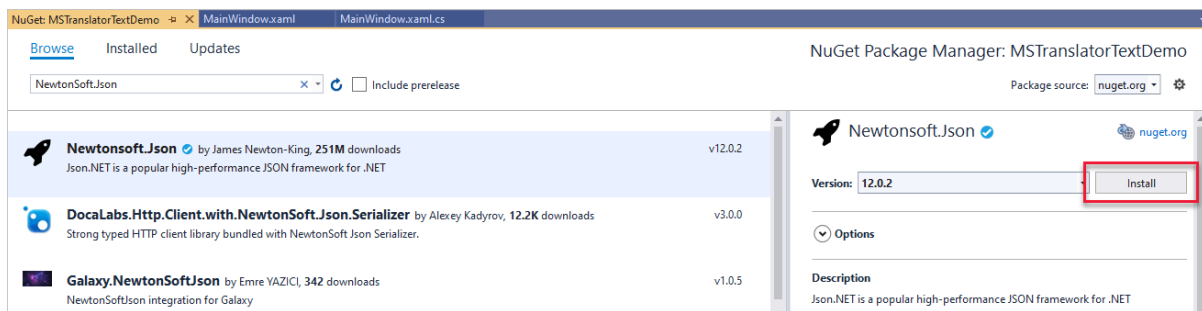
NOTE

Para más información sobre las referencias de ensamblado, consulte [Procedimiento: Agregar o quitar referencias con el Administrador de referencias](#).

Instalación de Newtonsoft.Json

Nuestra aplicación usará Newtonsoft.Json para deserializar los objetos JSON. Siga estas instrucciones para instalar el paquete.

1. Busque el proyecto en el Explorador de soluciones de Visual Studio y haga clic con el botón derecho en el proyecto. Seleccione **Administrar paquetes NuGet**.
2. Busque y seleccione la pestaña **Examinar**.
3. Escriba [NewtonSoft.Json](#) en la barra de búsqueda.

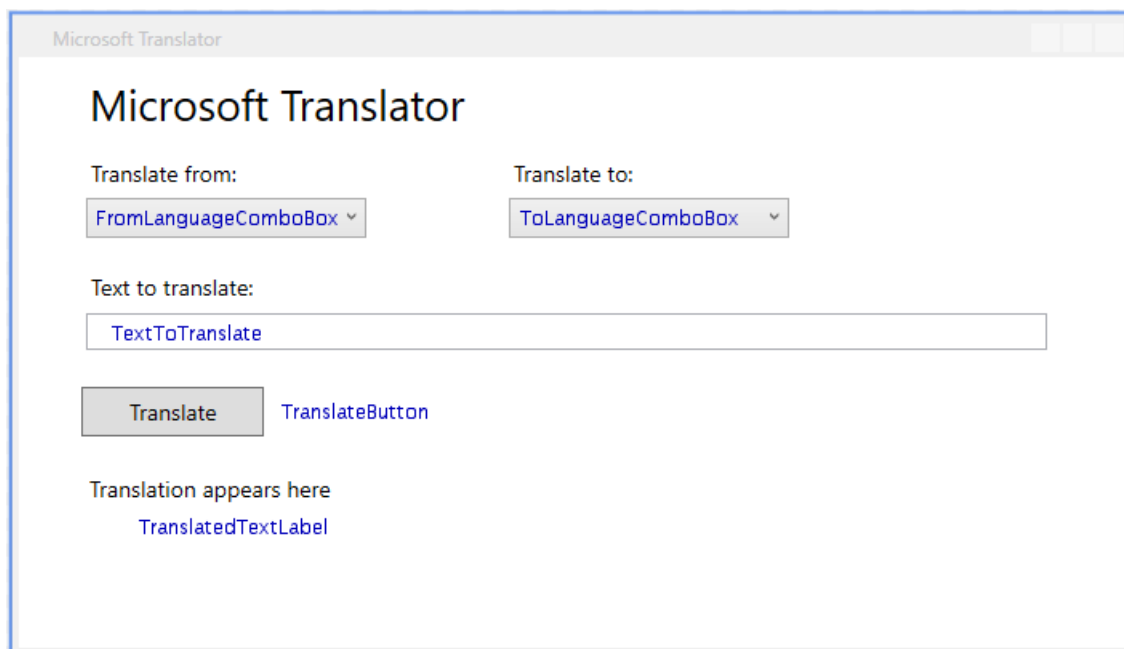


4. Seleccione el paquete y haga clic en **Instalar**.
5. Cuando la instalación se haya completado, cierre la pestaña.

Creación de un formulario WPF mediante XAML

Para usar la aplicación, va a necesitar una interfaz de usuario. Con XAML, vamos a crear un formulario que permite a los usuarios seleccionar los idiomas de entrada y de traducción, introducir el texto que se va a traducir y mostrar el resultado de la traducción.

Veamos lo que estamos creando.



La interfaz de usuario incluye los siguientes componentes:

NOMBRE	TIPO	DESCRIPCIÓN
FromLanguageComboBox	Cuadro combinado	Muestra una lista de los idiomas admitidos por Microsoft Translator para la traducción de texto. El usuario selecciona el idioma del que está traduciendo.
ToLanguageComboBox	Cuadro combinado	Muestra la misma lista de idiomas que FromLanguageComboBox, pero se utiliza para seleccionar el idioma al que se traduce.
TextToTranslate	TextBox	Permite al usuario escribir texto que se va a traducir.
TranslateButton	Botón	Use este botón para traducir texto.
TranslatedTextLabel	Etiqueta	Muestra la traducción.
DetectedLanguageLabel	Etiqueta	Muestra el idioma detectado del texto que se va a traducir (TextToTranslate).

NOTE

Estamos creando este formulario utilizando el código fuente de XAML; sin embargo, puede crear el formulario con el editor en Visual Studio.

Vamos a agregar el código al proyecto.

1. En Visual Studio, seleccione la pestaña para MainWindow.xaml.
2. Copie este código en el proyecto y, después, seleccione **Archivo > Guardar MainWindow.xaml** para guardar los cambios.

```

<Window x:Class="MSTranslatorTextDemo.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
        xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
        xmlns:local="clr-namespace:MSTranslatorTextDemo"
        mc:Ignorable="d"
        Title="Microsoft Translator" Height="400" Width="700" BorderThickness="0">
    <Grid>
        <Label x:Name="label" Content="Microsoft Translator" HorizontalAlignment="Left"
Margin="39,6,0,0" VerticalAlignment="Top" Height="49" FontSize="26.667"/>
        <TextBox x:Name="TextToTranslate" HorizontalAlignment="Left" Height="23" Margin="42,160,0,0"
TextWrapping="Wrap" VerticalAlignment="Top" Width="600" FontSize="14" TabIndex="3"/>
        <Label x:Name="EnterTextLabel" Content="Text to translate:" HorizontalAlignment="Left"
Margin="40,129,0,0" VerticalAlignment="Top" FontSize="14"/>
        <Label x:Name="toLabel" Content="Translate to:" HorizontalAlignment="Left" Margin="304,58,0,0"
VerticalAlignment="Top" FontSize="14"/>

        <Button x:Name="TranslateButton" Content="Translate" HorizontalAlignment="Left"
Margin="39,206,0,0" VerticalAlignment="Top" Width="114" Height="31" Click="TranslateButton_Click"
FontSize="14" TabIndex="4" IsDefault="True"/>
        <ComboBox x:Name="ToLanguageComboBox"
            HorizontalAlignment="Left"
            Margin="306,88,0,0"
            VerticalAlignment="Top"
            Width="175" FontSize="14" TabIndex="2">

        </ComboBox>
        <Label x:Name="fromLabel" Content="Translate from:" HorizontalAlignment="Left"
Margin="40,58,0,0" VerticalAlignment="Top" FontSize="14"/>
        <ComboBox x:Name="FromLanguageComboBox"
            HorizontalAlignment="Left"
            Margin="42,88,0,0"
            VerticalAlignment="Top"
            Width="175" FontSize="14" TabIndex="1"/>
        <Label x:Name="TranslatedTextLabel" Content="Translation is displayed here."
HorizontalAlignment="Left" Margin="39,255,0,0" VerticalAlignment="Top" Width="620" FontSize="14"
Height="85" BorderThickness="0"/>
        <Label x:Name="DetectedLanguageLabel" Content="Autodetected language is displayed here."
HorizontalAlignment="Left" Margin="39,288,0,0" VerticalAlignment="Top" Width="620" FontSize="14"
Height="84" BorderThickness="0"/>
    </Grid>
</Window>

```

Ahora debería ver una vista previa de la interfaz de usuario de la aplicación en Visual Studio. El archivo debe tener un aspecto similar a la imagen anterior.

Eso es todo, el formulario está listo. Ahora vamos a escribir código para usar Translator Text y Bing Spell Check.

NOTE

Puede adaptar este formulario o crear el suyo propio.

Creación de la aplicación

`MainWindow.xaml.cs` contiene el código que controla la aplicación. En las próximas secciones, se va a agregar código para rellenar los menús desplegables, y para llamar a un grupo de API expuestas por Translator Text y Bing Spell Check.

- Cuando el programa se inicia y se crea una instancia de `MainWindow`, se llama al método `Languages` de Translator Text API para recuperar y poblar las listas desplegables de selección de idioma. Esta tarea se realiza una vez al principio de cada sesión.

- Cuando se hace clic en el botón **Traducir**, se recupera el texto y la selección del idioma del usuario, se realiza una corrección ortográfica en la entrada y se muestran al usuario la traducción y el idioma detectado.
 - Se llama al método `Translate` de Translator Text API para traducir el texto desde `TextToTranslate`. Esta llamada también incluye los idiomas `to` y `from` seleccionados mediante los menús desplegables.
 - Se llama al método `Detect` de Translator Text API para determinar el idioma del texto de `TextToTranslate`.
 - Bing Spell Check se utiliza para validar `TextToTranslate` y ajustar los errores de ortografía.

Todo nuestro proyecto se encapsula en la clase `MainWindow : Window`. Comencemos por agregar código para establecer la clave de suscripción, declarar los puntos de conexión para Translator Text y Bing Spell Check, e inicializar la aplicación.

1. En Visual Studio, seleccione la pestaña para `MainWindow.xaml.cs`.
2. Reemplace las instrucciones `using` previamente rellenas por lo siguiente.

```
using System;
using System.Windows;
using System.Net;
using System.Net.Http;
using System.IO;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using Newtonsoft.Json;
```

3. Busque la clase `MainWindow : Window` y reemplácela por este código:

```

{
    // This sample uses the Cognitive Services subscription key for all services. To learn more about
    // authentication options, see: https://docs.microsoft.com/azure/cognitive-services/authentication.
    const string COGNITIVE_SERVICES_KEY = "YOUR_COG_SERVICES_KEY";
    // Endpoints for Translator Text and Bing Spell Check
    public static readonly string TEXT_TRANSLATION_API_ENDPOINT =
"https://api.cognitive.microsofttranslator.com/{0}?api-version=3.0";
    const string BING_SPELL_CHECK_API_ENDPOINT =
"https://westus.api.cognitive.microsoft.com/bing/v7.0/spellcheck/";
    // An array of language codes
    private string[] languageCodes;

    // Dictionary to map language codes from friendly name (sorted case-insensitively on language name)
    private SortedDictionary<string, string> languageCodesAndTitles =
        new SortedDictionary<string, string>(Comparer<string>.Create((a, b) => string.Compare(a, b,
true)));

    // Global exception handler to display error message and exit
    private static void HandleExceptions(object sender, UnhandledExceptionEventArgs args)
    {
        Exception e = (Exception)args.ExceptionObject;
        MessageBox.Show("Caught " + e.Message, "Error", MessageBoxButton.OK, MessageBoxImage.Error);
        System.Windows.Application.Current.Shutdown();
    }
    // MainWindow constructor
    public MainWindow()
    {
        // Display a message if unexpected error is encountered
        AppDomain.CurrentDomain.UnhandledException += new
        UnhandledExceptionEventHandler(HandleExceptions);

        if (COGNITIVE_SERVICES_KEY.Length != 32)
        {
            MessageBox.Show("One or more invalid API subscription keys.\n\n" +
                "Put your keys in the *_API_SUBSCRIPTION_KEY variables in MainWindow.xaml.cs.",
                "Invalid Subscription Key(s)", MessageBoxButton.OK, MessageBoxImage.Error);
            System.Windows.Application.Current.Shutdown();
        }
        else
        {
            // Start GUI
            InitializeComponent();
            // Get languages for drop-downs
            GetLanguagesForTranslate();
            // Populate drop-downs with values from GetLanguagesForTranslate
            PopulateLanguageMenus();
        }
    }
}
// NOTE:
// In the following sections, we'll add code below this.
}

```

4. Agregue la clave de suscripción de Cognitive Services y guárdela.

En este bloque de código, hemos declarado dos variables de miembro que contienen información sobre los idiomas disponibles para traducción:

VARIABLE	TIPO	DESCRIPCIÓN
<code>languageCodes</code>	Matriz de cadenas	Almacena en caché los códigos de idioma. El servicio de Translator utiliza códigos cortos, como <code>en</code> para inglés, para identificar los idiomas.

VARIABLE	TIPO	DESCRIPCIÓN
<code>languageCodesAndTitles</code>	Diccionario ordenado	Asigna los nombres "descriptivos" en la interfaz de usuario a los códigos cortos que se utilizan en la API. Se mantienen ordenados por orden alfabético sin tener en cuenta las mayúsculas y minúsculas.

A continuación, en el constructor `MainWindow`, hemos agregado el control de errores con `HandleExceptions`. Este control de errores asegura que se proporcione una alerta si no se controla una excepción. A continuación, se ejecuta una comprobación para confirmar que la clave de suscripción proporcionada tiene 32 caracteres. Se produce un error si la clave tiene menos o más de 32 caracteres.

Si hay claves que tengan al menos la longitud adecuada, la llamada `InitializeComponent()` da inicio a la interfaz de usuario al ubicar, cargar y crear instancias de la descripción de XAML de la ventana de la aplicación principal.

Por último, hemos agregado código a métodos de llamada para recuperar los idiomas para traducir y para rellenar los menús desplegables de la interfaz de usuario de la aplicación. Veremos el código subyacente de estas llamadas en breve.

Obtener idiomas admitidos

Translator Text API admite actualmente con más de 60 idiomas. Dado que con el tiempo se agregará nueva compatibilidad de idiomas, recomendamos llamar al recurso de idiomas que expone Translator Text en lugar de codificar la lista de idiomas de la aplicación.

En esta sección, se va a crear una solicitud `GET` al recurso de idiomas, especificando que queremos una lista de idiomas disponibles para traducir.

NOTE

El recurso de idiomas permite filtrar la compatibilidad de idiomas con los siguientes parámetros de consulta: transliteración, diccionario y traducción. Para más información, consulte la [referencia de API](#).

Antes de continuar, echemos un vistazo a una salida de ejemplo para una llamada al recurso de idiomas:

```
{
  "translation": {
    "af": {
      "name": "Afrikaans",
      "nativeName": "Afrikaans",
      "dir": "ltr"
    },
    "ar": {
      "name": "Arabic",
      "nativeName": "العربية",
      "dir": "rtl"
    }
  }
  // Additional languages are provided in the full JSON output.
}
```

En esta salida, podemos extraer el código de idioma y el `name` de un idioma específico. La aplicación usa `NewtonSoft.Json` para deserializar el objeto JSON (`JsonConvert.DeserializeObject`).

Retomando el punto donde se dejó en la última sección, agregamos un método para obtener idiomas compatibles con la aplicación.

1. En Visual Studio, abra la pestaña para `MainWindow.xaml.cs`.
2. Agregue este código al proyecto:

```
// ***** GET TRANSLATABLE LANGUAGE CODES
private void GetLanguagesForTranslate()
{
    // Send request to get supported language codes
    string uri = String.Format(TEXT_TRANSLATION_API_ENDPOINT, "languages") + "&scope=translation";
    WebRequest WebRequest = WebRequest.Create(uri);
    WebRequest.Headers.Add("Accept-Language", "en");
    WebResponse response = null;
    // Read and parse the JSON response
    response = WebRequest.GetResponse();
    using (var reader = new StreamReader(response.GetResponseStream(), UnicodeEncoding.UTF8))
    {
        var result = JsonConvert.DeserializeObject<Dictionary<string, Dictionary<string,
Dictionary<string, string>>>>(reader.ReadToEnd());
        var languages = result["translation"];

        languageCodes = languages.Keys.ToArray();
        foreach (var kv in languages)
        {
            languageCodesAndTitles.Add(kv.Value["name"], kv.Key);
        }
    }
}
// NOTE:
// In the following sections, we'll add code below this.
```

El método `GetLanguagesForTranslate()` crea una solicitud HTTP GET y utiliza el parámetro de cadena de consulta `scope=translation` para limitar el ámbito de la solicitud a los idiomas admitidos para traducción. Se agrega el encabezado `Accept-Language` con el valor `en` para que se devuelvan los idiomas admitidos en inglés.

La respuesta JSON se analiza y se convierte en un diccionario. Después, se agregan los códigos de idioma a la variable de miembro `languageCodes`. Se recorren los pares de clave/valor que contienen los códigos de idioma y los nombres de idioma descriptivos, y se agregan a la variable miembro `languageCodesAndTitles`. En los menús desplegables del formulario se muestran los nombres descriptivos, pero se necesitan los códigos para solicitar la traducción.

Rellenado de los menús desplegables de idioma

La interfaz de usuario se define mediante XAML, por lo que no es necesario hacer mucho para configurarla, además de llamar a `InitializeComponent()`. Lo único que necesita hacer es agregar los nombres de idiomas descriptivos en los menús desplegables **Translate from** (Traducir de) y **Translate to** (Traducir a). El método `PopulateLanguageMenus()` agrega los nombres.

1. En Visual Studio, abra la pestaña para `MainWindow.xaml.cs`.
2. Agregue este código al proyecto debajo del método `GetLanguagesForTranslate()`:

```
private void PopulateLanguageMenus()
{
    // Add option to automatically detect the source language
    FromLanguageComboBox.Items.Add("Detect");

    int count = languageCodesAndTitles.Count;
    foreach (string menuItem in languageCodesAndTitles.Keys)
    {
        FromLanguageComboBox.Items.Add(menuItem);
        ToLanguageComboBox.Items.Add(menuItem);
    }

    // Set default languages
    FromLanguageComboBox.SelectedItem = "Detect";
    ToLanguageComboBox.SelectedItem = "English";
}
// NOTE:
// In the following sections, we'll add code below this.
```

Este método itera el diccionario `languageCodesAndTitles` y agrega cada clave a ambos menús. Una vez rellenos los menús, los idiomas de origen y destino predeterminados se establecen en **Detectar** e **Inglés**, respectivamente.

TIP

Sin una selección predeterminada para los menús, el usuario puede hacer clic en **Translate** (Traducir) sin elegir el idioma de origen ni el de destino. Los valores predeterminados eliminan la necesidad de tratar este problema.

Ahora que `MainWindow` se ha inicializado y se ha creado la interfaz de usuario, este código no se ejecutará hasta que se haga clic en el botón **Translate** (Traducir).

Detección del idioma del texto de origen

Ahora vamos a crear un método para detectar el idioma del texto de origen (texto introducido en nuestra área de texto) mediante Translator Text API. El valor devuelto por esta solicitud se usará más adelante en la solicitud de traducción.

1. En Visual Studio, abra la pestaña para `MainWindow.xaml.cs`.
2. Agregue este código al proyecto debajo del método `PopulateLanguageMenus()`:

```
// ***** DETECT LANGUAGE OF TEXT TO BE TRANSLATED
private string DetectLanguage(string text)
{
    string detectUri = string.Format(TEXT_TRANSLATION_API_ENDPOINT , "detect");

    // Create request to Detect languages with Translator Text
    HttpRequest detectLanguageWebRequest = (HttpRequest)WebRequest.Create(detectUri);
    detectLanguageWebRequest.Headers.Add("Ocp-Apim-Subscription-Key", COGNITIVE_SERVICES_KEY);
    detectLanguageWebRequest.Headers.Add("Ocp-Apim-Subscription-Region", "westus");
    detectLanguageWebRequest.ContentType = "application/json; charset=utf-8";
    detectLanguageWebRequest.Method = "POST";

    // Send request
    var serializer = new System.Web.Script.Serialization.JavaScriptSerializer();
    string jsonText = serializer.Serialize(text);

    string body = "[{ \"Text\": \" + jsonText + \" }]";
    byte[] data = Encoding.UTF8.GetBytes(body);

    detectLanguageWebRequest.ContentLength = data.Length;

    using (var requestStream = detectLanguageWebRequest.GetRequestStream())
        requestStream.Write(data, 0, data.Length);

    HttpResponseMessage response = (HttpResponseMessage)detectLanguageWebRequest.GetResponse();

    // Read and parse JSON response
    var responseStream = response.GetResponseStream();
    var jsonString = new StreamReader(responseStream, Encoding.GetEncoding("utf-8")).ReadToEnd();
    dynamic jsonResponse = serializer.DeserializeObject(jsonString);

    // Fish out the detected language code
    var languageInfo = jsonResponse[0];
    if (languageInfo["score"] > (decimal)0.5)
    {
        DetectedLanguageLabel.Content = languageInfo["language"];
        return languageInfo["language"];
    }
    else
        return "Unable to confidently detect input language.";
}
// NOTE:
// In the following sections, we'll add code below this.
```

Este método crea una solicitud HTTP `POST` en el recurso de detección. Toma un único argumento, `text`, que se pasa como el cuerpo de la solicitud. Más tarde, cuando creamos nuestra solicitud de traducción, el texto introducido en la interfaz de usuario pasará a este método para la detección del idioma.

Además, este método evalúa la puntuación de confianza de la respuesta. Si la puntuación es superior a `0.5`, el idioma detectado se muestra en la interfaz de usuario.

Corrección ortográfica del texto de origen

Ahora vamos a crear un método para corregir la ortografía del texto de origen con Bing Spell Check API. El corrector ortográfico garantiza que obtendremos traducciones precisas de Translator Text API. Todas las correcciones del texto de origen se pasan a la solicitud de traducción cuando se hace clic en el botón **Traducir**.

1. En Visual Studio, abra la pestaña para `MainWindow.xaml.cs`.
2. Agregue este código al proyecto debajo del método `DetectLanguage()`:

```
// ***** CORRECT SPELLING OF TEXT TO BE TRANSLATED
private string CorrectSpelling(string text)
{
    string uri = BING_SPELL_CHECK_API_ENDPOINT + "?mode=spell&mkt=en-US";

    // Create a request to Bing Spell Check API
    HttpRequest spellCheckWebRequest = (HttpRequest)WebRequest.Create(uri);
    spellCheckWebRequest.Headers.Add("Ocp-Apim-Subscription-Key", COGNITIVE_SERVICES_KEY);
    spellCheckWebRequest.Method = "POST";
    spellCheckWebRequest.ContentType = "application/x-www-form-urlencoded"; // doesn't work without this

    // Create and send the request
    string body = "text=" + System.Web.HttpUtility.UrlEncode(text);
    byte[] data = Encoding.UTF8.GetBytes(body);
    spellCheckWebRequest.ContentLength = data.Length;
    using (var requestStream = spellCheckWebRequest.GetRequestStream())
        requestStream.Write(data, 0, data.Length);
    HttpResponse response = (HttpResponse)spellCheckWebRequest.GetResponse();

    // Read and parse the JSON response; get spelling corrections
    var serializer = new System.Web.Script.Serialization.JavaScriptSerializer();
    var responseStream = response.GetResponseStream();
    var jsonString = new StreamReader(responseStream, Encoding.GetEncoding("utf-8")).ReadToEnd();
    dynamic jsonResponse = serializer.DeserializeObject(jsonString);
    var flaggedTokens = jsonResponse["flaggedTokens"];

    // Construct sorted dictionary of corrections in reverse order (right to left)
    // This ensures that changes don't impact later indexes
    var corrections = new SortedDictionary<int, string[]>(Comparer<int>.Create((a, b) => b.CompareTo(a)));
    for (int i = 0; i < flaggedTokens.Length; i++)
    {
        var correction = flaggedTokens[i];
        var suggestion = correction["suggestions"][0]; // Consider only first suggestion
        if (suggestion["score"] > (decimal)0.7) // Take it only if highly confident
            corrections[(int)correction["offset"]] = new string[] // dict key = offset
                { correction["token"], suggestion["suggestion"] }; // dict value = {error, correction}
    }

    // Apply spelling corrections, in order, from right to left
    foreach (int i in corrections.Keys)
    {
        var oldtext = corrections[i][0];
        var newtext = corrections[i][1];

        // Apply capitalization from original text to correction - all caps or initial caps
        if (text.Substring(i, oldtext.Length).All(char.IsUpper)) newtext = newtext.ToUpper();
        else if (char.IsUpper(text[i])) newtext = newtext[0].ToString().ToUpper() + newtext.Substring(1);

        text = text.Substring(0, i) + newtext + text.Substring(i + oldtext.Length);
    }
    return text;
}
// NOTE:
// In the following sections, we'll add code below this.
```

Traducción de texto al hacer clic

Lo último que tenemos que hacer es crear un método que se invoque cuando se hace clic en el botón **Traducir** de la interfaz de usuario.

1. En Visual Studio, abra la pestaña para `MainWindow.xaml.cs`.
2. Agregue este código al proyecto debajo del método `CorrectSpelling()` y guárdelo:

```
// ***** PERFORM TRANSLATION ON BUTTON CLICK
```

```

private async void TranslateButton_Click(object sender, EventArgs e)
{
    string textToTranslate = TextToTranslate.Text.Trim();

    string fromLanguage = FromLanguageComboBox.SelectedValue.ToString();
    string fromLanguageCode;

    // auto-detect source language if requested
    if (fromLanguage == "Detect")
    {
        fromLanguageCode = DetectLanguage(textToTranslate);
        if (!languageCodes.Contains(fromLanguageCode))
        {
            MessageBox.Show("The source language could not be detected automatically " +
                "or is not supported for translation.", "Language detection failed",
                MessageBoxButton.OK, MessageBoxImage.Error);
            return;
        }
    }
    else
        fromLanguageCode = languageCodesAndTitles[fromLanguage];

    string toLanguageCode = languageCodesAndTitles[ToLanguageComboBox.SelectedValue.ToString()];

    // spell-check the source text if the source language is English
    if (fromLanguageCode == "en")
    {
        if (textToTranslate.StartsWith("-")) // don't spell check in this case
            textToTranslate = textToTranslate.Substring(1);
        else
        {
            textToTranslate = CorrectSpelling(textToTranslate);
            TextToTranslate.Text = textToTranslate; // put corrected text into input field
        }
    }

    // handle null operations: no text or same source/target languages
    if (textToTranslate == "" || fromLanguageCode == toLanguageCode)
    {
        TranslatedTextLabel.Content = textToTranslate;
        return;
    }

    // send HTTP request to perform the translation
    string endpoint = string.Format(TEXT_TRANSLATION_API_ENDPOINT, "translate");
    string uri = string.Format(endpoint + "&from={0}&to={1}", fromLanguageCode, toLanguageCode);

    System.Object[] body = new System.Object[] { new { Text = textToTranslate } };
    var requestBody = JsonConvert.SerializeObject(body);

    using (var client = new HttpClient())
    using (var request = new HttpRequestMessage())
    {
        request.Method = HttpMethod.Post;
        request.RequestUri = new Uri(uri);
        request.Content = new StringContent(requestBody, Encoding.UTF8, "application/json");
        request.Headers.Add("Ocp-Apim-Subscription-Key", COGNITIVE_SERVICES_KEY);
        request.Headers.Add("Ocp-Apim-Subscription-Region", "westus");
        request.Headers.Add("X-ClientTraceId", Guid.NewGuid().ToString());

        var response = await client.SendAsync(request);
        var responseBody = await response.Content.ReadAsStringAsync();

        var result = JsonConvert.DeserializeObject<List<Dictionary<string, List<Dictionary<string,
string>>>>>(responseBody);
        var translation = result[0]["translations"][0]["text"];

        // Update the translation field
        TranslatedTextLabel.Content = translation;
    }
}

```

```
}
```

El primer paso es obtener los idiomas de origen y destino, así como el texto que el usuario introdujo en nuestro formulario. Si el idioma de origen se establece en **Detect** (Detectar), se llama a `DetectLanguage()` para determinar el idioma del texto de origen. El texto puede estar en un idioma que no admita Translator API. En ese caso, se muestra un mensaje para informar al usuario y devolver el texto sin traducir.

Si el idioma de origen es el inglés (así se haya especificado o detectado), se revisa la ortografía del texto con `CorrectSpelling()` y se aplican las correcciones. El texto corregido se agrega de nuevo en el área de texto para que el usuario vea que se ha realizado una corrección.

El código para traducir el texto debe parecer familiar: crear el identificador URI, crear una solicitud, enviarla y analizar la respuesta. La matriz JSON puede contener más de un objeto para traducir; sin embargo, esta aplicación solo requiere uno.

Después de una solicitud correcta, `TranslatedTextLabel.Content` se reemplaza por `translation`, que actualiza la interfaz de usuario para mostrar el texto traducido.

Ejecución de la aplicación de WPF

Y eso es todo, ahora tenemos una aplicación de traducción que funciona creada con WPF. Para ejecutar la aplicación, haga clic en el botón **Iniciar** en Visual Studio.

Código fuente

El código fuente del proyecto está disponible en GitHub.

- [Exploración del código fuente](#)

Pasos siguientes

[Referencia de Microsoft Translator Text API](#)

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.

Esta 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+76PVCmY1" 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 Traductor 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 Traductor Text API

Lo primero que necesita hacer es escribir una función para llamar a Traductor 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 agregará 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 = synthesizer.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 | Yating, 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 -->

<!-- End Speech Synthesis button -->

```

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](#)

Personalización de las traducciones de texto

13/01/2020 • 4 minutes to read • [Edit Online](#)

Traductor personalizado de Microsoft es una característica del servicio Microsoft Translator que permite a los usuarios personalizar la traducción automática neuronal avanzada de Microsoft Translator cuando se traduce texto con Translator Text API (solo en la versión 3).

Esta característica solamente puede utilizarse para personalizar la traducción de voz cuando se usa con el [servicio Voz de Cognitive Services](#).

Custom Translator

Con Custom Translator puede crear sistemas de traducción neuronal que comprendan la terminología usada en su propia empresa y sector. El sistema de traducción personalizada se integrará en las aplicaciones, los flujos de trabajo y los sitios web existentes.

¿Cómo funciona?

Utilice los documentos traducidos anteriormente (prospectos, páginas web, documentación, etc.) para crear un sistema de traducción que refleje la terminología y el estilo específicos de su dominio, mejor que un sistema de traducción estándar. Los usuarios pueden cargar documentos TMX, XLIFF, TXT, DOCX y XLSX.

El sistema también acepta datos que sean paralelos a nivel de documento, pero que aún no estén alineados a nivel de frase. Si los usuarios tienen acceso a versiones del mismo contenido en varios idiomas, pero en documentos independientes Custom Translator podrá hacer concordar automáticamente las frases de los distintos documentos. El sistema también puede utilizar datos monolingües en uno de los idiomas, o en ambos, para complementar los datos de aprendizaje paralelos, con el fin de mejorar las traducciones.

A partir de ese momento, el sistema personalizado está disponible con una llamada normal a Microsoft Translator Text API con el parámetro de categoría.

Si se proporcionan el tipo y la cantidad apropiados de datos de aprendizaje, no es extraño que con Custom Translator la calidad de la traducción mejore entre 5 y 10 puntos BLEU, o incluso más.

Puede encontrar más detalles acerca de los diferentes niveles de personalización en función de los datos disponibles en el [Manual del usuario de Custom Translator](#).

Microsoft Translator Hub

NOTE

Microsoft Translator Hub (heredado) se retirará el 17 de mayo de 2019. [Ver información importante y fechas de migración](#).

Custom Translator frente a Hub

	HUB	CUSTOM TRANSLATOR
Estado de la característica de personalización	Disponibilidad general	Disponibilidad general
Versión de Text API	Solo v2	Solo v3

	HUB	CUSTOM TRANSLATOR
Personalización de SMT	Sí	Sin
Personalización de NMT	Sin	Sí
Nueva personalización unificada de servicios de voz	Sin	Sí
Sin seguimiento	Sí	Sí

Marco de traducciones en colaboración

NOTE

A partir del 1 de febrero de 2018, `AddTranslation()` y `AddTranslationArray()` no se pueden usar con la versión 2.0 de Translator Text API. Estos métodos generarán un error y no se escribirá nada. La versión 3.0 de Translator Text API no admite estos métodos.

Pasos siguientes

[Configuración de un sistema de idiomas personalizado mediante Custom Translator](#)

Cómo hace Translator Text API para contar caracteres

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

Translator Text API cuenta cada punto de código Unicode de texto de entrada como un carácter. Cada traducción de un texto a un idioma se considera una traducción independiente, incluso si la solicitud se hizo en una única llamada API de traducción a varios idiomas. La longitud de la respuesta no importa.

Lo que importa es:

- El texto que se pasa a Translator Text API en el cuerpo de la solicitud
 - `Text` al usar los métodos Translate, Transliterate y Dictionary Lookup
 - `Text` y `Translation` al usar el método Dictionary Examples
- Todo el marcado: HTML, etiquetas XML, etc., en el campo de texto del cuerpo de la solicitud. La notación JSON que se usa para compilar la solicitud (por ejemplo "Text:") no se cuenta.
- Una letra individual
- Signos de puntuación
- Un espacio, tabulación, marcado y cualquier tipo de carácter de espacio en blanco
- Cada punto de código que se define en Unicode
- Una traducción repetida, incluso si ha traducido previamente el mismo texto

Para los scripts basados en ideogramas, como el chino y kanji japonés, Translator Text API aún contará el número de puntos de código Unicode, un carácter por ideograma. Excepción: Los suplentes Unicode cuentan como dos caracteres.

El número de solicitudes, palabras, frases o bytes es irrelevante en el recuento de caracteres.

Las llamadas a los métodos Detect y BreakSentence no se cuentan en el consumo de caracteres. Sin embargo, se espera que las llamadas a los métodos Detect y BreakSentence tengan una proporción razonable frente al uso de otras funciones que se cuentan. Si el número de llamadas Detect o BreakSentence que hace supera en 100 veces el número de otros métodos contados, Microsoft se reserva el derecho a restringir el uso de los métodos Detect y BreakSentence.

Para obtener más información sobre los recuentos de caracteres, consulte las [preguntas más frecuentes sobre Microsoft Translator](#).

Cómo suscribirse a Translator Text API

13/01/2020 • 3 minutes to read • [Edit Online](#)

Inicio de sesión en Azure Portal

- ¿No tiene una cuenta? Puede crear una [cuenta gratuita](#) para probarlo sin costo.
- ¿Ya tiene una cuenta? [Sign in](#)

Crear una suscripción a Translator Text API

Después de iniciar sesión el portal, puede crear una suscripción a Translator Text API como se indica a continuación:

1. Seleccione **+ Crear un recurso**.
2. En el cuadro de búsqueda **Search the Marketplace** (Buscar en Marketplace), escriba **Translator Text** y, después, selecciónelo en los resultados.
3. Haga clic en **Crear** para definir los detalles de la suscripción.
4. Seleccione el plan de tarifa que mejor se adapte a sus necesidades en la lista **Plan de tarifa**.
 - a. Cada suscripción tiene un plan gratuito. El plan gratuito tiene las mismas características y funcionalidades que los planes de pago y no expira.
 - b. Solo puede tener una suscripción gratuita en su cuenta.
5. Haga clic en **Crear** para terminar de crear la suscripción.

Clave de autenticación

Cuando se registra en Translator Text, obtendrá una clave de acceso personalizado única para su suscripción. Esta clave es necesaria en cada llamada a Translator Text API.

1. Para recuperar su clave de autenticación, seleccione primero la suscripción adecuada.
2. Seleccione **Claves** en la sección **Administración de recursos** de los detalles de la suscripción.
3. Copie cualquiera de las claves que se muestran para la suscripción.

Obtener información, realizar pruebas y obtener soporte técnico

- [Ejemplos de código en GitHub](#)
- [Foro de soporte técnico de Microsoft Translator](#)

Microsoft Translator generalmente permitirá las primeras dos solicitudes antes de comprobar el estado de la cuenta de la suscripción. Si las primeras solicitudes a Microsoft Translator API se realizan correctamente y, a continuación, las llamadas producen un error, la respuesta del error indicará el problema. Registre la respuesta de la API para ver el motivo.

Opciones de precios

- [Translator Text API](#)

Personalización

Use Custom Translator para personalizar las traducciones y crear un sistema de traducción en sintonía con su propio estilo y terminología, empezando por los sistemas genéricos de traducción automática neuronal de

Recursos adicionales

- [Introducción a Azure \(vídeo de 3 minutos\)](#)
- [Enviar una solicitud para pagar una suscripción de Azure mediante factura](#)

Migrar Translator Text API V2 a V3

13/01/2020 • 10 minutes to read • [Edit Online](#)

NOTE

V2 quedó en desuso el 30 de abril de 2018. Migre sus aplicaciones a V3 para aprovechar la nueva funcionalidad disponible exclusivamente en V3.

Microsoft Translator Hub se retirará el 17 de mayo de 2019. [Ver información y fechas de migración importantes.](#)

El equipo de Microsoft Translator ha lanzado la versión 3 (V3) de Translator Text API. En esta versión se incluyen nuevas características, métodos en desuso y un nuevo formato para enviar y recibir datos del servicio Microsoft Translator. Este documento proporciona información para cambiar las aplicaciones para que usen V3.

El final de este documento contiene vínculos útiles para que pueda obtener más información.

Resumen de características

- Sin seguimiento: en V3 se aplica la función Sin seguimiento a todos los niveles de precios en Azure Portal. Esta característica significa que Microsoft no guardará ningún texto enviado a la API V3.
- JSON: XML se reemplaza con JSON. Todos los datos enviados al servicio y recibidos desde el mismo están en formato JSON.
- Varios idiomas de destino en una única solicitud: el método de traducción acepta varios idiomas de destino ("a") para la traducción en una única solicitud. Por ejemplo, una sola solicitud puede traducirse "desde" el inglés "al" alemán, español y japonés, o a cualquier otro grupo de idiomas.
- Diccionario bilingüe: se ha agregado un método de diccionario bilingüe a la API. Este método incluye las opciones "búsqueda" y "ejemplos".
- Transliterar: se ha agregado un método de transliteración a la API. Este método convertirá las palabras y oraciones de un script (por ejemplo, en árabe) en otro script (por ejemplo, en latín).
- Idiomas: un nuevo método denominado "idiomas" ofrece información sobre el idioma en formato JSON, y se puede usar con los métodos "traducir", "diccionario" y "transliterar".
- Novedades en Traducir: se han agregado nuevas capacidades al método "traducir" para admitir algunas de las características que se encontraban en la API V2 como métodos separados. Un ejemplo es TranslateArray.
- Método leer: la funcionalidad de conversión de texto a voz ya no se admite en la API de Microsoft Translator. La funcionalidad de texto a voz está disponible en el [servicio Voz de Microsoft](#).

La siguiente lista de métodos V2 y V3 identifica los métodos V3 y las API que proporcionarán la funcionalidad de V2.

MÉTODO DE API V2	COMPATIBILIDAD DE API V3
<code>Translate</code>	Traducir
<code>TranslateArray</code>	Traducir
<code>GetLanguageNames</code>	Idiomas
<code>GetLanguagesForTranslate</code>	Idiomas

MÉTODO DE API V2	COMPATIBILIDAD DE API V3
<code>GetLanguagesForSpeak</code>	Servicio Voz de Microsoft
<code>Speak</code>	Servicio Voz de Microsoft
<code>Detect</code>	Detectar
<code>DetectArray</code>	Detectar
<code>AddTranslation</code>	Esta característica ya no se admite.
<code>AddTranslationArray</code>	Esta característica ya no se admite.
<code>BreakSentences</code>	BreakSentence
<code>GetTranslations</code>	Esta característica ya no se admite.
<code>GetTranslationsArray</code>	Esta característica ya no se admite.

Cambiar al formato JSON

Microsoft Translator Text Translation V2 aceptaba y devolvía datos en formato XML. En V3, todos los datos que se envían y se reciben mediante la API están en formato JSON. XML ya no se aceptará ni devolverá datos en V3.

Este cambio afectará a varios aspectos de una aplicación escrita para la versión V2 de Text Translation API. Por ejemplo: la API de idiomas devuelve información sobre el idioma que se usará en la traducción de texto, la transliteración y los dos métodos de diccionario. Puede solicitar toda la información del idioma para todos los métodos en una sola llamada o solicitarlos individualmente.

El método de idiomas no requiere autenticación; si hace clic en el siguiente vínculo podrá ver toda la información de idioma para V3 en JSON:

<https://api.cognitive.microsofttranslator.com/languages?api-version=3.0&scope=translation,diccionario,transliteración>

Clave de autenticación

La clave de autenticación que está utilizando para V2 se aceptará en V3. No necesita obtener una nueva suscripción. Podrá mezclar V2 y V3 en sus aplicaciones durante el período de migración de un año; así será más fácil lanzar de nuevas versiones mientras todavía está migrando de V2-XML a V3-JSON.

Modelo de precios

Microsoft Translator V3 tiene el mismo precio que V2; esto es, por carácter e incluyendo los espacios. Las nuevas características de V3 realizan algunos cambios en los caracteres que se cuentan para la facturación.

MÉTODO V3	CARACTERES QUE SE CUENTAN PARA LA FACTURACIÓN
<code>Languages</code>	Si no se envían caracteres, no se cuenta ninguno y no hay cargo.

MÉTODO V3	CARACTERES QUE SE CUENTAN PARA LA FACTURACIÓN
<code>Translate</code>	El recuento se basa en la cantidad de caracteres que se envían para la traducción, y en el número de idiomas a los que se traducen los caracteres. 50 caracteres enviados más 5 idiomas solicitados, serán 50x5.
<code>Transliterate</code>	Se cuenta el número de caracteres que se piden para la transliteración.
<code>Dictionary lookup & example</code>	Se cuentan el número de caracteres enviados para la búsqueda de diccionario y los ejemplos.
<code>BreakSentence</code>	Sin cargo.
<code>Detect</code>	Sin cargo.

Puntos de conexión de V3

Global

- api.cognitive.microsofttranslator.com

Métodos de traducción de texto de API V3

Languages
Translate
Transliterate
BreakSentence
Detect
Dictionary/lookup
Dictionary/example

Compatibilidad y personalización

NOTE

Microsoft Translator Hub se retirará el 17 de mayo de 2019. [Ver información y fechas de migración importantes.](#)

Microsoft Translator V3 usa la traducción automática neuronal por defecto. Por lo tanto, no puede utilizarse con Microsoft Translator Hub. Translator Hub solo admite traducción automática estadística heredada. La personalización de la traducción neuronal está disponible si usa el Traductor personalizado. [Obtenga más información sobre cómo personalizar la traducción automática neuronal](#)

La traducción neuronal con Text API V3 no admite el uso de categorías estándar (SMT, voz, tech, generalnn).

	PUNTO DE CONEXIÓN	COMPATIBLE CON EL PROCESADOR GDPR	UTILIZA TRANSLATOR HUB	UTILIZA TRADUCTOR PERSONALIZADO (VERSIÓN PRELIMINAR)
Translator Text API versión 2	api.microsofttranslator.com	Sin	Sí	Sin
Translator Text API versión 3	api.cognitive.microsofttranslator.com	Sí	Sin	Sí

Translator Text API versión 3

- Está disponible con carácter general y es totalmente compatible.
- Es compatible con GDPR como procesador y satisface todas las estipulaciones de ISO 20001 y 20018, así como los requisitos de certificación de SOC 3.
- Permite invocar los sistemas de traducción de redes neuronales que se han personalizado con Traductor personalizado (versión preliminar), la nueva característica de personalización NMT de Translator.
- No proporciona acceso a los sistemas de traducción personalizada creados con Microsoft Translator Hub.

Si usa el punto de conexión api.cognitive.microsofttranslator.com, esta utilizando la versión 3 de Translator Text API.

Translator Text API versión 2

- No cumple todos los requisitos de certificación de ISO 20001, 20018 y SOC 3.
- No permite invocar los sistemas de traducción de redes neuronales que se han personalizado con la característica Traductor personalizado.
- Proporciona acceso a los sistemas de traducción personalizada creados con Microsoft Translator Hub.
- Si usa el punto de conexión api.microsofttranslator.com, está utilizando la versión 2 de Translator Text API.

Ninguna versión de Translator API crea un registro de las traducciones. Las traducciones nunca se comparten con nadie. Obtenga más información en la página [No hay rastro de Translator](#).

Vínculos

- [Directiva de privacidad de Microsoft](#)
- [Información legal de Microsoft Azure](#)
- [Términos de Online Services](#).

Pasos siguientes

[Ver la documentación de V3.0](#)

Agregar el filtrado de blasfemias con Translator Text API

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

Normalmente, el servicio de Translator conserva las blasfemias que están presentes en el origen de la traducción. El grado de blasfemia y el contexto que hace que las palabras sean soeces difieren entre las distintas culturas. Como consecuencia, el grado de blasfemia en el lenguaje de destino podría ampliarse o reducirse.

Si quiere evitar obtener blasfemias en la traducción (independientemente de su presencia en el texto de origen), use la opción de filtrado de blasfemias disponible en el método Translate(). Esta opción le permite elegir si quiere ver las palabras soeces eliminadas o marcadas con etiquetas adecuadas, o bien si no quiere realizar ninguna acción.

El método Translate() toma el parámetro "options", que contiene el nuevo elemento "ProfanityAction". Los valores aceptados de ProfanityAction son "NoAction", "Marked" y "Deleted".

Valores aceptados de ProfanityAction y ejemplos

VALOR DE PROFANITYACTION	.	EJEMPLO: ORIGEN: JAPONÉS	EJEMPLO: DESTINO: INGLÉS
NoAction	Predeterminada. Igual que si no se configura la opción. Las blasfemias pasan del origen al destino.	彼は変態です。	Es un estúpido.
Marked	Las palabras soeces aparecerán rodeadas por etiquetas XML <profanity> ... </profanity>.	彼は変態です。	Es un <profanity>estúpido</profanity>.
Deleted	Las palabras soeces se quitan de la salida sin reemplazo.	彼は。	Es un.

Pasos siguientes

[Aplicar filtrado de blasfemias con la llamada a Translator API](#)

Cómo recibir información de alineación de palabras

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

Recibir información de alineación de palabras

Para recibir información de alineación, utilice el método Traducir e incluya el parámetro `includeAlignment` opcional.

Formato de información de alineación

La alineación se devuelve como un valor de cadena del siguiente formato para cada palabra del origen. La información de cada palabra se divide por un espacio, incluso en el caso de los idiomas no separados por espacios (scripts), como el chino:

`[[SourceTextStartIndex]:[SourceTextEndIndex]-[TgtTextStartIndex]:[TgtTextEndIndex]] *`

Cadena de alineación de ejemplo: "0:0-7:10 1:2-11:20 3:4-0:3 3:4-4:6 5:5-21:21".

En otras palabras, los dos puntos separan el índice inicial y final, el guión separa los idiomas y el espacio separa las palabras. Una palabra se puede alinear con ninguna, una o varias palabras de otro idioma y las palabras alineadas pueden ser no contiguas. Si no existe información de alineación disponible, el elemento `alignment` estará vacío. El método no devuelve ningún error en ese caso.

Restricciones

La alineación solo se devuelve para un subconjunto de pares de idiomas en este momento:

- de inglés a cualquier otro idioma;
- de cualquier otro idioma a inglés, excepto de chino simplificado, chino tradicional y letón a inglés;
- de japonés a coreano o de coreano a japonés. No recibirá información de alineación si la frase es una traducción preestablecida. Un ejemplo de traducción preestablecida es "Esto es una prueba", "Te quiero" y otras frases que se usan con mucha frecuencia.

Ejemplo

Ejemplo de JSON

```
[
  {
    "translations": [
      {
        "text": "Kann ich morgen Ihr Auto fahren?",
        "to": "de",
        "alignment": {
          "proj": "0:2-0:3 4:4-5:7 6:10-25:30 12:15-16:18 17:19-20:23 21:28-9:14 29:29-31:31"
        }
      }
    ]
  }
]
```

Impedir la traducción de contenido con Translator Text API

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

Translator Text API permite etiquetar contenido para que no se traduzca. Por ejemplo, es posible que desee etiquetar código, un nombre de marca, una palabra o una frase que no tengan sentido si se traducen.

Métodos para impedir la traducción

1. Utilice caracteres de escape con las etiquetas de Twitter: @somethingtopassthrough o #somethingtopassthrough. Quite los caracteres de escape después de la traducción. Esta es la expresión regular para las etiquetas válidas de Twitter: `\B@[A-Za-z]+[A-Za-z0-9_]+`. Una etiqueta debe empezar con un signo "@", seguido de un carácter y, a continuación, seguido de uno o varios caracteres, dígitos o subrayado.
2. Etiquete el contenido con `notranslate`. Esto solo funciona cuando el elemento textType de entrada se establece como HTML.

Ejemplo:

```
<span class="notranslate">This will not be translated.</span>
<span>This will be translated. </span>
```

```
<div class="notranslate">This will not be translated.</div>
<div>This will be translated. </div>
```

3. Utilice el [diccionario dinámico](#) para prescribir una traducción específica.
4. No pase la cadena a Translator Text API para que se traduzca.
5. Traductor personalizado: utilice un [diccionario de Traductor personalizado](#) para prescribir la traducción de una frase cuya probabilidad es del 100 %.

Pasos siguientes

[Impedir la traducción en la llamada API de Translator](#)

Cómo usar un diccionario dinámico

15/01/2020 • 2 minutes to read • [Edit Online](#)

Si ya conoce la traducción que quiere aplicar a una palabra o frase, puede proporcionarla como marcado dentro de la solicitud. El diccionario dinámico solo es seguro para los nombres compuestos, como nombres propios y nombres de productos.

Sintaxis:

```
<mstrans:dictionary translation="traducción de la frase">phrase</mstrans:dictionary>
```

Requisitos:

- Los idiomas `From` y `To` deben incluir inglés y otro idioma compatible.
- Debe incluir el parámetro `From` en la solicitud de traducción de API en lugar de usar la característica de detección automática.

Ejemplo: en-de:

Entrada de origen:

```
The word <mstrans:dictionary translation=\"wordomatic\">word or phrase</mstrans:dictionary> is a dictionary entry.
```

Salida de destino: `Das Wort "wordomatic" ist ein Wörterbucheintrag.`

Esta característica funciona del mismo modo con y sin el modo HTML.

Úsela con moderación. Una manera mejor de personalizar la traducción es usar el Traductor personalizado. Custom Translator hace un uso completo del contexto y las probabilidades estadísticas. Si tiene o puede crear datos de aprendizaje que muestren el trabajo o la frase en contexto, obtendrá resultados mucho mejores. Para más información sobre el traductor personalizado, consulte <https://aka.ms/CustomTranslator>.

2 minutes to read

2 minutes to read

Traducción detrás de los firewalls de direcciones IP con Translator Text API

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

Translator Text API puede traducir detrás de los firewalls utilizando el filtrado de nombres de dominio o de direcciones IP. El filtrado de nombres de dominio es el método preferido. **No se recomienda** ejecutar Microsoft Translator desde detrás de un firewall con filtrado de direcciones IP. Es probable que la configuración se interrumpa en el futuro sin previo aviso.

Direcciones IP de Translator

Direcciones IP de `api.cognitive.microsofttranslator.com` en API Microsoft Translator Text desde el 21 de agosto de 2019:

- **Asia Pacífico:** 20.40.125.208, 20.43.88.240, 20.184.58.62, 40.90.139.163, 104.44.89.44
- **Europa:** 40.90.138.4, 40.90.141.99, 51.105.170.64, 52.155.218.251
- **Norteamérica:** 40.90.139.36, 40.90.139.2, 40.119.2.134, 52.224.200.129, 52.249.207.163

Pasos siguientes

[Translate behind IP firewalls in your Translator API call](#) (Traducción detrás de los firewalls de direcciones IP en la llamada API de Translator)

Translator Text API v3.0

13/01/2020 • 15 minutes to read • [Edit Online](#)

Novedades

La versión 3 de Translator Text API proporciona una API web moderna basada en JSON. Mejora la facilidad de uso y rendimiento mediante la consolidación de las características existentes en menos operaciones y proporciona nuevas características.

- Transliteración para convertir texto en un idioma de un script a otro.
- Traducción a varios idiomas en una sola solicitud.
- Detección de idioma, traducción y transliteración en una sola solicitud.
- Diccionario para buscar traducciones alternativas de un término, traducciones inversas y ejemplos que muestren los términos usados en contexto.
- Resultados de detección de idioma más informativos.

Direcciones URL base

Microsoft Translator está disponible en varias ubicaciones de centros de datos. En la actualidad, se encuentran en diez [regiones de Azure](#):

- **América:** Este de EE. UU., Centro-sur de EE. UU. , Centro-oeste de EE. UU. y Oeste de EE. UU. 2
- **Asia Pacífico:** Sur de Corea del Sur, Japón Oriental, Sudeste Asiático y Este de Australia
- **Europa:** Europa del Norte y Europa Occidental

En la mayoría de los casos, las solicitudes dirigidas a Microsoft Translator Text API se administran en el centro de datos que está más próximo a la ubicación donde se originó la solicitud. En caso de que se produzca un error en un centro de datos, la solicitud puede enrutarse fuera de la geografía de Azure.

Para hacer que la solicitud se controle en una geografía de Azure específica, cambie el punto de conexión Global en la solicitud de API por el punto de conexión regional que desee:

DESCRIPCIÓN	GEOGRAFÍA DE AZURE	URL BASE
Azure	Global (no regional)	api.cognitive.microsofttranslator.com
Azure	Estados Unidos	api-nam.cognitive.microsofttranslator.com
Azure	Europa	api-eur.cognitive.microsofttranslator.com
Azure	Asia Pacífico	api-apc.cognitive.microsofttranslator.com

Authentication

Suscríbase a Translator Text API o a los [varios servicios de Cognitive Services](#) en Azure Cognitive Services y use la clave de suscripción (disponible en Azure Portal) para autenticarse.

Hay tres encabezados que puede usar para autenticar su suscripción. En esta tabla, se explica cómo se utiliza cada

uno de ellos:

ENCABEZADOS	DESCRIPCIÓN
Ocp-Apim-Subscription-Key	Úselo con la suscripción a Cognitive Services si pasa su clave secreta. El valor es la clave secreta de Azure para su suscripción a Translator Text API.
Authorization	Úselo con la suscripción a Cognitive Services si pasa un token de autenticación. El valor es el token de portador: <code>Bearer <token></code> .
Ocp-Apim-Subscription-Region	Úselo con una suscripción a varios servicios de Cognitive Services si pasa una clave secreta de varios servicios. El valor es la región de la suscripción a varios servicios. Este valor es opcional cuando no se usa una suscripción de varios servicios.

Clave secreta

La primera opción consiste en realizar la autenticación con el encabezado `Ocp-Apim-Subscription-Key`. Agregue el encabezado `Ocp-Apim-Subscription-Key: <YOUR_SECRET_KEY>` a la solicitud.

Token de autorización

Si lo desea, también puede cambiar la clave secreta por un token de acceso. Este token se incluirá en cada solicitud como un encabezado `Authorization`. Para obtener un token de autorización, realice una solicitud `POST` a la dirección URL siguiente:

ENTORNO	URL DEL SERVICIO DE AUTENTICACIÓN
Azure	<code>https://api.cognitive.microsoft.com/sts/v1.0/issueToken</code>

Estas son algunas solicitudes de ejemplo para obtener un token una vez proporcionada una clave secreta:

```
// Pass secret key using header
curl --header 'Ocp-Apim-Subscription-Key: <your-key>' --data ""
'https://api.cognitive.microsoft.com/sts/v1.0/issueToken'

// Pass secret key using query string parameter
curl --data "" 'https://api.cognitive.microsoft.com/sts/v1.0/issueToken?Subscription-Key=<your-key>'
```

Una solicitud correcta devuelve el token de acceso codificado como texto sin formato en el cuerpo de respuesta. El token válido se pasa al servicio Translator como un token de portador en la autorización.

```
Authorization: Bearer <Base64-access_token>
```

Un token de autenticación tiene una validez de 10 minutos. El token debe volver a usarse al realizar varias llamadas a las API de Translator. Sin embargo, si el programa realiza las solicitudes a la API de Translator durante un período de tiempo prolongado, el programa debe solicitar un nuevo token de acceso a intervalos regulares (por ejemplo, cada 8 minutos).

Suscripción a varios servicios

La última opción de autenticación consiste en utilizar la suscripción a varios servicios de Cognitive Services. Esta opción le permite utilizar una única clave secreta para autenticar las solicitudes de varios servicios.

Si usa una clave secreta para varios servicios, debe incluir dos encabezados de autenticación con la solicitud. El primero pasa la clave secreta, mientras que el segundo especifica la región asociada con la suscripción.

- `Ocp-Apim-Subscription-Key`
- `Ocp-Apim-Subscription-Region`

La región es obligatoria en la suscripción de varios servicios de Text API. La región seleccionada es la única región que puede usar para la traducción de texto al usar la clave de suscripción de varios servicios, y debe ser la misma región que seleccionó al registrarse en su suscripción de varios servicios a través de Azure Portal.

Las regiones disponibles son `australiaeast`, `brazilsouth`, `canadacentral`, `centralindia`, `centralus`, `centraluseuap`, `eastasia`, `eastus`, `eastus2`, `francecentral`, `japaneast`, `japanwest`, `koreacentral`, `northcentralus`, `northeurope`, `southcentralus`, `southeastasia`, `uksouth`, `westcentralus`, `westeurope`, `westus`, `westus2` y `southafricanorth`.

Si decide pasar la clave secreta en la cadena de consulta con el parámetro `Subscription-Key`, tendrá que especificar la región con el parámetro de consulta `Subscription-Region`.

Si utiliza un token de portador, tendrá que obtener el token del punto de conexión de la región:

`https://<your-region>.api.cognitive.microsoft.com/sts/v1.0/issueToken`.

Errors

Una respuesta de error estándar es un objeto JSON con el par de nombre/valor denominado `error`. El valor también es un objeto JSON con propiedades:

- `code`: código de error definido por el servidor.
- `message`: cadena que proporciona una representación legible del error.

Por ejemplo, un cliente con una suscripción de prueba gratuita recibiría el error siguiente una vez agotada la cuota gratuita:

```
{
  "error": {
    "code": 403001,
    "message": "The operation is not allowed because the subscription has exceeded its free quota."
  }
}
```

El código de error es un número de 6 dígitos que combina el código de estado HTTP de 3 dígitos y otro número de 3 dígitos que ayuda a categorizar aún más el error. Los códigos de error comunes son:

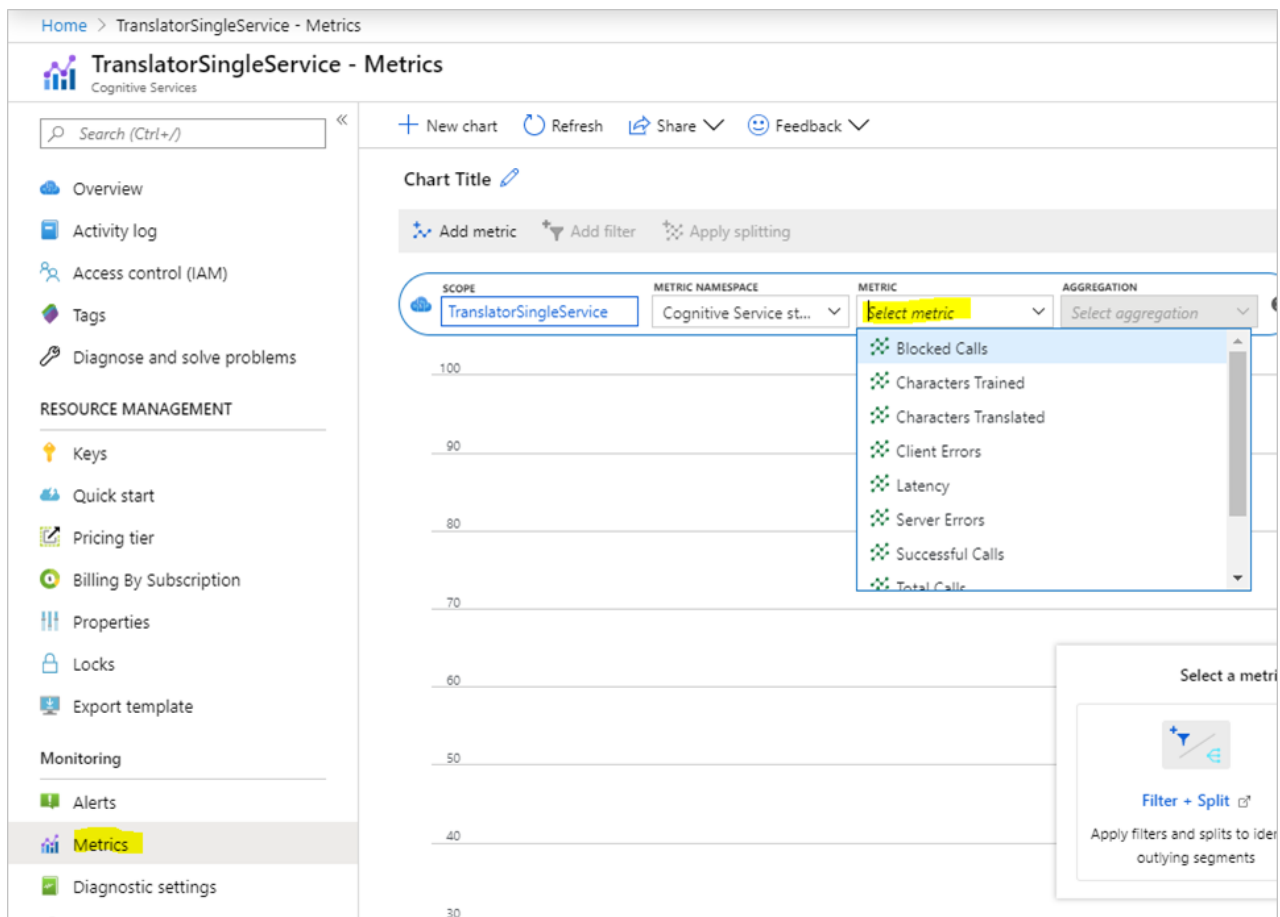
CÓDIGO	DESCRIPCIÓN
400000	Una de las entradas de la solicitud no es válida.
400001	El parámetro "scope" no es válido.
400002	El parámetro "category" no es válido.
400003	Falta un especificador de lenguaje o no es válido.
400004	Falta un especificador de script de destino ("To script") o no es válido.

CÓDIGO	DESCRIPCIÓN
400005	Falta un texto de entrada o no es válido.
400006	La combinación de lenguaje y script no es válida.
400018	Falta un especificador de script de origen ("From script") o no es válido.
400019	No se admite uno de los lenguajes especificados.
400020	Uno de los elementos de la matriz de texto de entrada no es válido.
400021	Falta un parámetro de la versión de API o no es válido.
400023	Uno de los pares de lenguaje especificados no es válido.
400035	El lenguaje fuente (campo "From") no es válido.
400036	Falta el lenguaje de destino (campo "To") o no es válido.
400042	Una de las opciones especificadas (campo "Options") no es válida.
400043	Falta el id. de seguimiento del cliente (campo ClientTraceld o encabezado X-ClientTraceld) o no es válido.
400050	El texto de entrada es demasiado largo. Consulte los límites de solicitud .
400064	Falta un parámetro "translation" o no es válido.
400070	El número de scripts de destino (parámetro ToScript) no coincide con el número de lenguajes de destino (parámetro To).
400071	Valor no válido para TextType.
400072	La matriz de texto de entrada tiene demasiados elementos.
400073	El parámetro de script no es válido.
400074	El cuerpo de la solicitud no es un JSON válido.
400075	La combinación de categoría y par de lenguaje no es válida.
400077	Se superó el tamaño máximo de solicitud. Consulte los límites de solicitud .
400079	El sistema personalizado solicitado para la traducción entre, desde y hacia el lenguaje no existe.
400080	La transliteración no se admite para el idioma o el script.

CÓDIGO	DESCRIPCIÓN
401000	La solicitud no está autorizada porque faltan las credenciales o no son válidas.
401015	"Las credenciales proporcionadas son para Speech API. Esta solicitud requiere credenciales para Text API. Use una suscripción a Translator Text API."
403000	No se permite la operación.
403001	No se permite la operación porque la suscripción superó su cuota disponible.
405000	No se admite el método de solicitud para el recurso solicitado.
408001	Se está preparando el sistema de traducción solicitado. Vuelva a intentarlo en unos minutos.
408002	La solicitud superó el tiempo de espera mientras se esperaba la secuencia entrante. El cliente no presentó una solicitud en el tiempo que el servidor estaba preparado para esperar. El cliente puede repetir la solicitud sin modificaciones en cualquier momento posterior.
415000	Falta el encabezado Content-Type o no es válido.
429000, 429001, 429002	El servidor rechazó la solicitud porque el cliente superó los límites de solicitudes.
500000	Se ha producido un error inesperado. Si el error continúa, notifíquelo con la fecha y hora del error, con el identificador de la solicitud del encabezado de respuesta X-RequestId y con el identificador de cliente del encabezado de solicitud X-ClientTraceId.
503000	El servicio no está disponible temporalmente. Inténtelo de nuevo. Si el error continúa, notifíquelo con la fecha y hora del error, con el identificador de la solicitud del encabezado de respuesta X-RequestId y con el identificador de cliente del encabezado de solicitud X-ClientTraceId.

Métricas

Las métricas le permiten ver la información de uso y disponibilidad del traductor en Azure Portal, en la sección de métricas, tal como se muestra en la captura de pantalla siguiente. Para obtener más información, vea [Métricas de datos y plataforma](#).



En esta tabla se enumeran las métricas disponibles con la descripción de cómo se usan para supervisar las llamadas a la API de traducción.

MÉTRICAS	DESCRIPCIÓN
TotalCalls	Número total de llamadas a API.
TotalTokenCalls	Número total de llamadas API a través del servicio de token mediante el token de autenticación.
SuccessfulCalls	Número de llamadas correctas.
TotalErrors	Número de llamadas con respuesta de error.
BlockedCalls	Número de llamadas que han superado la tasa o el límite de cuota.
ServerErrors	Número de llamadas con error interno del servidor (5XX).
ClientErrors	Número de llamadas con error en el cliente (4XX).
Latencia	Duración para completar la solicitud en milisegundos.
CharactersTranslated	Número total de caracteres de la solicitud entrante de texto.

Translator Text API 3.0: Lenguajes

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

Obtiene el conjunto de idiomas admitidos actualmente por otras operaciones de Translator Text API.

URL de la solicitud

Envíe una solicitud `GET` a:

```
https://api.cognitive.microsofttranslator.com/languages?api-version=3.0
```

Parámetros de solicitud

Los parámetros de solicitud que se pasaron en la cadena de consulta son:

PARÁMETRO DE CONSULTA	DESCRIPCIÓN
api-version	<i>Parámetro obligatorio.</i> Versión de la API que el cliente solicitó. El valor debe ser `3.0`.
scope	<i>*Parámetro opcional*.</i> Lista de nombres separados por comas que definen el grupo de idiomas que se devolverá. Los nombres de grupo permitidos son: `translation`, `transliteration` y `dictionary`. Si no se proporciona ningún ámbito, se devuelven todos los grupos, lo que es equivalente a pasar `scope=translation,transliteration,dictionary`. Para decidir qué conjunto de idiomas admitidos es adecuado para su escenario, vea la descripción del [objeto de respuesta](#response-body).

Los encabezados de solicitud son:

ENCABEZADOS	DESCRIPCIÓN
Accept-Language	<i>*Encabezado de solicitud opcional*.</i> Idioma que se usará para las cadenas de la interfaz de usuario. Algunos de los campos de la respuesta son nombres de idiomas o regiones. Utilice este parámetro para definir el idioma en que se devolverán los nombres. El idioma se especifica al proporcionar una etiqueta de idioma BCP 47 con el formato correcto. Por ejemplo, utilice el valor `fr` para solicitar nombres en francés o utilice el valor `zh-Hant` para solicitar nombres en chino tradicional. Los nombres se proporcionan en inglés cuando no se especifica un idioma de destino o cuando la localización no está disponible.
X-ClientTraceId	<i>*Encabezado de solicitud opcional*.</i> GUI generado por el cliente para identificar de forma única la solicitud.

No es obligatorio que la autenticación obtenga recursos de idioma.

Response body

Un cliente usa el parámetro de consulta `scope` para definir los grupos de idiomas en los que está interesado.

- `scope=translation` proporciona los idiomas admitidos para traducir texto de un idioma a otro;
- `scope=transliteration` proporciona funcionalidades para convertir texto en un idioma de un script a otro;

- `scope=dictionary` proporciona pares de idiomas para los que las operaciones de `Dictionary` devuelven datos.

Un cliente puede recuperar varios grupos simultáneamente mediante la especificación de una lista de nombres separados por comas. Por ejemplo, `scope=translation,transliteration,dictionary` devolvería los idiomas admitidos para todos los grupos.

Una respuesta correcta es un objeto JSON con una propiedad para cada grupo solicitado:

```
{
  "translation": {
    //... set of languages supported to translate text (scope=translation)
  },
  "transliteration": {
    //... set of languages supported to convert between scripts (scope=transliteration)
  },
  "dictionary": {
    //... set of languages supported for alternative translations and examples (scope=dictionary)
  }
}
```

El valor de cada propiedad es el siguiente.

- Propiedad `translation`

El valor de la propiedad `translation` es un diccionario de pares (clave, valor). Cada clave es una etiqueta de idioma BCP 47. Una clave identifica un idioma de origen o destino para la traducción de un texto. El valor asociado a la clave es un objeto JSON con propiedades que describen el idioma:

- `name`: nombre para mostrar del idioma en la configuración regional solicitada a través del encabezado `Accept-Language`.
- `nativeName`: nombre para mostrar del idioma en la configuración regional nativa del idioma.
- `dir`: direccionalidad, que es `rtl` para los idiomas de derecha a izquierda o `ltr` para los idiomas de izquierda a derecha.

Ejemplo:

```
{
  "translation": {
    ...
    "fr": {
      "name": "French",
      "nativeName": "Français",
      "dir": "ltr"
    },
    ...
  }
}
```

- Propiedad `transliteration`

El valor de la propiedad `transliteration` es un diccionario de pares (clave, valor). Cada clave es una etiqueta de idioma BCP 47. Una clave identifica un idioma para el que se puede convertir texto de un script a otro. El valor asociado a la clave es un objeto JSON con propiedades que describen el idioma y sus scripts admitidos:

- `name`: nombre para mostrar del idioma en la configuración regional solicitada a través del encabezado `Accept-Language`.

- `nativeName` : nombre para mostrar del idioma en la configuración regional nativa del idioma.
- `scripts` : lista de scripts de origen de la conversión. Cada elemento de la lista `scripts` tiene las siguientes propiedades:
 - `code` : código de identificación del script.
 - `name` : nombre para mostrar del script en la configuración regional solicitada a través del encabezado `Accept-Language` .
 - `nativeName` : nombre para mostrar del idioma en la configuración regional nativa del idioma.
 - `dir` : direccionalidad, que es `rtl` para los idiomas de derecha a izquierda o `ltr` para los idiomas de izquierda a derecha.
- `toScripts` : lista de scripts disponibles como destino para la conversión de texto. Cada elemento de la lista `toScripts` tiene las propiedades `code` , `name` , `nativeName` y `dir` descritas anteriormente.

Ejemplo:

```
{
  "transliteration": {
    ...
    "ja": {
      "name": "Japanese",
      "nativeName": "日本語",
      "scripts": [
        {
          "code": "Jpan",
          "name": "Japanese",
          "nativeName": "日本語",
          "dir": "ltr",
          "toScripts": [
            {
              "code": "Latn",
              "name": "Latin",
              "nativeName": "ラテン語",
              "dir": "ltr"
            }
          ]
        }
      ]
    },
    {
      "code": "Latn",
      "name": "Latin",
      "nativeName": "ラテン語",
      "dir": "ltr",
      "toScripts": [
        {
          "code": "Jpan",
          "name": "Japanese",
          "nativeName": "日本語",
          "dir": "ltr"
        }
      ]
    }
  ]
},
  ...
}
```

- Propiedad `dictionary`

El valor de la propiedad `dictionary` es un diccionario de pares (clave, valor). Cada clave es una etiqueta de idioma BCP 47. La clave identifica un idioma para el que existen traducciones alternativas y traducciones inversas disponibles. El valor es un objeto JSON que describe los idiomas de origen y destino con las traducciones disponibles:

- `name`: nombre para mostrar del idioma de origen en la configuración regional solicitada a través del encabezado `Accept-Language`.
- `nativeName`: nombre para mostrar del idioma en la configuración regional nativa del idioma.
- `dir`: direccionalidad, que es `rtl` para los idiomas de derecha a izquierda o `ltr` para los idiomas de izquierda a derecha.
- `translations`: lista de idiomas con traducciones alternativas y ejemplos para la consulta expresada en el idioma de origen. Cada elemento de la lista `translations` tiene las siguientes propiedades:
 - `name`: nombre para mostrar del idioma de destino en la configuración regional solicitada a través del encabezado `Accept-Language`.
 - `nativeName`: nombre para mostrar del idioma de destino en la configuración regional nativa del idioma de destino.
 - `dir`: direccionalidad, que es `rtl` para los idiomas de derecha a izquierda o `ltr` para los idiomas de izquierda a derecha.
 - `code`: código de idioma que identifica el idioma de destino.

Ejemplo:

```
"es": {
  "name": "Spanish",
  "nativeName": "Español",
  "dir": "ltr",
  "translations": [
    {
      "name": "English",
      "nativeName": "English",
      "dir": "ltr",
      "code": "en"
    }
  ]
},
```

La estructura del objeto de respuesta no cambiará sin un cambio en la versión de la API. Para la misma versión de la API, la lista de idiomas disponibles puede cambiar con el tiempo porque Microsoft Translator amplía continuamente la lista de idiomas admitidos en sus servicios.

La lista de idiomas admitidos no cambiará con frecuencia. Para ahorrar ancho de banda de red y mejorar la capacidad de respuesta, una aplicación cliente debe considerar el almacenamiento en caché de los recursos de idioma y la correspondiente etiqueta de entidad (`ETag`). A continuación, la aplicación cliente puede consultar periódicamente (por ejemplo, una vez cada 24 horas) al servicio para capturar el conjunto más reciente de idiomas admitidos. Pasar el valor `ETag` actual en un campo de encabezado `If-None-Match` permitirá al servicio optimizar la respuesta. Si no se ha modificado el recurso, el servicio devolverá el código de estado 304 y un cuerpo de respuesta vacío.

Encabezados de respuesta

ENCABEZADOS	DESCRIPCIÓN
-------------	-------------

ETag	Valor actual de la etiqueta de entidad para los grupos de idiomas admitidos solicitados. Para mejorar la eficacia de solicitudes posteriores, el cliente puede enviar el valor `ETag` en un campo de encabezado `If-None-Match`.
X-RequestId	Valor generado por el servicio para identificar la solicitud. Se usa para solucionar problemas.

Códigos de estado de respuesta

A continuación se indican los códigos de estado HTTP posibles que devuelve una solicitud.

CÓDIGO DE ESTADO	DESCRIPCIÓN
200	Correcta.
304	El recurso no se ha modificado desde la versión especificada por los encabezados de solicitud `If-None-Match`.
400	Uno de los parámetros de consulta falta o no es válido. Corrija los parámetros de la solicitud antes de volver a intentarlo.
429	El servidor rechazó la solicitud porque el cliente superó los límites de solicitudes.
500	Se ha producido un error inesperado. Si el error continúa, notifíquelo con: fecha y hora del error, identificador de la solicitud del encabezado de respuesta `X-RequestId` e identificador de cliente del encabezado de solicitud `X-ClientTraceId`.
503	Servidor no disponible temporalmente. Vuelva a intentarlo. Si el error continúa, notifíquelo con: fecha y hora del error, identificador de la solicitud del encabezado de respuesta `X-RequestId` e identificador de cliente del encabezado de solicitud `X-ClientTraceId`.

Si se produce un error, la solicitud también devolverá una respuesta de error JSON. El código de error es un número de 6 dígitos que combina el código de estado HTTP de 3 dígitos y otro número de 3 dígitos que ayuda a categorizar aún más el error. Códigos de error comunes que pueden encontrarse en la [página de referencia de Translator Text API v3](#).

Ejemplos

En el ejemplo siguiente se muestra cómo recuperar los idiomas admitidos para la traducción de texto.

```
curl "https://api.cognitive.microsofttranslator.com/languages?api-version=3.0&scope=translation"
```

Translator Text API 3.0: Translate

13/01/2020 • 26 minutes to read • [Edit Online](#)

Traduce el texto.

URL de la solicitud

Envíe una solicitud `POST` a:

```
https://api.cognitive.microsofttranslator.com/translate?api-version=3.0
```

Parámetros de solicitud

Los parámetros de solicitud que se pasaron en la cadena de consulta son:

Parámetros obligatorios

PARÁMETRO DE CONSULTA	DESCRIPCIÓN
api-version	<i>Parámetro obligatorio.</i> Versión de la API que el cliente solicitó. El valor debe ser <code>3.0</code> .
to	<i>Parámetro obligatorio.</i> Especifica el idioma del texto de salida. El idioma de destino debe ser uno de los idiomas admitidos que están incluidos en el ámbito <code>translation</code> . Por ejemplo, utilice <code>to=de</code> para traducir al alemán. Es posible traducir a varios idiomas simultáneamente mediante la repetición del parámetro en la cadena de consulta. Por ejemplo, utilice <code>to=de&to=it</code> para traducir al alemán e italiano.

Parámetros opcionales

PARÁMETRO DE CONSULTA	DESCRIPCIÓN
De	<i>Parámetro opcional.</i> Especifica el idioma del texto de entrada. Busque los idiomas que están disponibles desde los que realizar la traducción mediante la busca de idiomas admitidos con el ámbito <code>translation</code> . Si no se ha especificado el parámetro <code>from</code> , se aplica la detección de idioma automática para determinar el idioma de origen. Debe usar el parámetro <code>from</code> en lugar de la detección automática cuando use la característica de diccionario dinámico .
textType	<i>Parámetro opcional.</i> Define si el texto que se está traduciendo es texto sin formato o texto HTML. El código HTML debe ser un elemento completo y bien formado. Los valores posibles son <code>plain</code> (valor predeterminado) o <code>html</code> .
category	<i>Parámetro opcional.</i> Una cadena que especifica la categoría (dominio) de la traducción. Este parámetro se utiliza para obtener las traducciones de un sistema personalizado creado con Custom Translator . Agregue el identificador de categoría de los detalles del proyecto de Traductor personalizado a este parámetro para usar el sistema personalizado implementado. El valor predeterminado es <code>general</code> .

profanityAction	<p><i>Parámetro opcional.</i></p> <p>Especifica cómo se deben tratar las palabras soeces en las traducciones. Los valores posibles son <code>NoAction</code> (valor predeterminado), <code>Marked</code> o <code>Deleted</code>. Para entender las maneras de tratar las palabras soeces, consulte Control de palabras soeces.</p>
profanityMarker	<p><i>Parámetro opcional.</i></p> <p>Especifica cómo deben marcarse las palabras soeces en las traducciones. Los valores posibles son <code>Asterisk</code> (valor predeterminado) o <code>Tag</code>. Para entender las maneras de tratar las palabras soeces, consulte Control de palabras soeces.</p>
includeAlignment	<p><i>Parámetro opcional.</i></p> <p>Especifica si se debe incluir la proyección de la alineación del texto de origen en el texto traducido. Los valores posibles son <code>true</code> o <code>false</code> (valor predeterminado).</p>
includeSentenceLength	<p><i>Parámetro opcional.</i></p> <p>Especifica si se deben incluir los límites de oraciones del texto de entrada y el texto traducido. Los valores posibles son <code>true</code> o <code>false</code> (valor predeterminado).</p>
suggestedFrom	<p><i>Parámetro opcional.</i></p> <p>Especifica un idioma de reserva si no se puede identificar el idioma del texto de entrada. La detección de idioma automática se aplica cuando se omite el parámetro <code>from</code>. Si se produce un error en la detección, se asume el idioma <code>suggestedFrom</code>.</p>
fromScript	<p><i>Parámetro opcional.</i></p> <p>Especifica el script del texto de entrada.</p>
toScript	<p><i>Parámetro opcional.</i></p> <p>Especifica el script del texto traducido.</p>
allowFallback	<p><i>Parámetro opcional.</i></p> <p>Especifica que el servicio puede recurrir a un sistema general cuando no existe un sistema personalizado. Los valores posibles son <code>true</code> (valor predeterminado) o <code>false</code>.</p> <p><code>allowFallback=false</code> especifica que la traducción solo debe usar sistemas entrenados para la <code>category</code> especificada por la solicitud. Si una traducción del idioma X al idioma Y requiere de encadenamiento a través de un idioma puente E, entonces todos los sistemas de la cadena (X -> E y E -> Y) deberán estar personalizados y tener la misma categoría. Si no se encuentra ningún sistema con la categoría específica, la solicitud devolverá un código de estado de 400. <code>allowFallback=true</code> especifica que el servicio puede recurrir a un sistema general cuando no existe un sistema personalizado.</p>

Los encabezados de solicitud incluyen lo siguiente:

ENCABEZADOS	DESCRIPCIÓN
Encabezados de autenticación	<p><i>Encabezado de solicitud obligatorio.</i></p> <p>Consulte las opciones disponibles para la autenticación.</p>
Content-Type	<p><i>Encabezado de solicitud obligatorio.</i></p> <p>Especifica el tipo de contenido de la carga.</p> <p>El valor aceptado es <code>application/json; charset=UTF-8</code>.</p>
Content-Length	<p><i>Encabezado de solicitud obligatorio.</i></p> <p>Longitud del cuerpo de la solicitud.</p>

X-ClientTraceId

Opcional.

GUID generado por el cliente para identificar de forma única la solicitud. Puede omitir este encabezado si incluye el id. de seguimiento en la cadena de la consulta mediante un parámetro de consulta denominado `ClientTraceId`.

Cuerpo de la solicitud

El cuerpo de la solicitud es una matriz JSON. Cada elemento de la matriz es un objeto JSON con una propiedad de cadena denominada `Text`, que representa la cadena que se va a traducir.

```
[
  {
    "Text": "I would really like to drive your car around the block a few times."
  }
]
```

Se aplican las siguientes limitaciones:

- La matriz puede tener como máximo 100 elementos.
- El texto completo incluido en la solicitud no puede superar los 5000 caracteres, incluidos los espacios.

Response body

Una respuesta correcta es una matriz JSON con un resultado para cada cadena en la matriz de entrada. Un objeto del resultado incluye las siguientes propiedades:

- `detectedLanguage`: objeto que describe el idioma detectado mediante las siguientes propiedades:
 - `language`: cadena que representa el código del idioma detectado.
 - `score`: valor flotante que indica la confianza en el resultado. La puntuación varía entre cero y uno, y una puntuación baja indica una confianza baja.

La propiedad `detectedLanguage` solo está presente en el objeto de resultado cuando se solicita la detección de idioma automática.

- `translations`: matriz de resultados de la traducción. El tamaño de la matriz coincide con el número de idiomas de destino especificados a través del parámetro de consulta `to`. Cada elemento de la matriz incluye:
 - `to`: cadena que representa el código del idioma de destino.
 - `text`: cadena que proporciona el texto traducido.
 - `transliteration`: objeto que proporciona el texto traducido en el script especificado por el parámetro `toScript`.
 - `script`: cadena que especifica el script de destino.
 - `text`: cadena que proporciona el texto traducido en el script de destino.

El objeto `transliteration` no se incluye si no se produce la transliteración.

- `alignment`: objeto con una propiedad de cadena única denominada `proj`, que asigna el texto de entrada al texto traducido. La información de alineación solo se proporciona cuando el parámetro de solicitud `includeAlignment` es `true`. La alineación se devuelve como un valor de cadena del siguiente formato:

`[[SourceTextStartIndex]:[SourceTextEndIndex]-[TgtTextStartIndex]:[TgtTextEndIndex]]`. Los dos puntos separan el índice inicial y final, el guión separa los idiomas y el espacio separa las palabras.

Una palabra se puede alinear con ninguna, una o varias palabras de otro idioma y las palabras alineadas pueden ser no contiguas. Si no existe información de alineación disponible, el elemento `alignment` estará vacío. Consulte [Obtener información de alineación](#) para ver un ejemplo y restricciones.

- `sentLen` : objeto que devuelve los límites de las oraciones en los textos de entrada y salida.
- `srcSentLen` : matriz de enteros que representan las longitudes de las oraciones en el texto de entrada. La longitud de la matriz es el número de oraciones y los valores son la longitud de cada oración.
- `transSentLen` : matriz de enteros que representan las longitudes de las oraciones en el texto traducido. La longitud de la matriz es el número de oraciones y los valores son la longitud de cada oración.

Los límites de oraciones solo se incluyen cuando el parámetro de solicitud `includeSentenceLength` es `true`.

- `sourceText` : objeto con una propiedad de cadena única denominada `text`, que proporciona el texto de entrada en el script predeterminado del idioma de origen. La propiedad `sourceText` solo está presente cuando la entrada se expresa en un script que no es el habitual del idioma. Por ejemplo, si la entrada era árabe escrito en alfabeto latino, `sourceText.text` será el mismo texto árabe convertido al alfabeto árabe.

En la sección de [ejemplos](#) se proporcionan ejemplos de respuestas JSON.

Encabezados de respuesta

ENCABEZADOS	DESCRIPCIÓN
X-RequestId	Valor generado por el servicio para identificar la solicitud. Se usa para solucionar problemas.
Sistema X-MT	Especifica el tipo de sistema que se usó para la traducción en cada idioma de 'destino' solicitado para la traducción. El valor es una lista de cadenas separadas por comas. Cada cadena indica un tipo: <ul style="list-style-type: none">• Custom: la solicitud incluye un sistema personalizado y se usó al menos un sistema personalizado durante la traducción.• Team: todas las demás solicitudes.

Códigos de estado de respuesta

A continuación se indican los códigos de estado HTTP posibles que devuelve una solicitud.

CÓDIGO DE ESTADO	DESCRIPCIÓN
200	Correcta.
400	Uno de los parámetros de consulta falta o no es válido. Corrija los parámetros de la solicitud antes de volver a intentarlo.
401	No pudo autenticarse la solicitud. Compruebe que las credenciales que se especificaron sean correctas.
403	La solicitud no está autenticada. Compruebe los detalles del mensaje de error. Esto a menudo indica que todas las traducciones gratuitas que proporciona la suscripción de prueba se han agotado.

408	No se pudo satisfacer la solicitud porque falta un recurso. Compruebe los detalles del mensaje de error. Cuando se usa una <code>category</code> personalizada, a menudo, esto indica que el sistema de traducción personalizada todavía no está disponible para atender las solicitudes. Se debe reintentar la solicitud tras un período de espera (por ejemplo, 1 minuto).
429	El servidor rechazó la solicitud porque el cliente superó los límites de solicitudes.
500	Se ha producido un error inesperado. Si el error continúa, notifíquelo con: fecha y hora del error, identificador de la solicitud del encabezado de respuesta <code>X-RequestId</code> e identificador de cliente del encabezado de solicitud <code>X-ClientTraceId</code> .
503	Servidor no disponible temporalmente. Vuelva a intentarlo. Si el error continúa, notifíquelo con: fecha y hora del error, identificador de la solicitud del encabezado de respuesta <code>X-RequestId</code> e identificador de cliente del encabezado de solicitud <code>X-ClientTraceId</code> .

Si se produce un error, la solicitud también devolverá una respuesta de error JSON. El código de error es un número de 6 dígitos que combina el código de estado HTTP de 3 dígitos y otro número de 3 dígitos que ayuda a categorizar aún más el error. Códigos de error comunes que pueden encontrarse en la [página de referencia de Traductor Text API v3](#).

Ejemplos

Traducción de una única entrada

En este ejemplo se muestra cómo traducir una única oración del inglés al chino simplificado.

```
curl -X POST "https://api.cognitive.microsofttranslator.com/translate?api-version=3.0&from=en&to=zh-Hans" -H "Ocp-Apim-Subscription-Key: <client-secret>" -H "Content-Type: application/json; charset=UTF-8" -d "[{'Text':'Hello, what is your name?'}]"
```

El cuerpo de la respuesta es:

```
[
  {
    "translations":[
      {"text":"你好，你叫什么名字？","to":"zh-Hans"}
    ]
  }
]
```

La matriz `translations` incluye un elemento, que proporciona la traducción de la parte única de texto en la entrada.

Traducción de una única entrada con la detección de idioma automática

En este ejemplo se muestra cómo traducir una única oración del inglés al chino simplificado. La solicitud no especifica el idioma de entrada. En su lugar, se usa la detección automática del idioma de origen.

```
curl -X POST "https://api.cognitive.microsofttranslator.com/translate?api-version=3.0&to=zh-Hans" -H "Ocp-Apim-Subscription-Key: <client-secret>" -H "Content-Type: application/json; charset=UTF-8" -d "[{'Text':'Hello, what is your name?'}]"
```

El cuerpo de la respuesta es:

```
[
  {
    "detectedLanguage": {"language": "en", "score": 1.0},
    "translations": [
      {"text": "你好，你叫什么名字？", "to": "zh-Hans"}
    ]
  }
]
```

La respuesta es similar a la respuesta del ejemplo anterior. Dado que se solicitó la detección de idioma automática, la respuesta también incluye información sobre el idioma detectado del texto de entrada.

Traducción con transliteración

Vamos a ampliar el ejemplo anterior mediante la adición de la transliteración. La siguiente solicitud pide una traducción de chino escrita en alfabeto latino.

```
curl -X POST "https://api.cognitive.microsofttranslator.com/translate?api-version=3.0&to=zh-Hans&toScript=Latn" -H "Ocp-Apim-Subscription-Key: <client-secret>" -H "Content-Type: application/json; charset=UTF-8" -d "[{'Text': 'Hello, what is your name?'}]"
```

El cuerpo de la respuesta es:

```
[
  {
    "detectedLanguage": {"language": "en", "score": 1.0},
    "translations": [
      {
        "text": "你好，你叫什么名字？",
        "transliteration": {"script": "Latn", "text": "nǐ hǎo , nǐ jiào shén me míng zì ?"},
        "to": "zh-Hans"
      }
    ]
  }
]
```

El resultado de la traducción incluye ahora una propiedad `transliteration`, que proporciona el texto traducido con caracteres latinos.

Traducción de varios fragmentos de texto

Traducir varias cadenas a la vez es simplemente cuestión de especificar una matriz de cadenas en el cuerpo de la solicitud.

```
curl -X POST "https://api.cognitive.microsofttranslator.com/translate?api-version=3.0&from=en&to=zh-Hans" -H "Ocp-Apim-Subscription-Key: <client-secret>" -H "Content-Type: application/json; charset=UTF-8" -d "[{'Text': 'Hello, what is your name?'}, {'Text': 'I am fine, thank you.'}]"
```

El cuerpo de la respuesta es:

```
[
  {
    "translations":[
      {"text":"你好，你叫什么名字？","to":"zh-Hans"}
    ]
  },
  {
    "translations":[
      {"text":"我很好，谢谢你。","to":"zh-Hans"}
    ]
  }
]
```

Traducción a varios idiomas

En este ejemplo se muestra cómo traducir la misma entrada a diferentes idiomas en una sola solicitud.

```
curl -X POST "https://api.cognitive.microsofttranslator.com/translate?api-version=3.0&from=en&to=zh-Hans&to=de" -H "Ocp-Apim-Subscription-Key: <client-secret>" -H "Content-Type: application/json; charset=UTF-8" -d "[{'Text':'Hello, what is your name?'}]"
```

El cuerpo de la respuesta es:

```
[
  {
    "translations":[
      {"text":"你好，你叫什么名字？","to":"zh-Hans"},
      {"text":"Hallo, was ist dein Name?","to":"de"}
    ]
  }
]
```

Control de palabras soeces

Normalmente, el servicio de Traductor conserva en la traducción las palabras soeces que están presentes en el origen. El grado de palabras soeces y el contexto que las convierte en ello difieren entre culturas y, como resultado, el grado de palabras soeces del idioma de destino puede aumentar o reducirse.

Si quiere evitar obtener palabras soeces en la traducción, independientemente de su presencia en el texto de origen, puede usar la opción de filtrado de palabras soeces. La opción permite elegir si quiere que se eliminen las palabras soeces, si quiere marcarlas con etiquetas adecuadas (lo que le ofrece la opción de agregar su propio postprocesamiento) o si no quiere que se realice ninguna acción. Los valores aceptados de `ProfanityAction` son `Deleted`, `Marked` y `NoAction` (valor predeterminado).

PROFANITYACTION	.
<code>NoAction</code>	<p>Este es el comportamiento predeterminado. Las palabras soeces pasarán del origen al destino.</p> <p>Ejemplo de origen (japonés): 彼はジャッカスです。 Ejemplo de traducción (español): Es un idiota.</p>
<code>Deleted</code>	<p>Las palabras soeces se quitarán de la salida sin reemplazo.</p> <p>Ejemplo de origen (japonés): 彼はジャッカスです。 Ejemplo de traducción (español): Es un.</p>

Marked

Las palabras soeces se reemplazan por un marcador en la salida. El marcador depende del parámetro `ProfanityMarker`.

En el caso de `ProfanityMarker=Asterisk`, las palabras soeces se reemplazan por `***`:

Ejemplo de origen (japonés): 彼はジャッカスです。

Ejemplo de traducción (español): Es un `***`.

En el caso de `ProfanityMarker=Tag`, las palabras soeces aparecerán rodeadas por las etiquetas XML `<profanity>` y `</profanity>`:

Ejemplo de origen (japonés): 彼はジャッカスです。

Ejemplo de traducción (español): Es un `<profanity>estúpido</profanity>`.

Por ejemplo:

```
curl -X POST "https://api.cognitive.microsofttranslator.com/translate?api-version=3.0&from=en&to=de&profanityAction=Marked" -H "Ocp-Apim-Subscription-Key: <client-secret>" -H "Content-Type: application/json; charset=UTF-8" -d "[{'Text':'This is a freaking good idea.'}]"
```

Esto devuelve:

```
[
  {
    "translations":[
      {"text":"Das ist eine *** gute Idee.", "to":"de"}
    ]
  }
]
```

Comparar con:

```
curl -X POST "https://api.cognitive.microsofttranslator.com/translate?api-version=3.0&from=en&to=de&profanityAction=Marked&profanityMarker=Tag" -H "Ocp-Apim-Subscription-Key: <client-secret>" -H "Content-Type: application/json; charset=UTF-8" -d "[{'Text':'This is a freaking good idea.'}]"
```

La última solicitud devuelve:

```
[
  {
    "translations":[
      {"text":"Das ist eine <profanity>verdammt</profanity> gute Idee.", "to":"de"}
    ]
  }
]
```

Traducción del contenido con marcado y decisión de qué traducir

Es común traducir el contenido que incluye marcado como contenido de una página HTML o de un documento XML. Incluya el parámetro de consulta `textType=html` al traducir contenido con etiquetas. Además, a veces resulta útil para excluir contenido específico de la traducción. Puede utilizar el atributo `class=notranslate` para especificar contenido que debería permanecer en el idioma original. En el ejemplo siguiente, el contenido del primer elemento `div` no se traducirá, mientras que el del segundo elemento `div`, sí.

```
<div class="notranslate">This will not be translated.</div>
<div>This will be translated. </div>
```

A continuación, se muestra una solicitud de ejemplo para ilustrar.

```
curl -X POST "https://api.cognitive.microsofttranslator.com/translate?api-version=3.0&from=en&to=zh-Hans&textType=html" -H "Ocp-Apim-Subscription-Key: <client-secret>" -H "Content-Type: application/json; charset=UTF-8" -d "[{'Text':'<div class=\"notranslate\">This will not be translated.</div><div>This will be translated.</div>}']"
```

La respuesta es:

```
[
  {
    "translations":[
      {
        "text":"<div class=\"notranslate\">This will not be translated.</div><div>这将被翻译。</div>","to":"zh-Hans"}
    ]
  }
]
```

Obtención de información de alineación

Para recibir información de alineación, especifique `includeAlignment=true` en la cadena de consulta.

```
curl -X POST "https://api.cognitive.microsofttranslator.com/translate?api-version=3.0&from=en&to=fr&includeAlignment=true" -H "Ocp-Apim-Subscription-Key: <client-secret>" -H "Content-Type: application/json; charset=UTF-8" -d "[{'Text':'The answer lies in machine translation.'}]"
```

La respuesta es:

```
[
  {
    "translations":[
      {
        "text":"La réponse se trouve dans la traduction automatique.",
        "to":"fr",
        "alignment":{"proj":"0:2-0:1 4:9-3:9 11:14-11:19 16:17-21:24 19:25-40:50 27:37-29:38 38:38-51:51"}
      }
    ]
  }
]
```

La información de alineación empieza por `0:2-0:1`, lo que significa que los tres primeros caracteres del texto de origen (`The`) se asignan a los dos primeros caracteres del texto traducido (`La`).

Limitaciones

Tenga en cuenta las restricciones que se indican a continuación:

- La alineación no está disponible para texto en formato HTML, es decir, `textType = html`.
- La alineación solo se devuelve para un subconjunto de pares de idiomas:
 - de inglés a cualquier otro idioma;
 - de cualquier otro idioma a inglés, excepto de chino simplificado, chino tradicional y letón a inglés;
 - de japonés a coreano, o viceversa.
- No recibirá alineación si la oración es una traducción preestablecida. Un ejemplo de traducción preestablecida es "Esto es una prueba", "Te quiero" y otras frases que se usan con mucha frecuencia.

Obtención de los límites de oración

Para recibir información acerca de la longitud de la oración del texto de origen y del texto traducido, especifique `includeSentenceLength=true` en la cadena de consulta.

```
curl -X POST "https://api.cognitive.microsofttranslator.com/translate?api-version=3.0&from=en&to=fr&includeSentenceLength=true" -H "Ocp-Apim-Subscription-Key: <client-secret>" -H "Content-Type: application/json; charset=UTF-8" -d "[{'Text':'The answer lies in machine translation. The best machine translation technology cannot always provide translations tailored to a site or users like a human. Simply copy and paste a code snippet anywhere.'}]"
```

La respuesta es:

```
[
  {
    "translations":[
      {
        "text":"La réponse se trouve dans la traduction automatique. La meilleure technologie de traduction automatique ne peut pas toujours fournir des traductions adaptées à un site ou des utilisateurs comme un être humain. Il suffit de copier et coller un extrait de code n'importe où.",
        "to":"fr",
        "sentLen":{"srcSentLen":[40,117,46],"transSentLen":[53,157,62]}
      }
    ]
  }
]
```

Traducción con diccionario dinámico

Si ya conoce la traducción que quiere aplicar a una palabra o frase, puede proporcionarla como marcado dentro de la solicitud. El diccionario dinámico solo es seguro para los nombres compuestos, como nombres propios y nombres de productos.

El marcado que se va a proporcionar utiliza la sintaxis siguiente.

```
<mstrans:dictionary translation="translation of phrase">phrase</mstrans:dictionary>
```

Por ejemplo, considere la frase en inglés "The word wordomatic is a dictionary entry" (La palabra "wordomatic" es una entrada del diccionario). Para conservar la palabra *wordomatic* en la traducción, envíe la solicitud:

```
curl -X POST "https://api.cognitive.microsofttranslator.com/translate?api-version=3.0&from=en&to=de" -H "Ocp-Apim-Subscription-Key: <client-secret>" -H "Content-Type: application/json; charset=UTF-8" -d "[{'Text':'The word <mstrans:dictionary translation=\"wordomatic\">word or phrase</mstrans:dictionary> is a dictionary entry.'}]"
```

El resultado es el siguiente:

```
[
  {
    "translations":[
      {
        "text":"Das Wort \"wordomatic\" ist ein Wörterbucheintrag.",
        "to":"de"
      }
    ]
  }
]
```

Esta característica funciona del mismo modo con `textType=text` o con `textType=html`. La característica debe usarse con moderación. La mejor forma de personalización de traducción, y más adecuada, es mediante el uso de Custom Translator. Custom Translator hace un uso completo del contexto y las probabilidades estadísticas. Si tiene o puede permitirse crear datos de entrenamiento que muestren el trabajo o una frase en contexto, obtendrá resultados mucho mejores. [Más información acerca de Custom Translator](#).

Translator Text API 3.0: Transliterar

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

Convierte el texto en un idioma de un script a otro.

URL de la solicitud

Envíe una solicitud `POST` a:

```
https://api.cognitive.microsofttranslator.com/transliterate?api-version=3.0
```

Parámetros de solicitud

Los parámetros de solicitud que se pasaron en la cadena de consulta son:

PARÁMETRO DE CONSULTA	DESCRIPCIÓN
api-version	*Parámetro obligatorio*. Versión de la API que el cliente solicitó. El valor debe ser `3.0`.
language	*Parámetro obligatorio*. Especifica el idioma del texto que se va a convertir de un script a otro. Los idiomas posibles se enumeran en el ámbito `transliteration` obtenido al consultar el servicio de sus [idiomas admitidos](./v3-0-languages.md).
fromScript	*Parámetro obligatorio*. Especifica el script que usa el texto de entrada. Busque los [idiomas admitidos](./v3-0-languages.md) con el ámbito `transliteration`, para ver los scripts de entrada disponibles para el idioma seleccionado.
toScript	*Parámetro obligatorio*. Especifica el nombre del script de salida. Busque los [idiomas admitidos](./v3-0-languages.md) con el ámbito `transliteration`, para ver los scripts de salida disponibles para la combinación de idiomas y el script de entrada seleccionados.

Los encabezados de solicitud incluyen lo siguiente:

ENCABEZADOS	DESCRIPCIÓN
Encabezados de autenticación	<i>Encabezado de solicitud obligatorio.</i> Consulte las opciones disponibles para la autenticación .
Content-Type	*Encabezado de solicitud obligatorio*. Especifica el tipo de contenido de la carga. Los valores posibles son: `application/json`.
Content-Length	*Encabezado de solicitud obligatorio*. Longitud del cuerpo de la solicitud.
X-ClientTraceId	*Opcional*. GUID generado por el cliente para identificar de forma única la solicitud. Puede omitir este encabezado si incluye el id. de seguimiento en la cadena de la consulta mediante un parámetro de consulta denominado `ClientTraceId`.

Cuerpo de la solicitud

El cuerpo de la solicitud es una matriz JSON. Cada elemento de la matriz es un objeto JSON con una propiedad de cadena denominada `Text`, que representa la cadena que se va a convertir.

```
[
  {
    "Text": "こんにちは",
  },
  {
    "Text": "さようなら"
  }
]
```

Se aplican las siguientes limitaciones:

- La matriz puede tener como máximo 10 elementos.
- El valor de texto de un elemento de la matriz no puede superar los 1 000 caracteres, incluyendo los espacios.
- El texto completo incluido en la solicitud no puede superar los 5000 caracteres, incluidos los espacios.

Response body

Una respuesta correcta es una matriz JSON, que cuenta con un resultado para cada elemento en la matriz de entrada. Un objeto del resultado incluye las siguientes propiedades:

- `text`: cadena que es el resultado de convertir la cadena de entrada en el script de salida.
- `script`: cadena que especifica el script que se usa en la salida.

Un ejemplo de respuesta JSON es:

```
[
  {
    "text": "konnichiha", "script": "Latn"},
  {
    "text": "sayounara", "script": "Latn"}
]
```

Encabezados de respuesta

ENCABEZADOS	DESCRIPCIÓN
X-RequestId	Valor generado por el servicio para identificar la solicitud. Se usa para solucionar problemas.

Códigos de estado de respuesta

A continuación se indican los códigos de estado HTTP posibles que devuelve una solicitud.

CÓDIGO DE ESTADO	DESCRIPCIÓN
200	Correcta.
400	Uno de los parámetros de consulta falta o no es válido. Corrija los parámetros de la solicitud antes de volver a intentarlo.
401	No pudo autenticarse la solicitud. Compruebe que las credenciales que se especificaron sean correctas.
403	La solicitud no está autenticada. Compruebe los detalles del mensaje de error. Esto a menudo indica que todas las traducciones gratuitas que proporciona la suscripción de prueba se han agotado.

429	El servidor rechazó la solicitud porque el cliente superó los límites de solicitudes.
500	Se ha producido un error inesperado. Si el error continúa, notifíquelo con: fecha y hora del error, identificador de la solicitud del encabezado de respuesta `X-RequestId` e identificador de cliente del encabezado de solicitud `X-ClientTraceId`.
503	Servidor no disponible temporalmente. Vuelva a intentarlo. Si el error continúa, notifíquelo con: fecha y hora del error, identificador de la solicitud del encabezado de respuesta `X-RequestId` e identificador de cliente del encabezado de solicitud `X-ClientTraceId`.

Si se produce un error, la solicitud también devolverá una respuesta de error JSON. El código de error es un número de 6 dígitos que combina el código de estado HTTP de 3 dígitos y otro número de 3 dígitos que ayuda a categorizar aún más el error. Códigos de error comunes que pueden encontrarse en la [página de referencia de Translator Text API v3](#).

Ejemplos

En el ejemplo siguiente se muestra cómo convertir dos cadenas escritas en japonés a japonés escrito con el alfabeto latino.

Esta es la carga de JSON para la solicitud en este ejemplo:

```
[{"text": "こんにちは", "script": "jpan"}, {"text": "さようなら", "script": "jpan"}]
```

Si está usando cURL en una ventana de línea de comandos que no admite caracteres Unicode, use la siguiente carga JSON y guárdela en un archivo denominado `request.txt`. Asegúrese de guardar el archivo con una codificación `UTF-8`.

```
curl -X POST "https://api.cognitive.microsofttranslator.com/transliterate?api-version=3.0&language=ja&fromScript=Jpan&toScript=Latn" -H "X-ClientTraceId: 875030C7-5380-40B8-8A03-63DACC69C11" -H "Ocp-Apim-Subscription-Key: <client-secret>" -H "Content-Type: application/json" -d @request.txt
```

Translator Text API 3.0: Detect

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

Identifica el idioma de un fragmento de texto.

URL de la solicitud

Envíe una solicitud `POST` a:

```
https://api.cognitive.microsofttranslator.com/detect?api-version=3.0
```

Parámetros de solicitud

Los parámetros de solicitud que se pasaron en la cadena de consulta son:

PARÁMETRO DE CONSULTA	DESCRIPCIÓN
api-version	*Parámetro obligatorio*. Versión de la API que el cliente solicitó. El valor debe ser `3.0`.

Los encabezados de solicitud incluyen lo siguiente:

ENCABEZADOS	DESCRIPCIÓN
Encabezados de autenticación	<i>Encabezado de solicitud obligatorio.</i> Consulte las opciones disponibles para la autenticación .
Content-Type	*Encabezado de solicitud obligatorio*. Especifica el tipo de contenido de la carga. Los valores posibles son: `application/json`.
Content-Length	*Encabezado de solicitud obligatorio*. Longitud del cuerpo de la solicitud.
X-ClientTraceld	*Opcional*. GUID generado por el cliente para identificar de forma única la solicitud. Puede omitir este encabezado si incluye el id. de seguimiento en la cadena de la consulta mediante un parámetro de consulta denominado `ClientTraceld`.

Cuerpo de la solicitud

El cuerpo de la solicitud es una matriz JSON. Cada elemento de la matriz es un objeto JSON con una propiedad de cadena denominada `Text`. La detección de idioma se aplica al valor de la propiedad `Text`. Un cuerpo de solicitud de muestra tiene el siguiente aspecto:

```
[  
  { "Text": "Ich würde wirklich gern Ihr Auto um den Block fahren ein paar Mal." }  
]
```

Se aplican las siguientes limitaciones:

- La matriz puede tener como máximo 100 elementos.
- El valor de texto de un elemento de la matriz no puede superar los 10 000 caracteres, incluyendo los espacios.
- El texto completo incluido en la solicitud no puede superar los 50 000 caracteres, incluidos los espacios.

Response body

Una respuesta correcta es una matriz JSON con un resultado para cada cadena en la matriz de entrada. Un objeto del resultado incluye las siguientes propiedades:

- `language`: Código del idioma detectado.
- `score`: valor flotante que indica la confianza en el resultado. La puntuación varía entre cero y uno, y una puntuación baja indica una confianza baja.
- `isTranslationSupported`: Valor booleano que tiene el valor "true" si el idioma detectado es uno de los idiomas admitidos para la traducción de texto.
- `isTransliterationSupported`: Valor booleano que tiene el valor "true" si el idioma detectado es uno de los idiomas admitidos para la transliteración.
- `alternatives`: Matriz de otros idiomas posibles. Cada elemento de la matriz es otro objeto con las mismas propiedades enumeradas anteriormente: `language`, `score`, `isTranslationSupported` y `isTransliterationSupported`.

Un ejemplo de respuesta JSON es:

```
[
  {
    "language": "de",
    "score": 0.92,
    "isTranslationSupported": true,
    "isTransliterationSupported": false,
    "alternatives": [
      {
        "language": "pt",
        "score": 0.23,
        "isTranslationSupported": true,
        "isTransliterationSupported": false
      },
      {
        "language": "sk",
        "score": 0.23,
        "isTranslationSupported": true,
        "isTransliterationSupported": false
      }
    ]
  }
]
```

Encabezados de respuesta

ENCABEZADOS	DESCRIPCIÓN
X-RequestId	Valor generado por el servicio para identificar la solicitud. Se usa para solucionar problemas.

Códigos de estado de respuesta

A continuación se indican los códigos de estado HTTP posibles que devuelve una solicitud.

CÓDIGO DE ESTADO	DESCRIPCIÓN
200	Correcta.
400	Uno de los parámetros de consulta falta o no es válido. Corrija los parámetros de la solicitud antes de volver a intentarlo.
401	No pudo autenticarse la solicitud. Compruebe que las credenciales que se especificaron sean correctas.
403	La solicitud no está autenticada. Compruebe los detalles del mensaje de error. Esto a menudo indica que todas las traducciones gratuitas que proporciona la suscripción de prueba se han agotado.
429	El servidor rechazó la solicitud porque el cliente superó los límites de solicitudes.
500	Se ha producido un error inesperado. Si el error continúa, notifíquelo con: fecha y hora del error, identificador de la solicitud del encabezado de respuesta `X-RequestId` e identificador de cliente del encabezado de solicitud `X-ClientTraceId`.
503	Servidor no disponible temporalmente. Vuelva a intentarlo. Si el error continúa, notifíquelo con: fecha y hora del error, identificador de la solicitud del encabezado de respuesta `X-RequestId` e identificador de cliente del encabezado de solicitud `X-ClientTraceId`.

Si se produce un error, la solicitud también devolverá una respuesta de error JSON. El código de error es un número de 6 dígitos que combina el código de estado HTTP de 3 dígitos y otro número de 3 dígitos que ayuda a categorizar aún más el error. Códigos de error comunes que pueden encontrarse en la [página de referencia de Translator Text API v3](#).

Ejemplos

En el ejemplo siguiente se muestra cómo recuperar los idiomas admitidos para la traducción de texto.

```
curl -X POST "https://api.cognitive.microsofttranslator.com/detect?api-version=3.0" -H "Ocp-Apim-Subscription-Key: <client-secret>" -H "Content-Type: application/json" -d "[{'Text': 'What language is this text written in?'}]"
```

Translator Text API 3.0: BreakSentence

13/01/2020 • 7 minutes to read • [Edit Online](#)

Identifica el posicionamiento de los límites de las oraciones en un fragmento de texto.

URL de la solicitud

Envíe una solicitud `POST` a:

```
https://api.cognitive.microsofttranslator.com/breaksentence?api-version=3.0
```

Parámetros de solicitud

Los parámetros de solicitud que se pasaron en la cadena de consulta son:

PARÁMETRO DE CONSULTA	DESCRIPCIÓN
api-version	*parámetro de consulta obligatorio*. Versión de la API que el cliente solicitó. El valor debe ser `3.0`.
language	*Parámetro de consulta opcional*. Etiqueta de idioma que identifica el idioma del texto de entrada. Si no se especifica un código, se aplicará la detección automática del idioma.
script	*Parámetro de consulta opcional*. Etiqueta de script que identifica el script que usa el texto de entrada. Si no se especifica un script, se asumirá el script predeterminado del idioma.

Los encabezados de solicitud incluyen lo siguiente:

ENCABEZADOS	DESCRIPCIÓN
Encabezados de autenticación	<i>Encabezado de solicitud obligatorio.</i> Consulte las opciones disponibles para la autenticación .
Content-Type	*Encabezado de solicitud obligatorio*. Especifica el tipo de contenido de la carga. Los valores posibles son: `application/json`.
Content-Length	*Encabezado de solicitud obligatorio*. Longitud del cuerpo de la solicitud.
X-ClientTraceId	*Opcional*. GUID generado por el cliente para identificar de forma única la solicitud. Puede omitir este encabezado si incluye el id. de seguimiento en la cadena de la consulta mediante un parámetro de consulta denominado `ClientTraceId`.

Cuerpo de la solicitud

El cuerpo de la solicitud es una matriz JSON. Cada elemento de la matriz es un objeto JSON con una propiedad de cadena denominada `Text`. Los límites de la oración se calculan en función del valor de la propiedad `Text`. Un

cuerpo de la solicitud de muestra con una sola pieza de texto tiene este aspecto:

```
[
  { "Text": "How are you? I am fine. What did you do today?" }
]
```

Se aplican las siguientes limitaciones:

- La matriz puede tener como máximo 100 elementos.
- El valor de texto de un elemento de la matriz no puede superar los 10 000 caracteres, incluyendo los espacios.
- El texto completo incluido en la solicitud no puede superar los 50 000 caracteres, incluidos los espacios.
- Si se especifica el parámetro de consulta `language`, todos los elementos de la matriz deberán estar en el mismo idioma. De lo contrario, la autodetección de idioma se aplica a cada elemento de la matriz de forma independiente.

Response body

Una respuesta correcta es una matriz JSON con un resultado para cada cadena en la matriz de entrada. Un objeto del resultado incluye las siguientes propiedades:

- `sentLen`: es una matriz de enteros que representan las longitudes de las oraciones en el elemento de texto. La longitud de la matriz es el número de oraciones y los valores son la longitud de cada oración.
- `detectedLanguage`: objeto que describe el idioma detectado mediante las siguientes propiedades:
 - `language`: Código del idioma detectado.
 - `score`: valor flotante que indica la confianza en el resultado. La puntuación varía entre cero y uno, y una puntuación baja indica una confianza baja.

Tenga en cuenta que la propiedad `detectedLanguage` solo está presente en el objeto de resultado cuando se solicita la detección automática del idioma.

Un ejemplo de respuesta JSON es:

```
[
  {
    "sentLen": [ 13, 11, 22 ]
    "detectedLanguage": {
      "language": "en",
      "score": 401
    },
  }
]
```

Encabezados de respuesta

ENCABEZADOS	DESCRIPCIÓN
X-RequestId	Valor generado por el servicio para identificar la solicitud. Se usa para solucionar problemas.

Códigos de estado de respuesta

A continuación se indican los códigos de estado HTTP posibles que devuelve una solicitud.

CÓDIGO DE ESTADO	DESCRIPCIÓN
200	Correcto.
400	Uno de los parámetros de consulta falta o no es válido. Corrija los parámetros de la solicitud antes de volver a intentarlo.
401	No pudo autenticarse la solicitud. Compruebe que las credenciales que se especificaron sean correctas.
403	La solicitud no está autenticada. Compruebe los detalles del mensaje de error. Esto a menudo indica que todas las traducciones gratuitas que proporciona la suscripción de prueba se han agotado.
429	El servidor rechazó la solicitud porque el cliente superó los límites de solicitudes.
500	Se ha producido un error inesperado. Si el error continúa, notifíquelo con: fecha y hora del error, identificador de la solicitud del encabezado de respuesta `X-RequestId` e identificador de cliente del encabezado de solicitud `X-ClientTraceId`.
503	Servidor no disponible temporalmente. Vuelva a intentarlo. Si el error continúa, notifíquelo con: fecha y hora del error, identificador de la solicitud del encabezado de respuesta `X-RequestId` e identificador de cliente del encabezado de solicitud `X-ClientTraceId`.

Si se produce un error, la solicitud también devolverá una respuesta de error JSON. El código de error es un número de 6 dígitos que combina el código de estado HTTP de 3 dígitos y otro número de 3 dígitos que ayuda a categorizar aún más el error. Códigos de error comunes que pueden encontrarse en la [página de referencia de Translator Text API v3](#).

Ejemplos

En el siguiente ejemplo se muestra cómo obtener límites de oraciones para una sola oración. El servicio se encarga de detectar automáticamente el idioma de la frase.

```
curl -X POST "https://api.cognitive.microsofttranslator.com/breaksentence?api-version=3.0" -H "Ocp-Apim-Subscription-Key: <client-secret>" -H "Content-Type: application/json" -d "[{'Text': 'How are you? I am fine. What did you do today?'}]"
```


Translator Text API 3.0: Búsqueda en diccionario

13/01/2020 • 11 minutes to read • [Edit Online](#)

Proporciona traducciones alternativas para una palabra y un pequeño número de frases hechas. Cada traducción tiene funciones de sintaxis y una lista de traducciones inversas. Las traducciones inversas permiten al usuario comprender la traducción en contexto. La operación [Ejemplo de diccionario](#) le permite profundizar más para ver ejemplos de usos de cada par de traducción.

URL de la solicitud

Envíe una solicitud `POST` a:

```
https://api.cognitive.microsofttranslator.com/dictionary/lookup?api-version=3.0
```

Parámetros de solicitud

Los parámetros de solicitud que se pasaron en la cadena de consulta son:

PARÁMETRO DE CONSULTA	DESCRIPCIÓN
api-version	*Parámetro obligatorio*. Versión de la API que el cliente solicitó. El valor debe ser `3.0`.
De	*Parámetro obligatorio*. Especifica el idioma del texto de entrada. El idioma de origen debe ser uno de los [idiomas admitidos] (/v3-0-languages.md) que están incluidos en el ámbito `dictionary`.
to	*Parámetro obligatorio*. Especifica el idioma del texto de salida. El idioma de origen debe ser uno de los [idiomas admitidos] (/v3-0-languages.md) que están incluidos en el ámbito `dictionary`.

Los encabezados de solicitud incluyen lo siguiente:

ENCABEZADOS	DESCRIPCIÓN
Encabezados de autenticación	<i>Encabezado de solicitud obligatorio.</i> Consulte las opciones disponibles para la autenticación .
Content-Type	*Encabezado de solicitud obligatorio*. Especifica el tipo de contenido de la carga. Los valores posibles son: `application/json`.
Content-Length	*Encabezado de solicitud obligatorio*. Longitud del cuerpo de la solicitud.
X-ClientTraceId	*Opcional*. GUID generado por el cliente para identificar de forma única la solicitud. Puede omitir este encabezado si incluye el id. de seguimiento en la cadena de la consulta mediante un parámetro de consulta denominado `ClientTraceId`.

Cuerpo de la solicitud

El cuerpo de la solicitud es una matriz JSON. Cada elemento de la matriz es un objeto JSON con una propiedad de cadena denominada `Text`, que representa el término que se va a buscar.

```
[
  {"Text": "fly"}
]
```

Se aplican las siguientes limitaciones:

- La matriz puede tener como máximo 10 elementos.
- El valor de texto de un elemento de la matriz no puede superar los 100 caracteres, incluyendo los espacios.

Response body

Una respuesta correcta es una matriz JSON con un resultado para cada cadena en la matriz de entrada. Un objeto del resultado incluye las siguientes propiedades:

- `normalizedSource`: cadena que proporciona la forma normalizada del término de origen. Por ejemplo, si la solicitud es "JOHN", la forma normalizada será "john". El contenido de este campo se convierte en la entrada de los [ejemplos de búsqueda](#).
- `displaySource`: es una cadena que proporciona el término de origen de la forma más adecuada para mostrársela al usuario final. Por ejemplo, si la entrada es "JOHN", el formulario de visualización reflejará la ortografía habitual del nombre: "John".
- `translations`: es una lista de traducciones del término de origen. Cada elemento de la lista es un objeto que consta de las siguientes propiedades:
 - `normalizedTarget`: es una cadena que proporciona la forma normalizada del término en el idioma de destino. Este valor debe usarse como entrada para [buscar ejemplos](#).
 - `displayTarget`: es una cadena que proporciona el término en el idioma de destino de una forma más adecuada para mostrársela al usuario final. Generalmente, esto solo se diferencia de `normalizedTarget` en lo que respecta al uso de las mayúsculas. Por ejemplo, un nombre propio como "Juan" tendrá `normalizedTarget = "juan"` y `displayTarget = "Juan"`.
 - `posTag`: es una cadena que asocia este término con una etiqueta de categoría gramatical.

NOMBRE DE ETIQUETA	DESCRIPCIÓN
ADJ	Adjetivos
ADV	Adverbios
CONJ	Conjunciones
DET	Determinantes
MODAL	Verbos
SUSTANTIVO	Sustantivos
PREP	Preposiciones

NOMBRE DE ETIQUETA	DESCRIPCIÓN
PRON	Pronombres
VERBO	Verbos
OTHER	Otros

Como nota de implementación, estas etiquetas se determinaron como funciones de sintaxis mediante el etiquetado de la parte en inglés, y usando la etiqueta más frecuente para cada par de origen o destino. Por lo tanto, si las personas traducen con frecuencia una palabra en español a una etiqueta de función de sintaxis que está en inglés, las etiquetas pueden terminar siendo incorrectas (con respecto a la palabra en español).

- `confidence`: es un valor comprendido entre 0,0 y 1,0 que representa la "confianza" (o más exactamente la "probabilidad en los datos de aprendizaje") del par de traducción. La suma de las puntuaciones de confianza para una palabra de origen puede, o no, sumar 1,0.
- `prefixWord`: es una cadena que indica que la palabra que se va mostrar funciona como prefijo de la traducción. En la actualidad, este es el determinante de género de los sustantivos, que se usa en idiomas que tienen determinantes de género. Por ejemplo, el prefijo de la palabra en español "mosca" es "la", ya que "mosca" es un nombre femenino en español. Esto solo depende de la traducción, y no del origen. Si no hay prefijo, la cadena quedará vacía.
- `backTranslations`: es una lista de "traducciones inversas" del destino. Por ejemplo, las palabras de origen que se pueden traducir al idioma de destino. La lista garantiza que se encontrará la palabra de origen que se solicitó (por ejemplo, si la palabra de origen que se busca es "fly", entonces se garantiza que "fly" estará en la lista `backTranslations`). Sin embargo, no se garantiza que esté en la primera posición, y a menudo no lo estará. Cada elemento de la lista `backTranslations` es un objeto que consta de las siguientes propiedades:
 - `normalizedText`: es una cadena que proporciona la forma normalizada del término de origen, que es a su vez una traducción inversa del destino. Este valor debe usarse como entrada para [buscar ejemplos](#).
 - `displayText`: es una cadena que proporciona el término de origen, que es una traducción inversa del destino, que se muestra de la forma más adecuada para el usuario final.
 - `numExamples`: es un entero que representa el número de ejemplos de este par de traducción que están disponibles. Los ejemplos reales deben obtenerse con una llamada independiente a la opción [buscar ejemplos](#). El número está diseñado principalmente para facilitar la presentación en un diseño UX. Por ejemplo, una interfaz de usuario puede agregar un hipervínculo a la traducción inversa si el número de ejemplos es mayor que cero, y si se muestra la traducción inversa como texto sin formato si es que no hay ejemplos. Tenga en cuenta que el número real de ejemplos que devuelve una llamada a la opción [buscar ejemplos](#) puede ser menor que `numExamples`, ya que se puede aplicar un filtro adicional sobre la marcha para quitar ejemplos que sean "incorrectos".
 - `frequencyCount`: es un entero que representa la frecuencia de este par de traducción en los datos. El propósito principal de este campo es proporcionar una interfaz de usuario con un medio para ordenar traducciones inversas, y así conseguir que los términos más frecuentes aparezcan en primer lugar.

NOTE

Si el término que está buscando no está en el diccionario, la respuesta es 200 (OK), pero la lista `translations` será una lista vacía.

Ejemplos

En este ejemplo se muestra cómo buscar traducciones alternativas en español del término inglés `fly`.

```
curl -X POST "https://api.cognitive.microsofttranslator.com/dictionary/lookup?api-version=3.0&from=en&to=es"
-H "Ocp-Apim-Subscription-Key: <client-secret>" -H "Content-Type: application/json" -d "[{'Text':'fly'}]"
```

El cuerpo de la respuesta (abreviado para mayor claridad) es:

```
[
  {
    "normalizedSource": "fly",
    "displaySource": "fly",
    "translations": [
      {
        "normalizedTarget": "volar",
        "displayTarget": "volar",
        "posTag": "VERB",
        "confidence": 0.4081,
        "prefixWord": "",
        "backTranslations": [
          { "normalizedText": "fly", "displayText": "fly", "numExamples": 15, "frequencyCount": 4637 },
          { "normalizedText": "flying", "displayText": "flying", "numExamples": 15, "frequencyCount": 1365 },
          { "normalizedText": "blow", "displayText": "blow", "numExamples": 15, "frequencyCount": 503 },
          { "normalizedText": "flight", "displayText": "flight", "numExamples": 15, "frequencyCount": 135 }
        ]
      },
      {
        "normalizedTarget": "mosca",
        "displayTarget": "mosca",
        "posTag": "NOUN",
        "confidence": 0.2668,
        "prefixWord": "",
        "backTranslations": [
          { "normalizedText": "fly", "displayText": "fly", "numExamples": 15, "frequencyCount": 1697 },
          { "normalizedText": "flyweight", "displayText": "flyweight", "numExamples": 0, "frequencyCount": 48 },
          { "normalizedText": "flies", "displayText": "flies", "numExamples": 9, "frequencyCount": 34 }
        ]
      }
    ],
    // ...list abbreviated for documentation clarity
  }
]
```

En este ejemplo se muestra lo que sucede cuando el término que se va a buscar no existe para el par de diccionario válido.

```
curl -X POST "https://api.cognitive.microsofttranslator.com/dictionary/lookup?api-version=3.0&from=en&to=es"
-H "X-ClientTraceId: 875030C7-5380-40B8-8A03-63DACC69C11" -H "Ocp-Apim-Subscription-Key: <client-secret>" -H
"Content-Type: application/json" -d "[{'Text':'fly123456'}]"
```

Dado que el término no se encuentra en el diccionario, el cuerpo de la respuesta incluye la lista vacía

`translations`.

```
[
  {
    "normalizedSource": "fly123456",
    "displaySource": "fly123456",
    "translations": []
  }
]
```

Translator Text API 3.0: Ejemplos de diccionario

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

Proporciona ejemplos que muestran cómo se usan los términos del diccionario en el contexto. Esta operación se usa junto con la opción [Búsqueda de diccionario](#).

URL de la solicitud

Envíe una solicitud `POST` a:

```
https://api.cognitive.microsofttranslator.com/dictionary/examples?api-version=3.0
```

Parámetros de solicitud

Los parámetros de solicitud que se pasaron en la cadena de consulta son:

PARÁMETRO DE CONSULTA	DESCRIPCIÓN
api-version	*Parámetro obligatorio*. Versión de la API que el cliente solicitó. El valor debe ser `3.0`.
De	*Parámetro obligatorio*. Especifica el idioma del texto de entrada. El idioma de origen debe ser uno de los [idiomas admitidos](./v3-0-languages.md) que están incluidos en el ámbito `dictionary`.
to	*Parámetro obligatorio*. Especifica el idioma del texto de salida. El idioma de origen debe ser uno de los [idiomas admitidos](./v3-0-languages.md) que están incluidos en el ámbito `dictionary`.

Los encabezados de solicitud incluyen lo siguiente:

ENCABEZADOS	DESCRIPCIÓN
Encabezados de autenticación	<i>Encabezado de solicitud obligatorio.</i> Consulte las opciones disponibles para la autenticación .
Content-Type	*Encabezado de solicitud obligatorio*. Especifica el tipo de contenido de la carga. Los valores posibles son: `application/json`.
Content-Length	*Encabezado de solicitud obligatorio*. Longitud del cuerpo de la solicitud.
X-ClientTraceId	*Opcional*. GUID generado por el cliente para identificar de forma única la solicitud. Puede omitir este encabezado si incluye el id. de seguimiento en la cadena de la consulta mediante un parámetro de consulta denominado `ClientTraceId`.

Cuerpo de la solicitud

El cuerpo de la solicitud es una matriz JSON. Cada elemento de la matriz es un objeto JSON que consta de las

siguientes propiedades:

- `Text`: cadena que especifica el término que se va a buscar. Este debería ser el valor de un campo `normalizedText` que proceda de las traducciones inversas encontradas cuando se realizó una solicitud de una [búsqueda de diccionario](#) anterior. También puede ser el valor del campo `normalizedSource`.
- `Translation`: cadena que especifica el texto traducido que previamente devolvió la operación [Búsqueda de diccionario](#). Este debería ser el valor del campo `normalizedTarget` en la lista `translations` de la respuesta [Búsqueda de diccionario](#). El servicio devolverá ejemplos para el par de palabras específico "fuente-objetivo".

Ejemplo:

```
[
  { "Text": "fly", "Translation": "volar" }
]
```

Se aplican las siguientes limitaciones:

- La matriz puede tener como máximo 10 elementos.
- El valor de texto de un elemento de la matriz no puede superar los 100 caracteres, incluyendo los espacios.

Response body

Una respuesta correcta es una matriz JSON con un resultado para cada cadena en la matriz de entrada. Un objeto del resultado incluye las siguientes propiedades:

- `normalizedSource`: cadena que proporciona la forma normalizada del término de origen. En general, esto debería ser idéntico al valor del campo `Text` en el índice de la lista de coincidencias del cuerpo de la solicitud.
- `normalizedTarget`: cadena que proporciona la forma normalizada del término de destino. En general, esto debería ser idéntico al valor del campo `Translation` en el índice de la lista de coincidencias del cuerpo de la solicitud.
- `examples`: lista de ejemplos del par (término de origen y de destino). Cada elemento de la lista es un objeto que consta de las siguientes propiedades:
 - `sourcePrefix`: cadena que se va a concatenar *antes* del valor de `sourceTerm` para formar un ejemplo completo. No agregue un carácter de espacio, ya que ya estará allí en el momento apropiado. Este valor puede ser una cadena vacía.
 - `sourceTerm`: cadena equivalente al término real que se está buscando. La cadena se agrega con `sourcePrefix` y `sourceSuffix` para formar el ejemplo completo. Su valor está separado, por lo que se puede marcar en una interfaz de usuario poniéndolo en negrita, por ejemplo.
 - `sourceSuffix`: cadena que se va a concatenar *después* del valor de `sourceTerm` para formar un ejemplo completo. No agregue un carácter de espacio, ya que ya estará allí en el momento apropiado. Este valor puede ser una cadena vacía.
 - `targetPrefix`: cadena similar a `sourcePrefix` que se usa en el destino.
 - `targetTerm`: cadena similar a `sourceTerm` que se usa en el destino.
 - `targetSuffix`: cadena similar a `sourceSuffix` que se usa en el destino.

NOTE

Si no hay ejemplos en el diccionario, la respuesta es 200 (OK), pero la lista `examples` será una lista vacía.

Ejemplos

En este ejemplo se muestra cómo buscar ejemplos de la pareja formada por el término inglés `fly` y su traducción al español `volar`.

```
curl -X POST "https://api.cognitive.microsofttranslator.com/dictionary/examples?api-version=3.0&from=en&to=es" -H "Ocp-Apim-Subscription-Key: <client-secret>" -H "Content-Type: application/json" -d "[{'Text':'fly', 'Translation':'volar'}]"
```

Es el cuerpo de la respuesta (abreviado para mayor claridad):

```
[
  {
    "normalizedSource":"fly",
    "normalizedTarget":"volar",
    "examples":[
      {
        "sourcePrefix":"They need machines to ",
        "sourceTerm":"fly",
        "sourceSuffix":".",
        "targetPrefix":"Necesitan máquinas para ",
        "targetTerm":"volar",
        "targetSuffix":"."
      },
      {
        "sourcePrefix":"That should really ",
        "sourceTerm":"fly",
        "sourceSuffix":".",
        "targetPrefix":"Eso realmente debe ",
        "targetTerm":"volar",
        "targetSuffix":"."
      }
    ],
    //
    // ...list abbreviated for documentation clarity
    //
  ]
}
```


Translator Text API v2.0

13/01/2020 • 63 minutes to read • [Edit Online](#)

IMPORTANT

Esta versión de Translator Text API está en desuso. [Consulte la documentación de la versión 3 de Translator Text API.](#)

La versión 2 de Translator Text API puede integrarse perfectamente en las aplicaciones, sitios web, herramientas u otras soluciones para proporcionar experiencias de usuario con varios idiomas. Se puede utilizar en cualquier plataforma de hardware y con cualquier sistema operativo para realizar la traducción de idiomas y otras operaciones lingüísticas relacionadas, como la detección del idioma de texto y texto a voz, según los estándares del sector. Para obtener más información, consulte [Translator Text API](#).

Introducción

Para acceder a Translator Text API, debe [suscribirse a Microsoft Azure](#).

Authentication

Todas las llamadas a Translator Text API requieren una clave de suscripción para la autenticación. La API admite tres métodos de autenticación:

- Un token de acceso. Use la clave de suscripción para crear un token de acceso mediante la realización de una solicitud POST al servicio de autenticación. Consulte la documentación de servicio de token para obtener más información. Pase el token de acceso al servicio de Translator con el encabezado de `Authorization` o el parámetro de consulta `access_token`. Cada token de acceso tiene una validez de 10 minutos. Obtenga un nuevo token de acceso cada 10 minutos y siga usando el mismo para solicitudes repetidas durante esos 10 minutos.
- Una clave de suscripción usada directamente. Pase la clave de suscripción como valor en el encabezado de `Ocp-Apim-Subscription-Key` junto con la solicitud a Translator Text API. Al usar la clave de suscripción directamente, no es necesario llamar al servicio de autenticación de token para crear un token de acceso.
- Una [suscripción a varios servicios de Azure Cognitive Services](#). Este método le permite usar una única clave secreta para autenticar las solicitudes de varios servicios. Si usa una clave secreta para varios servicios, debe incluir dos encabezados de autenticación con la solicitud. El primer encabezado pasa la clave secreta. El segundo encabezado especifica la región asociada con la suscripción:
 - `Ocp-Apim-Subscription-Key`
 - `Ocp-Apim-Subscription-Region`

La región es obligatoria en la suscripción de varios servicios de Text API. La región que selecciona es la única región que puede usar para la traducción de texto al usar la clave de suscripción a varios servicios. Debe ser la misma región que seleccionó al registrarse para obtener la suscripción a varios servicios en Azure Portal.

Las regiones disponibles son `australiaeast`, `brazilsouth`, `canadacentral`, `centralindia`, `centraluseuap`, `eastasia`, `eastus`, `eastus2`, `japaneast`, `northeurope`, `southcentralus`, `southeastasia`, `uksouth`, `westcentralus`, `westeurope`, `westus` y `westus2`.

La clave de suscripción y el token de acceso son secretos que deben ocultarse.

Control de palabras soeces

Normalmente, el servicio de Translator conserva las palabras soeces que están presentes en el origen. El grado de blasfemia y el contexto que hace que las palabras sean soeces difieren según la cultura. Así pues, el grado de blasfemia en el idioma de destino podría ampliarse o reducirse.

Si quiere evitar palabras soeces en la traducción, incluso si se encuentran presentes en el texto de origen, puede usar la opción de filtrado de palabras soeces en los métodos que la admiten. La opción le permite elegir si quiere ver las palabras soeces eliminadas o marcadas con etiquetas adecuadas, o bien si quiere permitir las en el destino. Los valores aceptados de `ProfanityAction` son `NoAction` (valor predeterminado), `Marked` y `Deleted`.

PROFANITYACTION		EJEMPLO DE ORIGEN (JAPONÉS)	EJEMPLO DE TRADUCCIÓN (ESPAÑOL)
NoAction	Predeterminada. Igual que si no se configura la opción. Las palabras soeces pasarán del origen al destino.	彼はジャッカスです。	Es un idiota.
Marked	Las palabras soeces aparecerán rodeadas por las etiquetas XML <code><profanity></code> y <code></profanity></code> .	彼はジャッカスです。	Es un <code><profanity>idiota</profanity></code> .
Deleted	Las palabras soeces se quitarán de la salida sin reemplazo.	彼はジャッカスです。	Es un.

Contenido excluido de la traducción

Al traducir el contenido con etiquetas, como HTML (`contentType=text/html`), a veces resulta útil para excluir contenido específico de la traducción. Puede utilizar el atributo `class=notranslate` para especificar contenido que debería permanecer en el idioma original. En el ejemplo siguiente, el contenido del primer elemento `div` no se traducirá, mientras que el del segundo elemento `div`, sí.

```
<div class="notranslate">This will not be translated.</div>
<div>This will be translated. </div>
```

GET /Translate

Notas de implementación

Traduce una cadena de texto de un idioma a otro.

El URI de solicitud es `https://api.microsofttranslator.com/V2/Http.svc/Translate`.

Valor devuelto: Una cadena que representa el texto traducido.

Si anteriormente se usaba `AddTranslation` o `AddTranslationArray` para introducir una traducción con una clasificación de 5 o superior para la misma oración de origen, `Translate` devuelve solo la principal opción que está disponible para el sistema. "Misma oración de origen" significa exactamente lo mismo (coincidencia 100 %), excepto en el caso del uso de mayúsculas y minúsculas, espacio en blanco, valores de etiquetas y puntuación al final de una oración. Si no se almacena ninguna clasificación con una puntuación de 5 o superior, el resultado devuelto será la traducción automática de Microsoft Translator.

Clase de respuesta (estado 200)

string

Tipo de contenido de la respuesta: application/xml

Parámetros

PARÁMETRO	VALOR	DESCRIPCIÓN	TIPO DE PARÁMETRO	TIPO DE DATOS
appid	(vacío)	Necesario. Si se usa el encabezado <code>Authorization</code> o <code>Ocp-Apim-Subscription-Key</code> , deje vacío el campo <code>appid</code> . De lo contrario, incluya una cadena que contenga <code>"Bearer" + " " + "access_token"</code> .	query	string
text	(vacío)	Necesario. Una cadena que representa el texto a traducir. El texto no puede contener más de 10 000 caracteres.	query	string
De	(vacío)	Opcional. Una cadena que representa el código de idioma del texto que se traduce. Por ejemplo, en para inglés.	query	string
to	(vacío)	Necesario. Una cadena que representa el código del idioma al que se va a traducir el texto.	query	string
contentType	(vacío)	Opcional. El formato del texto que se va a traducir. Los formatos admitidos son <code>text/plain</code> (valor predeterminado) y <code>text/html</code> . Cualquier elemento HTML debe ser un elemento completo y con formato correcto.	query	string
category	(vacío)	Opcional. Una cadena que contiene la categoría (dominio) de la traducción. El valor predeterminado es <code>general</code> .	query	string

PARÁMETRO	VALOR	DESCRIPCIÓN	TIPO DE PARÁMETRO	TIPO DE DATOS
Authorization	(vacío)	Se requiere tanto si el campo <code>appid</code> como el encabezado <code>Ocp-Apim-Subscription-Key</code> se dejan vacíos. Token de autorización: <code>"Bearer" + " " + "access_token"</code> .	encabezado	string
Ocp-Apim-Subscription-Key	(vacío)	Se requiere tanto si el campo <code>appid</code> como el encabezado <code>Authorization</code> se dejan vacíos.	encabezado	string

Mensajes de respuesta

CÓDIGO DE ESTADO HTTP	MOTIVO
400	Solicitud incorrecta. Compruebe los parámetros de entrada y la respuesta del error detallado.
401	Credenciales no válidas.
500	Error de servidor. Si el error continúa, infórmenos. Facilítenos la fecha y hora aproximadas de la solicitud, junto con el Id. de solicitud incluido en el encabezado de respuesta <code>X-MS-Trans-Info</code> .
503	Servicio no disponible temporalmente. Vuelva a intentarlo e infórmenos si el error continúa.

POST /TranslateArray

Notas de implementación

Recupera traducciones de varios textos de origen.

El URI de solicitud es `https://api.microsofttranslator.com/V2/Http.svc/TranslateArray` .

Este es el formato del cuerpo de la solicitud:

```

<TranslateArrayRequest>
  <AppId />
  <From>language-code</From>
  <Options>
    <Category xmlns="http://schemas.datacontract.org/2004/07/Microsoft.MT.Web.Service.V2" >string-
value</Category>
    <ContentType
xmlns="http://schemas.datacontract.org/2004/07/Microsoft.MT.Web.Service.V2">text/plain</ContentType>
    <ReservedFlags xmlns="http://schemas.datacontract.org/2004/07/Microsoft.MT.Web.Service.V2" />
    <State xmlns="http://schemas.datacontract.org/2004/07/Microsoft.MT.Web.Service.V2" >int-value</State>
    <Uri xmlns="http://schemas.datacontract.org/2004/07/Microsoft.MT.Web.Service.V2" >string-value</Uri>
    <User xmlns="http://schemas.datacontract.org/2004/07/Microsoft.MT.Web.Service.V2" >string-value</User>
  </Options>
  <Texts>
    <string xmlns="http://schemas.microsoft.com/2003/10/Serialization/Arrays">string-value</string>
    <string xmlns="http://schemas.microsoft.com/2003/10/Serialization/Arrays">string-value</string>
  </Texts>
  <To>language-code</To>
</TranslateArrayRequest>

```

Estos elementos están en `TranslateArrayRequest` :

- `AppId` : Necesario. Si se usa el encabezado `Authorization` o `Ocp-Apim-Subscription-Key` , deje vacío el campo `AppId` . De lo contrario, incluya una cadena que contenga `"Bearer" + " " + "access_token"` .
- `From` : Opcional. Una cadena que representa el código de idioma del texto que se traduce. Si se deja vacío este campo, la respuesta incluirá el resultado de la detección automática de idioma.
- `Options` : Opcional. Un objeto `Options` que contiene los valores siguientes. Son todos opcionales y toman como valor predeterminado las opciones de configuración más comunes. Los elementos especificados deben incluirse en orden alfabético.
 - `Category` : una cadena que contiene la categoría (dominio) de la traducción. El valor predeterminado es `general` .
 - `ContentType` : El formato del texto que se va a traducir. Los formatos admitidos son `text/plain` (valor predeterminado), `text/xml` y `text/html` . Cualquier elemento HTML debe ser un elemento completo y con formato correcto.
 - `ProfanityAction` : especifica cómo se tratan las blasfemias según se explicó anteriormente. Los valores aceptados son `NoAction` (valor predeterminado), `Marked` y `Deleted` .
 - `State` : estado del usuario para ayudar a poner en correlación la solicitud y la respuesta. Se devolverá el mismo contenido en la respuesta.
 - `Uri` : Filtrar los resultados por este URI. Valor predeterminado: `all` .
 - `User` : Filtrar los resultados por este usuario. Valor predeterminado: `all` .
- `Texts` : Necesario. Una matriz que contiene el texto para la traducción. Todas las cadenas deben estar en el mismo idioma. El total de todo el texto que se debe traducir no puede superar los 10 000 caracteres. El número máximo de elementos de la matriz es 2000.
- `To` : Necesario. Una cadena que representa el código del idioma al que se va a traducir el texto.

Puede omitir los elementos opcionales. Los elementos que son elementos secundarios directos de `TranslateArrayRequest` deben aparecer en orden alfabético.

El método `TranslateArray` acepta `application/xml` o `text/xml` para `Content-Type` .

Valor devuelto: Una matriz `TranslateArrayResponse` . Cada `TranslateArrayResponse` tiene estos elementos:

- `Error` : indica un error si se produce. En caso contrario, se establece en null.
- `OriginalSentenceLengths` : una matriz de enteros que indica la longitud de cada oración del texto de origen. La longitud de la matriz indica el número de oraciones.

- `TranslatedText` : El texto traducido.
- `TranslatedSentenceLengths` : una matriz de enteros que indica la longitud de cada oración del texto traducido. La longitud de la matriz indica el número de oraciones.
- `State` : estado del usuario para ayudar a poner en correlación la solicitud y la respuesta. Devuelve el mismo contenido que la solicitud.

Este es el formato del cuerpo de la respuesta:

```
<ArrayOfTranslateArrayResponse xmlns="http://schemas.datacontract.org/2004/07/Microsoft.MT.Web.Service.V2"
  xmlns:i="https://www.w3.org/2001/XMLSchema-instance">
  <TranslateArrayResponse>
    <From>language-code</From>
    <OriginalTextSentenceLengths xmlns:a="http://schemas.microsoft.com/2003/10/Serialization/Arrays">
      <a:int>int-value</a:int>
    </OriginalTextSentenceLengths>
    <State/>
    <TranslatedText>string-value</TranslatedText>
    <TranslatedTextSentenceLengths xmlns:a="http://schemas.microsoft.com/2003/10/Serialization/Arrays">
      <a:int>int-value</a:int>
    </TranslatedTextSentenceLengths>
  </TranslateArrayResponse>
</ArrayOfTranslateArrayResponse>
```

Clase de respuesta (estado 200)

Una respuesta correcta incluye una matriz de matrices `TranslateArrayResponse` en el formato descrito anteriormente.

string

Tipo de contenido de la respuesta: application/xml

Parámetros

PARÁMETRO	VALOR	DESCRIPCIÓN	TIPO DE PARÁMETRO	TIPO DE DATOS
Authorization	(vacío)	Se requiere tanto si el campo <code>appid</code> como el encabezado <code>Ocp-Apim-Subscription-Key</code> se dejan vacíos. Token de autorización: <code>"Bearer" + " " + "access_token"</code> .	encabezado	string
Ocp-Apim-Subscription-Key	(vacío)	Se requiere tanto si el campo <code>appid</code> como el encabezado <code>Authorization</code> se dejan vacíos.	encabezado	string

Mensajes de respuesta

CÓDIGO DE ESTADO HTTP	MOTIVO
-----------------------	--------

CÓDIGO DE ESTADO HTTP	MOTIVO
400	<p>Solicitud incorrecta. Compruebe los parámetros de entrada y la respuesta del error detallado. Estos son algunos de los errores comunes:</p> <ul style="list-style-type: none"> • El elemento de matriz no puede estar vacío. • Categoría no válida. • El idioma de origen no es válido. • El idioma de destino no es válido. • La solicitud contiene demasiados elementos. • El idioma de origen no se admite. • El idioma de destino no se admite. • La solicitud de traducción tiene demasiados datos. • HTML no tiene el formato correcto. • Se pasaron demasiadas cadenas en la solicitud de traducción.
401	Credenciales no válidas.
500	<p>Error de servidor. Si el error continúa, infórmenos. Facilítenos la fecha y hora aproximadas de la solicitud, junto con el Id. de solicitud incluido en el encabezado de respuesta</p> <p><code>X-MS-Trans-Info</code>.</p>
503	Servicio no disponible temporalmente. Vuelva a intentarlo e infórmenos si el error continúa.

POST /GetLanguageNames

Notas de implementación

Recupera los nombres descriptivos de los idiomas que se pasaron como parámetro `languageCodes`, localizados en el idioma `locale` que se pasó.

El URI de solicitud es `https://api.microsofttranslator.com/V2/Http.svc/GetLanguageNames`.

El cuerpo de la solicitud incluye una matriz de cadenas que representa los códigos de idioma ISO 639-1 para los que se van a recuperar los nombres descriptivos. Este es un ejemplo:

```
<ArrayOfstring xmlns:i="https://www.w3.org/2001/XMLSchema-instance"
xmlns="http://schemas.microsoft.com/2003/10/Serialization/Arrays">
  <string>zh</string>
  <string>en</string>
</ArrayOfstring>
```

Valor devuelto: una matriz de cadenas que contiene los nombres de idiomas que admite el servicio de Translator, localizados en el idioma solicitado.

Clase de respuesta (estado 200)

Una matriz de cadenas que contiene los nombres de idiomas que admite el servicio de Translator, localizados en el idioma solicitado.

string

Tipo de contenido de la respuesta: application/xml

Parámetros

PARÁMETRO	VALOR	DESCRIPCIÓN	TIPO DE PARÁMETRO	TIPO DE DATOS
appid	(vacío)	Necesario. Si se usa el encabezado <code>Authorization</code> o <code>Ocp-Apim-Subscription-Key</code> , deje vacío el campo <code>appid</code> . De lo contrario, incluya una cadena que contenga <code>"Bearer" + " " + "access_token"</code> .	query	string
locale	(vacío)	Necesario. Una cadena que representa uno de los siguientes elementos, que se usa para localizar los nombres de idiomas: <ul style="list-style-type: none"> La combinación de un código de cultura en minúsculas de dos letras ISO 639 asociado a un idioma y un código de subcultura en mayúsculas de dos letras ISO 3166. Un código de cultura en minúsculas ISO 639 por sí mismo. 	query	string
Authorization	(vacío)	Se requiere tanto si el campo <code>appid</code> como el encabezado <code>Ocp-Apim-Subscription-Key</code> se dejan vacíos. Token de autorización: <code>"Bearer" + " " + "access_token"</code> .	encabezado	string
Ocp-Apim-Subscription-Key	(vacío)	Se requiere tanto si el campo <code>appid</code> como el encabezado <code>Authorization</code> se dejan vacíos.	encabezado	string

Mensajes de respuesta

CÓDIGO DE ESTADO HTTP	MOTIVO
400	Solicitud incorrecta. Compruebe los parámetros de entrada y la respuesta del error detallado.
401	Credenciales no válidas.
500	Error de servidor. Si el error continúa, infórmenos. Facilítenos la fecha y hora aproximadas de la solicitud, junto con el Id. de solicitud incluido en el encabezado de respuesta <code>X-MS-Trans-Info</code> .
503	Servicio no disponible temporalmente. Vuelva a intentarlo e infórmenos si el error continúa.

GET /GetLanguagesForTranslate

Notas de implementación

Obtiene una lista de códigos de idioma que representan los idiomas que admite el servicio de traducción.

`Translate` y `TranslateArray` puede traducir entre cualquier par de estos idiomas.

El URI de solicitud es `https://api.microsofttranslator.com/V2/Http.svc/GetLanguagesForTranslate`.

Valor devuelto: una matriz de cadenas que contiene los códigos de idioma que admite el servicio de Translator.

Clase de respuesta (estado 200)

Una matriz de cadenas que contiene los códigos de idioma que admite el servicio de Translator.

string

Tipo de contenido de la respuesta: application/xml

Parámetros

PARÁMETRO	VALOR	DESCRIPCIÓN	TIPO DE PARÁMETRO	TIPO DE DATOS
appid	(vacío)	Necesario. Si se usa el encabezado <code>Authorization</code> o <code>Ocp-Apim-Subscription-Key</code> , deje vacío el campo <code>appid</code> . De lo contrario, incluya una cadena que contenga <code>"Bearer" + " " + "access_token"</code> .	query	string
Authorization	(vacío)	Se requiere tanto si el campo <code>appid</code> como el encabezado <code>Ocp-Apim-Subscription-Key</code> se dejan vacíos. Token de autorización: <code>"Bearer" + " " + "access_token"</code> .	encabezado	string

PARÁMETRO	VALOR	DESCRIPCIÓN	TIPO DE PARÁMETRO	TIPO DE DATOS
Ocp-Apim-Subscription-Key	(vacío)	Se requiere tanto si el campo <code>appid</code> como el encabezado <code>Authorization</code> se dejan vacíos.	encabezado	string

Mensajes de respuesta

CÓDIGO DE ESTADO HTTP	MOTIVO
400	Solicitud incorrecta. Compruebe los parámetros de entrada y la respuesta del error detallado.
401	Credenciales no válidas.
500	Error de servidor. Si el error continúa, infórmenos. Facilítenos la fecha y hora aproximadas de la solicitud, junto con el Id. de solicitud incluido en el encabezado de respuesta <code>X-MS-Trans-Info</code> .
503	Servicio no disponible temporalmente. Vuelva a intentarlo e infórmenos si el error continúa.

GET /GetLanguagesForSpeak

Notas de implementación

Recupera los idiomas disponibles para la síntesis de voz.

El URI de solicitud es `https://api.microsofttranslator.com/V2/Http.svc/GetLanguagesForSpeak`.

Valor devuelto: una matriz de cadenas que contiene los códigos de idioma que admite la síntesis de voz del servicio de Translator.

Clase de respuesta (estado 200)

Una matriz de cadenas que contiene los códigos de idioma que admite la síntesis de voz del servicio de Translator.

string

Tipo de contenido de la respuesta: application/xml

Parámetros

PARÁMETRO	VALOR	DESCRIPCIÓN	TIPO DE PARÁMETRO	TIPO DE DATOS
appid	(vacío)	Necesario. Si se usa el encabezado <code>Authorization</code> o <code>Ocp-Apim-Subscription-Key</code> , deje vacío el campo <code>appid</code> . De lo contrario, incluya una cadena que contenga <code>"Bearer" + " " + "access_token"</code> .	query	string

PARÁMETRO	VALOR	DESCRIPCIÓN	TIPO DE PARÁMETRO	TIPO DE DATOS
Authorization	(vacío)	Se requiere tanto si el campo <code>appid</code> como el encabezado <code>Ocp-Apim-Subscription-Key</code> se dejan vacíos. Token de autorización: <code>"Bearer" + " " + "access_token"</code> .	encabezado	string
Ocp-Apim-Subscription-Key	(vacío)	Se requiere tanto si el campo <code>appid</code> como el encabezado <code>Authorization</code> se dejan vacíos.	encabezado	string

Mensajes de respuesta

CÓDIGO DE ESTADO HTTP	MOTIVO
400	Solicitud incorrecta. Compruebe los parámetros de entrada y la respuesta del error detallado.
401	Credenciales no válidas.
500	Error de servidor. Si el error continúa, infórmenos. Facilítenos la fecha y hora aproximadas de la solicitud, junto con el Id. de solicitud incluido en el encabezado de respuesta <code>X-MS-Trans-Info</code> .
503	Servicio no disponible temporalmente. Vuelva a intentarlo e infórmenos si el error continúa.

GET /Speak

Notas de implementación

Devuelve una secuencia con formato WAV o MP3 del texto que se ha pasado reproducido en el idioma deseado.

El URI de solicitud es `https://api.microsofttranslator.com/V2/Http.svc/Speak` .

Valor devuelto: una secuencia con formato WAV o MP3 del texto que se ha pasado reproducido en el idioma deseado.

Clase de respuesta (estado 200)

binary

Tipo de contenido de la respuesta: application/xml

Parámetros

PARÁMETRO	VALOR	DESCRIPCIÓN	TIPO DE PARÁMETRO	TIPO DE DATOS
-----------	-------	-------------	-------------------	---------------

PARÁMETRO	VALOR	DESCRIPCIÓN	TIPO DE PARÁMETRO	TIPO DE DATOS
appid	(vacío)	Necesario. Si se usa el encabezado <code>Authorization</code> o <code>Ocp-Apim-Subscription-Key</code> , deje vacío el campo <code>appid</code> . De lo contrario, incluya una cadena que contenga <code>"Bearer" + " " + "access_token"</code> .	query	string
text	(vacío)	Necesario. Una cadena que contiene una o varias oraciones que se van a reproducir para la secuencia en el idioma especificado. El texto no debe tener más de 2000 caracteres.	query	string
language	(vacío)	Necesario. Una cadena que representa el código de idioma admitido del idioma en el que se va a reproducir el texto. El código debe ser uno de los códigos devueltos por el método <code>GetLanguagesForSpeak</code> .	query	string
format	(vacío)	Opcional. Una cadena que especifica el Id. de content-type. Actualmente, <code>audio/wav</code> y <code>audio/mp3</code> están disponibles. El valor predeterminado es <code>audio/wav</code> .	query	string

PARÁMETRO	VALOR	DESCRIPCIÓN	TIPO DE PARÁMETRO	TIPO DE DATOS
options	(vacío)	<p>Opcional. Una cadena que especifica propiedades de la voz sintetizada:</p> <ul style="list-style-type: none"> <code>MaxQuality</code> y <code>MinSize</code> especifican la calidad de la señal de audio. <code>MaxQuality</code> proporciona la máxima calidad. <code>MinSize</code> proporciona el tamaño de archivo más pequeño. El valor predeterminado es <code>MinSize</code>. <code>female</code> y <code>male</code> especifican el género de voz deseado. El valor predeterminado es <code>female</code>. Use la barra vertical (<code> </code>) para incluir varias opciones. Por ejemplo, <code>MaxQuality Male</code>. 	query	string
Authorization	(vacío)	<p>Se requiere tanto si el campo <code>appid</code> como el encabezado <code>Ocp-Apim-Subscription-Key</code> se dejan vacíos. Token de autorización:</p> <pre>"Bearer" + " " + "access_token"</pre>	encabezado	string
Ocp-Apim-Subscription-Key	(vacío)	<p>Se requiere tanto si el campo <code>appid</code> como el encabezado <code>Authorization</code> se dejan vacíos.</p>	encabezado	string

Mensajes de respuesta

CÓDIGO DE ESTADO HTTP	MOTIVO
400	Solicitud incorrecta. Compruebe los parámetros de entrada y la respuesta del error detallado.
401	Credenciales no válidas.
500	Error de servidor. Si el error continúa, infórmenos. Facilítenos la fecha y hora aproximadas de la solicitud, junto con el Id. de solicitud incluido en el encabezado de respuesta <code>X-MS-Trans-Info</code> .
503	Servicio no disponible temporalmente. Vuelva a intentarlo e infórmenos si el error continúa.

GET /Detect

Notas de implementación

Identifica el idioma de una sección de texto.

El URI de solicitud es `https://api.microsofttranslator.com/V2/Http.svc/Detect` .

Valor devuelto: una cadena que contiene un código de idioma de dos caracteres para el texto.

Clase de respuesta (estado 200)

string

Tipo de contenido de la respuesta: application/xml

Parámetros

PARÁMETRO	VALOR	DESCRIPCIÓN	TIPO DE PARÁMETRO	TIPO DE DATOS
appid	(vacío)	Necesario. Si se usa el encabezado <code>Authorization</code> o <code>Ocp-Apim-Subscription-Key</code> , deje vacío el campo <code>appid</code> . De lo contrario, incluya una cadena que contenga <code>"Bearer" + " " + "access_token"</code> .	query	string
text	(vacío)	Necesario. Una cadena que contiene el texto cuyo idioma se va a identificar. El texto no debe tener más de 10 000 caracteres.	query	string

PARÁMETRO	VALOR	DESCRIPCIÓN	TIPO DE PARÁMETRO	TIPO DE DATOS
Authorization	(vacío)	Se requiere tanto si el campo <code>appid</code> como el encabezado <code>Ocp-Apim-Subscription-Key</code> se dejan vacíos. Token de autorización: <code>"Bearer" + " " + "access_token"</code> .	encabezado	string
Ocp-Apim-Subscription-Key	(vacío)	Se requiere tanto si el campo <code>appid</code> como el encabezado <code>Authorization</code> se dejan vacíos.	encabezado	string

Mensajes de respuesta

CÓDIGO DE ESTADO HTTP	MOTIVO
400	Solicitud incorrecta. Compruebe los parámetros de entrada y la respuesta del error detallado.
401	Credenciales no válidas.
500	Error de servidor. Si el error continúa, infórmenos. Facilítenos la fecha y hora aproximadas de la solicitud, junto con el Id. de solicitud incluido en el encabezado de respuesta <code>X-MS-Trans-Info</code> .
503	Servicio no disponible temporalmente. Vuelva a intentarlo e infórmenos si el error continúa.

POST /DetectArray

Notas de implementación

Identifica los idiomas en una matriz de cadenas. Detecta de forma independiente el idioma de cada elemento de matriz individual y devuelve un resultado para cada fila de la matriz.

El URI de solicitud es `https://api.microsofttranslator.com/V2/Http.svc/DetectArray` .

Este es el formato del cuerpo de la solicitud:

```
<ArrayOfstring xmlns="http://schemas.microsoft.com/2003/10/Serialization/Arrays">
  <string>string-value-1</string>
  <string>string-value-2</string>
</ArrayOfstring>
```

El texto no puede tener más de 10 000 caracteres.

Valor devuelto: una matriz de cadenas que contiene un código de idioma de dos caracteres para cada fila de la matriz de entrada.

Este es el formato del cuerpo de la respuesta:

```
<ArrayOfstring xmlns="http://schemas.microsoft.com/2003/10/Serialization/Arrays"
xmlns:i="https://www.w3.org/2001/XMLSchema-instance">
  <string>language-code-1</string>
  <string>language-code-2</string>
</ArrayOfstring>
```

Clase de respuesta (estado 200)

`DetectArray` se realizó correctamente. Devuelve una matriz de cadenas que contiene un código de idioma de dos caracteres para cada fila de la matriz de entrada.

string

Tipo de contenido de la respuesta: application/xml

Parámetros

PARÁMETRO	VALOR	DESCRIPCIÓN	TIPO DE PARÁMETRO	TIPO DE DATOS
appid	(vacío)	Necesario. Si se usa el encabezado <code>Authorization</code> o <code>Ocp-Apim-Subscription-Key</code> , deje vacío el campo <code>appid</code> . De lo contrario, incluya una cadena que contenga <code>"Bearer" + " " + "access_token"</code> .	query	string
Authorization	(vacío)	Se requiere tanto si el campo <code>appid</code> como el encabezado <code>Ocp-Apim-Subscription-Key</code> se dejan vacíos. Token de autorización: <code>"Bearer" + " " + "access_token"</code> .	encabezado	string
Ocp-Apim-Subscription-Key	(vacío)	Se requiere tanto si el campo <code>appid</code> como el encabezado <code>Authorization</code> se dejan vacíos.	encabezado	string

Mensajes de respuesta

CÓDIGO DE ESTADO HTTP	MOTIVO
400	Solicitud incorrecta. Compruebe los parámetros de entrada y la respuesta del error detallado.
401	Credenciales no válidas.

CÓDIGO DE ESTADO HTTP	MOTIVO
500	Error de servidor. Si el error continúa, infórmenos. Facilítenos la fecha y hora aproximadas de la solicitud, junto con el Id. de solicitud incluido en el encabezado de respuesta <code>X-MS-Trans-Info</code> .
503	Servicio no disponible temporalmente. Vuelva a intentarlo e infórmenos si el error continúa.

GET /AddTranslation

Notas de implementación

IMPORTANT

Nota de desuso: Después del 31 de enero de 2018, este método no aceptará nuevos envíos de oraciones. Obtendrá un mensaje de error. Consulte el anuncio acerca de los cambios realizados en Collaborative Translation Framework (CTF).

Agrega una traducción a la memoria de traducción.

El URI de solicitud es `https://api.microsofttranslator.com/V2/Http.svc/AddTranslation`.

Clase de respuesta (estado 200)

string

Tipo de contenido de la respuesta: application: xml

Parámetros

PARÁMETRO	VALOR	DESCRIPCIÓN	TIPO DE PARÁMETRO	TIPO DE DATOS
appid	(vacío)	Necesario. Si se usa el encabezado <code>Authorization</code> o <code>Ocp-Apim-Subscription-Key</code> , deje vacío el campo <code>appid</code> . De lo contrario, incluya una cadena que contenga <code>"Bearer" + " " + "access_token"</code> .	query	string
originalText	(vacío)	Necesario. Una cadena que representa el texto a traducir. La longitud máxima de la cadena es de 1000 caracteres.	query	string

PARÁMETRO	VALOR	DESCRIPCIÓN	TIPO DE PARÁMETRO	TIPO DE DATOS
translatedText	(vacío)	Necesario. Una cadena que contiene texto traducido en el idioma de destino. La longitud máxima de la cadena es de 2000 caracteres.	query	string
De	(vacío)	Necesario. Una cadena que representa el código de idioma del idioma original del texto. Por ejemplo, en para inglés y de para alemán.	query	string
to	(vacío)	Necesario. Una cadena que representa el código de idioma del idioma al que se va a traducir el texto.	query	string
rating	(vacío)	Opcional. Un entero que representa la clasificación de calidad de la cadena. El valor se encuentra entre -10 y 10. El valor predeterminado es 1.	query	integer
contentType	(vacío)	Opcional. El formato del texto que se va a traducir. Los formatos admitidos son <code>text/plain</code> y <code>text/html</code> . Cualquier elemento HTML debe ser un elemento completo y con formato correcto.	query	string
category	(vacío)	Opcional. Una cadena que contiene la categoría (dominio) de la traducción. El valor predeterminado es <code>general</code> .	query	string
user	(vacío)	Necesario. Una cadena que se usa para realizar un seguimiento del originador del envío.	query	string

PARÁMETRO	VALOR	DESCRIPCIÓN	TIPO DE PARÁMETRO	TIPO DE DATOS
uri	(vacío)	Opcional. Una cadena que contiene la ubicación del contenido de la traducción.	query	string
Authorization	(vacío)	Se requiere tanto si el campo <code>appid</code> como el encabezado <code>Ocp-Apim-Subscription-Key</code> se dejan vacíos. Token de autorización: <code>"Bearer" + " " + "access_token"</code> .	encabezado	string
Ocp-Apim-Subscription-Key	(vacío)	Se requiere tanto si el campo <code>appid</code> como el encabezado <code>Authorization</code> se dejan vacíos.	encabezado	string

Mensajes de respuesta

CÓDIGO DE ESTADO HTTP	MOTIVO
400	Solicitud incorrecta. Compruebe los parámetros de entrada y la respuesta del error detallado.
401	Credenciales no válidas.
410	<code>AddTranslation</code> ya no se admite.
500	Error de servidor. Si el error continúa, infórmenos. Facilítenos la fecha y hora aproximadas de la solicitud, junto con el Id. de solicitud incluido en el encabezado de respuesta <code>X-MS-Trans-Info</code> .
503	Servicio no disponible temporalmente. Vuelva a intentarlo e infórmenos si el error continúa.

POST /AddTranslationArray

Notas de implementación

IMPORTANT

Nota de desuso: Después del 31 de enero de 2018, este método no aceptará nuevos envíos de oraciones. Obtendrá un mensaje de error. Consulte el anuncio acerca de los cambios realizados en Collaborative Translation Framework (CTF).

Agrega una matriz de traducciones a la memoria de traducción. Este método es una versión de la matriz de

`AddTranslation` .

El URI de solicitud es `https://api.microsofttranslator.com/V2/Http.svc/AddTranslationArray` .

Este es el formato del cuerpo de la solicitud:

```
<AddtranslationsRequest>
  <AppId></AppId>
  <From>A string containing the language code of the source language</From>
  <Options>
    <Category xmlns="http://schemas.datacontract.org/2004/07/Microsoft.MT.Web.Service.V2">string-
value</Category>
    <ContentType
xmlns="http://schemas.datacontract.org/2004/07/Microsoft.MT.Web.Service.V2">text/plain</ContentType>
    <User xmlns="http://schemas.datacontract.org/2004/07/Microsoft.MT.Web.Service.V2">string-value</User>
    <Uri xmlns="http://schemas.datacontract.org/2004/07/Microsoft.MT.Web.Service.V2">string-value</Uri>
  </Options>
  <To>A string containing the language code of the target language</To>
  <Translations>
    <Translation xmlns="http://schemas.datacontract.org/2004/07/Microsoft.MT.Web.Service.V2">
      <OriginalText>string-value</OriginalText>
      <Rating>int-value</Rating>
      <Sequence>int-value</Sequence>
      <TranslatedText>string-value</TranslatedText>
    </Translation>
  </Translations>
</AddtranslationsRequest>
```

Estos elementos están en `AddtranslationsRequest` :

- `AppId` : Necesario. Si se usa el encabezado `Authorization` o `Ocp-Apim-Subscription-Key` , deje vacío el campo `AppId` . De lo contrario, incluya una cadena que contenga `"Bearer" + " " + "access_token"` .
- `From` : Necesario. Una cadena que contiene el código de idioma del idioma de origen. Debe ser uno de los idiomas devueltos por el método `GetLanguagesForTranslate` .
- `To` : Necesario. Una cadena que contiene el código de idioma del idioma de destino. Debe ser uno de los idiomas devueltos por el método `GetLanguagesForTranslate` .
- `Translations` : Necesario. Una matriz de traducciones para agregar memoria de traducción. Cada traducción debe contener `OriginalText` , `TranslatedText` y `Rating` . El tamaño máximo de cada `OriginalText` y `TranslatedText` es 1000 caracteres. El total de todos los elementos `OriginalText` y `TranslatedText` no puede ser superior a 10 000 caracteres. El número máximo de elementos de la matriz es 100.
- `options` : Necesario. Un conjunto de opciones, entre las que se incluyen `Category` , `ContentType` , `Uri` y `User` . `User` es obligatorio. `Category` , `ContentType` y `Uri` son opcionales. Los elementos especificados deben incluirse en orden alfabético.

Clase de respuesta (estado 200)

El método `AddTranslationArray` se ha llevado a cabo de forma correcta.

Después del 31 de enero de 2018, no se aceptarán envíos de oraciones. El servicio responderá con el código de error 410.

string

Tipo de contenido de la respuesta: application/xml

Parámetros

PARÁMETRO	VALOR	DESCRIPCIÓN	TIPO DE PARÁMETRO	TIPO DE DATOS
-----------	-------	-------------	-------------------	---------------

PARÁMETRO	VALOR	DESCRIPCIÓN	TIPO DE PARÁMETRO	TIPO DE DATOS
Authorization	(vacío)	Se requiere tanto si el campo <code>appid</code> como el encabezado <code>Ocp-Apim-Subscription-Key</code> se dejan vacíos. Token de autorización: <code>"Bearer" + " " + "access_token"</code> .	encabezado	string
Ocp-Apim-Subscription-Key	(vacío)	Se requiere tanto si el campo <code>appid</code> como el encabezado <code>Authorization</code> se dejan vacíos.	encabezado	string

Mensajes de respuesta

CÓDIGO DE ESTADO HTTP	MOTIVO
400	Solicitud incorrecta. Compruebe los parámetros de entrada y la respuesta del error detallado.
401	Credenciales no válidas.
410	<code>AddTranslation</code> ya no se admite.
500	Error de servidor. Si el error continúa, infórmenos. Facilítenos la fecha y hora aproximadas de la solicitud, junto con el Id. de solicitud incluido en el encabezado de respuesta <code>X-MS-Trans-Info</code> .
503	Servicio no disponible temporalmente. Vuelva a intentarlo e infórmenos si el error continúa.

GET /BreakSentences

Notas de implementación

Divide una sección de texto en oraciones y devuelve una matriz que contiene la longitud de cada oración.

El URI de solicitud es `https://api.microsofttranslator.com/V2/Http.svc/BreakSentences` .

Valor devuelto: una matriz de enteros que representa las longitudes de las oraciones. La longitud de la matriz representa el número de oraciones. Los valores representan la longitud de cada oración.

Clase de respuesta (estado 200)

Una matriz de enteros que representa las longitudes de las oraciones. La longitud de la matriz representa el número de oraciones. Los valores representan la longitud de cada oración.

integer

Tipo de contenido de la respuesta: application/xml

Parámetros

PARÁMETRO	VALOR	DESCRIPCIÓN	TIPO DE PARÁMETRO	TIPO DE DATOS
appid	(vacío)	Necesario. Si se usa el encabezado <code>Authorization</code> o <code>Ocp-Apim-Subscription-Key</code> , deje vacío el campo <code>appid</code> . De lo contrario, incluya una cadena que contenga <code>"Bearer" + " " + "access_token"</code> .	query	string
text	(vacío)	Necesario. Una cadena que representa el texto que se va a dividir en oraciones. El tamaño máximo del texto es 10 000 caracteres.	query	string
language	(vacío)	Necesario. Una cadena que representa el código de idioma del texto de entrada.	query	string
Authorization	(vacío)	Se requiere tanto si el campo <code>appid</code> como el encabezado <code>Ocp-Apim-Subscription-Key</code> se dejan vacíos. Token de autorización: <code>"Bearer" + " " + "access_token"</code> .	encabezado	string
Ocp-Apim-Subscription-Key	(vacío)	Se requiere tanto si el campo <code>appid</code> como el encabezado <code>Authorization</code> se dejan vacíos.	encabezado	string

Mensajes de respuesta

CÓDIGO DE ESTADO HTTP	MOTIVO
400	Solicitud incorrecta. Compruebe los parámetros de entrada y la respuesta del error detallado.
401	Credenciales no válidas.
500	Error de servidor. Si el error continúa, infórmenos. Facilítenos la fecha y hora aproximadas de la solicitud, junto con el Id. de solicitud incluido en el encabezado de respuesta <code>X-MS-Trans-Info</code> .

CÓDIGO DE ESTADO HTTP	MOTIVO
503	Servicio no disponible temporalmente. Vuelva a intentarlo e infórmenos si el error continúa.

POST /GetTranslations

Notas de implementación

Recupera una matriz de traducciones de un par de idiomas determinado desde el almacén y el motor de traducción automática. `GetTranslations` es diferente de `Translate` en que devuelve todas las traducciones disponibles.

El URI de solicitud es `https://api.microsofttranslator.com/V2/Http.svc/GetTranslations`.

El cuerpo de la solicitud incluye el objeto opcional `TranslationOptions`, que tiene este formato:

```
<TranslateOptions xmlns="http://schemas.datacontract.org/2004/07/Microsoft.MT.Web.Service.V2">
  <Category>string-value</Category>
  <ContentType>text/plain</ContentType>
  <ReservedFlags></ReservedFlags>
  <State>int-value</State>
  <Uri>string-value</Uri>
  <User>string-value</User>
</TranslateOptions>
```

El objeto `TranslateOptions` contiene los valores en la siguiente lista. Son todos opcionales y toman como valor predeterminado las opciones de configuración más comunes. Los elementos especificados deben incluirse en orden alfabético.

- `Category`: una cadena que contiene la categoría (dominio) de la traducción. El valor predeterminado es `general`.
- `ContentType`: la única opción admitida, así como la predeterminada, es `text/plain`.
- `IncludeMultipleMTAlternatives`: una marca booleana para especificar si se debe devolver más de una alternativa del motor de traducción automática. Los valores válidos son `true` y `false` (distinguen mayúsculas de minúsculas). El valor predeterminado es `false`, que devuelve solo una alternativa. El establecimiento de la marca en `true` permite crear alternativas artificiales en la traducción, totalmente integradas con Collaborative Translation Framework (CTF). La característica permite la devolución de alternativas de oraciones que no tienen ninguna traducción en CTF agregando alternativas artificiales de la lista de las N mejores opciones del decodificador.
 - Clasificaciones. Las clasificaciones se aplican de la siguiente manera:
 - la mejor traducción automática tiene una clasificación de 5.
 - Las alternativas de CTF reflejan la autoridad del revisor. Varían de -10 a +10.
 - Las alternativas de traducción (las N mejores) generadas automáticamente tienen una clasificación de 0 y un grado de coincidencia de 100.
 - Número de alternativas. El número de alternativas devueltas puede ser tan alto como el nivel especificado en `maxTranslations`, pero puede ser más bajo.
 - Pares de idiomas. Esta funcionalidad no está disponible para las traducciones entre chino simplificado y tradicional, en ambas direcciones. Está disponible para el resto de pares de idiomas admitidos de Microsoft Translator.
- `State`: estado del usuario para ayudar a poner en correlación la solicitud y la respuesta. Se devolverá el mismo contenido en la respuesta.
- `Uri`: Filtrar los resultados por este URI. Si no se establece ningún valor, el valor predeterminado es `all`.
- `User`: Filtrar los resultados por este usuario. Si no se establece ningún valor, el valor predeterminado es `all`.

La solicitud `Content-Type` debe ser `text/xml`.

Valor devuelto: Este es el formato de la respuesta:

```
<GetTranslationsResponse xmlns="http://schemas.datacontract.org/2004/07/Microsoft.MT.Web.Service.V2"
  xmlns:i="https://www.w3.org/2001/XMLSchema-instance">
  <From>Two character language code</From>
  <State/>
  <Translations>
    <TranslationMatch>
      <Count>int-value</Count>
      <MatchDegree>int-value</MatchDegree>
      <MatchedOriginalText/>
      <Rating>int value</Rating>
      <TranslatedText>string-value</TranslatedText>
    </TranslationMatch>
  </Translations>
</GetTranslationsResponse>
```

Esta respuesta incluye un elemento `GetTranslationsResponse` que contiene los valores siguientes:

- `Translations`: una matriz de las coincidencias encontradas, almacenada en los objetos `TranslationMatch` (descritos en la siguiente sección). Las traducciones pueden incluir ligeras variantes del texto original (coincidencia aproximada). Las traducciones se ordenarán: las coincidencias del 100 % en primer lugar, luego las coincidencias aproximadas.
- `From`: si el método no especifica un idioma `From`, este valor procederá de la detección de idioma automática. En caso contrario, será el idioma `From` especificado.
- `State`: estado del usuario para ayudar a poner en correlación la solicitud y la respuesta. Contiene el valor proporcionado en el parámetro `TranslateOptions`.

El objeto `TranslationMatch` consta de estos valores:

- `Error`: el código de error, si se ha producido un error de una cadena de entrada específica. De lo contrario, el campo está vacío.
- `MatchDegree`: indica el grado de coincidencia del texto de entrada con el texto original encontrado en el almacén. El sistema hace coincidir las oraciones según el almacén, incluidas las coincidencias inexactas. El valor devuelto oscila entre 0 y 100, donde 0 equivale a ninguna similitud y 100 a una coincidencia exacta con distinción de mayúsculas y minúsculas.
- `MatchedOriginalText`: El texto original del que se encontraron coincidencias para este resultado. Este valor solo se devuelve si el texto original de la coincidencia era diferente del texto de entrada. Se utiliza para devolver el texto de origen de una coincidencia aproximada. Este valor no se devuelve para los resultados de Microsoft Translator.
- `Rating`: Indica la autoridad de la persona que toma la decisión de calidad. Los resultados de la traducción automática tienen una clasificación de 5. Generalmente, las traducciones proporcionadas de forma anónima tienen una clasificación de 1 a 4. Generalmente, las traducciones proporcionadas de forma autoritaria tienen una clasificación de 6 a 10.
- `Count`: El número de veces que se ha seleccionado esta traducción con esta clasificación. El valor es 0 para la respuesta traducida automáticamente.
- `TranslatedText`: El texto traducido.

Clase de respuesta (estado 200)

Un objeto `GetTranslationsResponse` en el formato que se ha descrito anteriormente.

string

Tipo de contenido de la respuesta: application/xml

Parámetros

PARÁMETRO	VALOR	DESCRIPCIÓN	TIPO DE PARÁMETRO	TIPO DE DATOS
appid	(vacío)	Necesario. Si se usa el encabezado <code>Authorization</code> o <code>Ocp-Apim-Subscription-Key</code> , deje vacío el campo <code>appid</code> . De lo contrario, incluya una cadena que contenga <code>"Bearer" + " " + "access_token"</code> .	query	string
text	(vacío)	Necesario. Una cadena que representa el texto a traducir. El tamaño máximo del texto es 10 000 caracteres.	query	string
De	(vacío)	Necesario. Una cadena que representa el código de idioma del texto que se traduce.	query	string
to	(vacío)	Necesario. Una cadena que representa el código de idioma del idioma al que se va a traducir el texto.	query	string
maxTranslations	(vacío)	Necesario. Un entero que representa el número máximo de traducciones que se van a devolver.	query	integer
Authorization	(vacío)	Se requiere tanto si el campo <code>appid</code> como el encabezado <code>Ocp-Apim-Subscription-Key</code> se dejan vacíos. Token de autorización: <code>"Bearer" + " " + "access_token"</code> .	string	encabezado
Ocp-Apim-Subscription-Key	(vacío)	Se requiere tanto si el campo <code>appid</code> como el encabezado <code>Authorization</code> se dejan vacíos.	encabezado	string

Mensajes de respuesta

CÓDIGO DE ESTADO HTTP	MOTIVO
400	Solicitud incorrecta. Compruebe los parámetros de entrada y la respuesta del error detallado.
401	Credenciales no válidas.
500	Error de servidor. Si el error continúa, infórmenos. Facilítenos la fecha y hora aproximadas de la solicitud, junto con el Id. de solicitud incluido en el encabezado de respuesta <code>X-MS-Trans-Info</code> .
503	Servicio no disponible temporalmente. Vuelva a intentarlo e infórmenos si el error continúa.

POST /GetTranslationsArray

Notas de implementación

Recupere varios candidatos de traducción de varios textos de origen.

El URI de solicitud es `https://api.microsofttranslator.com/V2/Http.svc/GetTranslationsArray`.

Este es el formato del cuerpo de la solicitud:

```
<GetTranslationsArrayRequest>
  <AppId></AppId>
  <From>language-code</From>
  <MaxTranslations>int-value</MaxTranslations>
  <Options>
    <Category xmlns="http://schemas.datacontract.org/2004/07/Microsoft.MT.Web.Service.V2">string-
value</Category>
    <ContentType
xmlns="http://schemas.datacontract.org/2004/07/Microsoft.MT.Web.Service.V2">text/plain</ContentType>
    <ReservedFlags xmlns="http://schemas.datacontract.org/2004/07/Microsoft.MT.Web.Service.V2"/>
    <State xmlns="http://schemas.datacontract.org/2004/07/Microsoft.MT.Web.Service.V2">int-value</State>
    <Uri xmlns="http://schemas.datacontract.org/2004/07/Microsoft.MT.Web.Service.V2">string-value</Uri>
    <User xmlns="http://schemas.datacontract.org/2004/07/Microsoft.MT.Web.Service.V2">string-value</User>
  </Options>
  <Texts>
    <string xmlns="http://schemas.microsoft.com/2003/10/Serialization/Arrays">string-value</string>
  </Texts>
  <To>language-code</To>
</GetTranslationsArrayRequest>
```

`GetTranslationsArrayRequest` incluye estos elementos:

- `AppId`: Necesario. Si se usa el encabezado `Authorization`, deje vacío el campo `AppId`. De lo contrario, incluya una cadena que contenga `"Bearer" + " " + "access_token"`.
- `From`: Necesario. Una cadena que representa el código de idioma del texto que se traduce.
- `MaxTranslations`: Necesario. Un entero que representa el número máximo de traducciones que se van a devolver.
- `options`: Opcional. Un objeto `options` que contiene los valores siguientes. Son todos opcionales y toman como valor predeterminado las opciones de configuración más comunes. Los elementos especificados deben incluirse en orden alfabético.
 - `Category`: una cadena que contiene la categoría (dominio) de la traducción. El valor predeterminado es `general`.

- `ContentType` : la única opción admitida, así como la predeterminada, es `text/plain`.
- `IncludeMultipleMTAlternatives` : una marca booleana para especificar si se debe devolver más de una alternativa del motor de traducción automática. Los valores válidos son `true` y `false` (distinguen mayúsculas de minúsculas). El valor predeterminado es `false`, que devuelve solo una alternativa. El establecimiento de la marca en `true` permite generar alternativas artificiales en la traducción, totalmente integradas con Collaborative Translation Framework (CTF). La característica permite la devolución de alternativas de oraciones que no tienen ninguna alternativa en CTF agregando alternativas artificiales de la lista de las N mejores opciones del descodificador.
- Clasificaciones. Las clasificaciones se aplican de la siguiente manera:
 - la mejor traducción automática tiene una clasificación de 5.
 - Las alternativas de CTF reflejan la autoridad del revisor. Varían de -10 a +10.
 - Las alternativas de traducción (las N mejores) generadas automáticamente tienen una clasificación de 0 y un grado de coincidencia de 100.
- Número de alternativas. El número de alternativas devueltas puede ser tan alto como el nivel especificado en `maxTranslations`, pero puede ser más bajo.
- Pares de idiomas. Esta funcionalidad no está disponible para las traducciones entre chino simplificado y tradicional, en ambas direcciones. Está disponible para el resto de pares de idiomas admitidos de Microsoft Translator.
- `State` : estado del usuario para ayudar a poner en correlación la solicitud y la respuesta. Se devolverá el mismo contenido en la respuesta.
- `Uri` : Filtrar los resultados por este URI. Si no se establece ningún valor, el valor predeterminado es `all`.
- `User` : Filtrar los resultados por este usuario. Si no se establece ningún valor, el valor predeterminado es `all`.
- `Texts` : Necesario. Una matriz que contiene el texto para la traducción. Todas las cadenas deben estar en el mismo idioma. El total de todo el texto que se debe traducir no puede superar los 10 000 caracteres. El número máximo de elementos de la matriz es 10.
- `To` : Necesario. Una cadena que representa el código de idioma del idioma al que se va a traducir el texto.

Puede omitir los elementos opcionales. Los elementos que son elementos secundarios directos de `GetTranslationsArrayRequest` deben aparecer en orden alfabético.

La solicitud `Content-Type` debe ser `text/xml`.

Valor devuelto: Este es el formato de la respuesta:

```
<ArrayOfGetTranslationsResponse xmlns="http://schemas.datacontract.org/2004/07/Microsoft.MT.Web.Service.V2"
xmlns:i="https://www.w3.org/2001/XMLSchema-instance">
  <GetTranslationsResponse>
    <From>language-code</From>
    <State/>
    <Translations>
      <TranslationMatch>
        <Count>int-value</Count>
        <MatchDegree>int-value</MatchDegree>
        <MatchedOriginalText>string-value</MatchedOriginalText>
        <Rating>int-value</Rating>
        <TranslatedText>string-value</TranslatedText>
      </TranslationMatch>
      <TranslationMatch>
        <Count>int-value</Count>
        <MatchDegree>int-value</MatchDegree>
        <MatchedOriginalText/>
        <Rating>int-value</Rating>
        <TranslatedText>string-value</TranslatedText>
      </TranslationMatch>
    </Translations>
  </GetTranslationsResponse>
</ArrayOfGetTranslationsResponse>
```

Cada elemento `GetTranslationsResponse` contiene estos valores:

- `Translations`: una matriz de las coincidencias encontradas, almacenada en los objetos `TranslationMatch` (descritos en la siguiente sección). Las traducciones pueden incluir ligeras variantes del texto original (coincidencia aproximada). Las traducciones se ordenarán: las coincidencias del 100 % en primer lugar, luego las coincidencias aproximadas.
- `From`: si el método no especifica un idioma `From`, este valor procederá de la detección de idioma automática. En caso contrario, será el idioma `From` especificado.
- `State`: estado del usuario para ayudar a poner en correlación la solicitud y la respuesta. Contiene el valor proporcionado en el parámetro `TranslateOptions`.

El objeto `TranslationMatch` contiene los valores siguientes:

- `Error`: el código de error, si se ha producido un error de una cadena de entrada específica. De lo contrario, el campo está vacío.
- `MatchDegree`: indica el grado de coincidencia del texto de entrada con el texto original encontrado en el almacén. El sistema hace coincidir las oraciones según el almacén, incluidas las coincidencias inexactas. El valor devuelto oscila entre 0 y 100, donde 0 equivale a ninguna similitud y 100 a una coincidencia exacta con distinción de mayúsculas y minúsculas.
- `MatchedOriginalText`: El texto original del que se encontraron coincidencias para este resultado. Este valor solo se devuelve si el texto original de la coincidencia era diferente del texto de entrada. Se utiliza para devolver el texto de origen de una coincidencia aproximada. Este valor no se devuelve para los resultados de Microsoft Translator.
- `Rating`: Indica la autoridad de la persona que toma la decisión de calidad. Los resultados de la traducción automática tienen una clasificación de 5. Generalmente, las traducciones proporcionadas de forma anónima tienen una clasificación de 1 a 4. Generalmente, las traducciones proporcionadas de forma autoritaria tienen una clasificación de 6 a 10.
- `count`: El número de veces que se ha seleccionado esta traducción con esta clasificación. El valor es 0 para la respuesta traducida automáticamente.
- `TranslatedText`: El texto traducido.

Clase de respuesta (estado 200)

string

Tipo de contenido de la respuesta: application/xml

Parámetros

PARÁMETRO	VALOR	DESCRIPCIÓN	TIPO DE PARÁMETRO	TIPO DE DATOS
Authorization	(vacío)	Se requiere tanto si el campo <code>appid</code> como el encabezado <code>Ocp-Apim-Subscription-Key</code> se dejan vacíos. Token de autorización: <code>"Bearer" + " " + "access_token"</code> .	encabezado	string
Ocp-Apim-Subscription-Key	(vacío)	Se requiere tanto si el campo <code>appid</code> como el encabezado <code>Authorization</code> se dejan vacíos.	encabezado	string

Mensajes de respuesta

CÓDIGO DE ESTADO HTTP	MOTIVO
400	Solicitud incorrecta. Compruebe los parámetros de entrada y la respuesta del error detallado.
401	Credenciales no válidas.
500	Error de servidor. Si el error continúa, infórmenos. Facilítenos la fecha y hora aproximadas de la solicitud, junto con el Id. de solicitud incluido en el encabezado de respuesta <code>X-MS-Trans-Info</code> .
503	Servicio no disponible temporalmente. Vuelva a intentarlo e infórmenos si el error continúa.

Pasos siguientes

[Migrar Translator Text API de V2 a V3](#)

Uso del método TransformText

13/01/2020 • 4 minutes to read • [Edit Online](#)

NOTE

Este método está obsoleto. No está disponible en la versión 3.0 de Translator Text API.

El método TransformText es una función de normalización de texto para los medios sociales, que devuelve una forma normalizada de la entrada. El método se puede utilizar como un paso de preprocesamiento de traducción automática u otras aplicaciones que esperan un texto de entrada limpio que no se encuentra normalmente en el contenido de los medios sociales o generado por el usuario. La función actualmente solo funciona con texto en inglés.

El método es un servicio RESTful que usa GET a través de HTTP. Admite la serialización de JSON y XML simple.

Parámetros

PARÁMETRO	DESCRIPCIÓN
Encabezado de autorización	Obligatorio Encabezado HTTP utilizado para identificar la aplicación. Usar la clave: "Autorización" y el valor: "Portador" + " " + token de acceso. Para obtener más información, vaya aquí.
language	Obligatorio Una cadena que representa el código de idioma. Este parámetro solo admite inglés con en como nombre del idioma.
category	Opcional Una cadena que contiene la categoría o el dominio de la traducción. Este parámetro admite solo la opción predeterminada general .
sentence	Necesario Una frase que desee corregir.

Valor devuelto

El valor devuelto proporciona la frase transformada.

```
GetTranslationsResponse Microsoft.Translator.GetTranslations(appId, text, from, to, maxTranslations, options);
TransformTextResponse
{
    int ec;           // A positive number representing an error condition
    string em;        // A descriptive error message
    string sentence;   // transformed text
}
```

Ejemplo

```
using System;
using System.Text;
```

```

using System.Net;
using System.IO;
//Requires System.Runtime.Serialization assembly to be referenced
using System.Runtime.Serialization.Json;
using System.Runtime.Serialization;
//Requires System.Web assembly to be referenced
using System.Web;
using System.Media;
using System.Threading;
namespace MicrosoftTranslatorSdk.HttpSamples
{
    class Program
    {
        static void Main(string[] args)
        {
            AdmAccessToken admToken;
            string headerValue;
            //Get Client Id and Client Secret from https://datamarket.azure.com/developer/applications/
            //Refer obtaining AccessToken (https://msdn.microsoft.com/library/hh454950.aspx)
            AdmAuthentication admAuth = new AdmAuthentication("clientId", "client secret");

            try
            {
                admToken = admAuth.GetAccessToken();
                // Create a header with the access_token property of the returned token
                headerValue = "Bearer " + admToken.access_token;
                TransformTextMethod(headerValue);
                Console.WriteLine("Press any key to continue...");
                Console.ReadKey(true);
            }
            catch (WebException e)
            {
                ProcessWebException(e);
                Console.WriteLine("Press any key to continue...");
                Console.ReadKey(true);
            }
            catch (Exception ex)
            {
                Console.WriteLine(ex.Message);
                Console.WriteLine("Press any key to continue...");
                Console.ReadKey(true);
            }
        }
        private static void TransformTextMethod(string authToken)
        {
            Console.WriteLine("Enter Text to transform: (e.g Dis is 2 strange i juss wanna go home soooooooooo)");
            string textToTransform = Console.ReadLine();
            string language = "en";
            string domain = "general";
            //Keep appId parameter blank as we are sending access token in authorization header.
            string url = string.Format("https://api.microsofttranslator.com/V3/json/TransformText?sentence={0}&category={1}&language={2}", textToTransform, domain, language); ;
            TransformTextResponse transformTextResponse = new TransformTextResponse();
            HttpWebRequest httpWebRequest = (HttpWebRequest)WebRequest.Create(url);
            httpWebRequest.Headers.Add("Authorization", authToken);
            using (WebResponse response = httpWebRequest.GetResponse())
            {
                using( StreamReader sr = new StreamReader(response.GetResponseStream()))
                {
                    using (MemoryStream ms = new MemoryStream(Encoding.UTF8.GetBytes(sr.ReadToEnd())))
                    {
                        DataContractJsonSerializer serializer = new
DataContractJsonSerializer(typeof(TransformTextResponse));
                        //Get deserialized TransformTextResponse object from JSON stream
                        transformTextResponse = (TransformTextResponse)serializer.ReadObject(ms);
                        Console.WriteLine("Transformed sentence: {0}", transformTextResponse.sentence);
                    }
                }
            }
        }
    }
}

```

```

    }
}
private static void ProcessWebException(WebException e)
{
    Console.WriteLine("{0}", e.ToString());
    // Obtain detailed error information
    string strResponse = string.Empty;
    using (HttpWebResponse response = (HttpWebResponse)e.Response)
    {
        using (Stream responseStream = response.GetResponseStream())
        {
            using (StreamReader sr = new StreamReader(responseStream, System.Text.Encoding.ASCII))
            {
                strResponse = sr.ReadToEnd();
            }
        }
    }
    Console.WriteLine("Http status code={0}, error message={1}", e.Status, strResponse);
}
}
[DataContract]
public class TransformTextResponse
{
    [DataMember]
    public int ec { get; set; }
    [DataMember]
    public string em { get; set; }
    [DataMember]
    public string sentence { get; set; }
}
[DataContract]
public class AdmAccessToken
{
    [DataMember]
    public string access_token { get; set; }
    [DataMember]
    public string token_type { get; set; }
    [DataMember]
    public string expires_in { get; set; }
    [DataMember]
    public string scope { get; set; }
}
public class AdmAuthentication
{
    public static readonly string DatamarketAccessUri =
"https://datamarket.accesscontrol.windows.net/v2/OAuth2-13";
    private string clientId;
    private string clientSecret;
    private string request;
    private AdmAccessToken token;
    private Timer accessTokenRenewer;
    //Access token expires every 10 minutes. Renew it every 9 minutes only.
    private const int RefreshTokenDuration = 9;
    public AdmAuthentication(string clientId, string clientSecret)
    {
        this.clientId = clientId;
        this.clientSecret = clientSecret;
        //If clientid or client secret has special characters, encode before sending request
        this.request = string.Format("grant_type=client_credentials&client_id={0}&client_secret={1}&scope=http://api.microsofttranslator.com", HttpUtility.UrlEncode(clientId),
HttpUtility.UrlEncode(clientSecret));
        this.token = HttpPost(DatamarketAccessUri, this.request);
        //renew the token every specified minutes
        accessTokenRenewer = new Timer(new TimerCallback(OnTokenExpiredCallback), this,
TimeSpan.FromMinutes(RefreshTokenDuration), TimeSpan.FromMilliseconds(-1));
    }
    public AdmAccessToken GetAccessToken()
    {
        return this.token;
    }
}

```



```

    }
    private void RenewAccessToken()
    {
        AdmAccessToken newAccessToken = HttpPost(DatamarketAccessUri, this.request);
        this.token = newAccessToken;
        Console.WriteLine(string.Format("Renewed token for user: {0} is: {1}", this.clientId,
this.token.access_token));
    }
    private void OnTokenExpiredCallback(object stateInfo)
    {
        try
        {
            RenewAccessToken();
        }
        catch (Exception ex)
        {
            Console.WriteLine(string.Format("Failed renewing access token. Details: {0}", ex.Message));
        }
        finally
        {
            try
            {
                accessTokenRenewer.Change(TimeSpan.FromMinutes(RefreshTokenDuration),
TimeSpan.FromMilliseconds(-1));
            }
            catch (Exception ex)
            {
                Console.WriteLine(string.Format("Failed to reschedule the timer to renew access token.
Details: {0}", ex.Message));
            }
        }
    }
}
private AdmAccessToken HttpPost(string DatamarketAccessUri, string requestDetails)
{
    //Prepare OAuth request
    WebRequest webRequest = WebRequest.Create(DatamarketAccessUri);
    webRequest.ContentType = "application/x-www-form-urlencoded";
    webRequest.Method = "POST";
    byte[] bytes = Encoding.ASCII.GetBytes(requestDetails);
    webRequest.ContentLength = bytes.Length;
    using (Stream outputStream = webRequest.GetRequestStream())
    {
        outputStream.Write(bytes, 0, bytes.Length);
    }
    using (WebResponse webResponse = webRequest.GetResponse())
    {
        DataContractJsonSerializer serializer = new DataContractJsonSerializer(typeof(AdmAccessToken));
        //Get deserialized object from JSON stream
        AdmAccessToken token = (AdmAccessToken)serializer.ReadObject(webResponse.GetResponseStream());
        return token;
    }
}
}
}
}

```

Cómo usar los informes de Collaborative Translation Framework (CTF)

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

NOTE

Este método está obsoleto. No está disponible en la versión 3.0 de Translator Text API.

Collaborative Translations Framework (CTF), que se utilizaba antes para la versión 2.0 de Translator Text API, ha quedado en desuso a partir del 1 de febrero de 2018. Las funciones `AddTranslation` y `AddTranslationArray` permiten a los usuarios proporcionar correcciones a través del marco Collaborative Translation Framework. Después del 31 de enero de 2018, estas dos funciones no aceptan nuevos envíos de frases, y los usuarios reciben un mensaje de error. Estas funciones se retiraron y no se reemplazarán.

La API de informes de Collaborative Translation Framework (CTF) devuelve estadísticas y el contenido real en el almacén de CTF. Esta API es diferente del método `GetTranslations()` por lo siguiente:

- Devuelve el contenido traducido y su recuento total solo de su cuenta (aplicación o cuenta de Azure Marketplace).
- Devuelve el contenido traducido y su recuento total sin necesidad de una coincidencia de la frase de origen.
- No devuelve la traducción automática.

Punto de conexión

El punto de conexión de la API de informes de CTF es <https://api.microsofttranslator.com/v2/beta/ctfreporting.svc>.

Métodos

NOMBRE	DESCRIPCIÓN
Método <code>GetUserTranslationCounts</code>	Obtiene los recuentos de las traducciones creadas por el usuario.
Método <code>GetUserTranslations</code>	Recupera las traducciones creadas por el usuario.

Estos métodos le permiten:

- Recuperar el conjunto completo de traducciones y correcciones del usuario del identificador de la cuenta para su descarga.
- Obtener la lista de los colaboradores frecuentes. Asegúrese de que se proporciona el nombre de usuario correcto en `AddTranslation()`.
- Crear una interfaz de usuario (IU) que permite a los usuarios de confianza ver todos los candidatos disponibles, si es necesario restringido a una parte de su sitio, según el prefijo URI.

NOTE

Ambos métodos son relativamente lentos y costosos. Se recomienda usarlos con moderación.

Método `GetUserTranslationCounts`

Este método obtiene el recuento de traducciones que se crean por el usuario. Proporciona la lista de recuentos de traducción agrupados por los parámetros uriPrefix, from, to, user, minRating y maxRating.

Sintaxis

```
UserTranslationCount[] GetUserTranslationCounts(  
    string appId,  
    string uriPrefix,  
    string from,  
    string to,  
    int? minRating,  
    int? maxRating,  
    string user,  
    string category  
    DateTime? minDateUtc,  
    DateTime? maxDateUtc,  
    int? skip,  
    int? take);
```

Parámetros

PARÁMETRO	DESCRIPCIÓN
appId	Obligatorio Si se usa el encabezado de autorización, deje el campo appId vacío o incluya una cadena que contenga "Bearer" + " " + token de acceso.
uriPrefix	Opcional Una cadena que contiene el prefijo del URI de la traducción.
De	Opcional Una cadena que representa el código de idioma del texto de traducción.
to	Opcional Una cadena que representa el código de idioma al que se va a traducir el texto.
minRating	Opcional Un valor entero que representa la clasificación de calidad mínima para el texto traducido. El valor válido se encuentra entre -10 y 10. El valor predeterminado es 1.
maxRating	Opcional Un valor entero que representa la clasificación de calidad máxima para el texto traducido. El valor válido se encuentra entre -10 y 10. El valor predeterminado es 1.
user	Opcional Una cadena que se utiliza para filtrar los resultados según el autor del envío.
category	Opcional Una cadena que contiene la categoría o el dominio de la traducción. Este parámetro admite solo la opción predeterminada general.
minDateUtc	Opcional La fecha desde la que desea recuperar las traducciones. La fecha debe tener el formato UTC.
maxDateUtc	Opcional La fecha hasta la que desea recuperar las traducciones. La fecha debe tener el formato UTC.

PARÁMETRO	DESCRIPCIÓN
skip	Opcional El número de resultados que desea omitir en una página. Por ejemplo, si desea omitir las 20 primeras filas de los resultados y ver a partir del 21º registro de resultados, especifique 20 en este parámetro. El valor predeterminado para este parámetro es 0.
take	Opcional El número de resultados que desea recuperar. El número máximo de cada solicitud es 100. El valor predeterminado es 100.

NOTE

Los parámetros de solicitud skip y take habilitan la paginación para un gran número de registros de resultados.

Valor devuelto

El conjunto de resultados contiene la matriz de **UserTranslationCount**. Cada UserTranslationCount tiene los siguientes elementos:

CAMPO	DESCRIPCIÓN
Count	Número de resultados que se recupera
De	Idioma de origen
Rating	Clasificación que se aplica por el remitente en la llamada al método AddTranslation()
Para	Idioma de destino
Identificador URI	URI que se aplica en la llamada al método AddTranslation()
Usuario	Nombre del usuario

Excepciones

EXCEPCIÓN	MESSAGE	CONDICIONES
ArgumentOutOfRangeException	El parámetro " maxDateUtc " debe ser mayor o igual que " minDateUtc ".	El valor del parámetro maxDateUtc es menor que el valor del parámetro minDateUtc .
TranslateApiException	IP está por encima de la cuota.	<ul style="list-style-type: none"> Se alcanzó el límite para el número de solicitudes por minuto. El tamaño de la solicitud sigue estando limitado a 10 000 caracteres. Una cuota por hora y por día limitan el número de caracteres que aceptará la API de Microsoft Translator.

EXCEPCIÓN	MESSAGE	CONDICIONES
TranslateApiException	AppId está por encima de la cuota.	El identificador de la aplicación superó la cuota por hora o por día.

NOTE

La cuota se ajustará para garantizar la equidad entre todos los usuarios del servicio.

Vea ejemplos de código en GitHub

- [C#](#)
- [PHP](#)

Método GetUserTranslations

Este método recupera las traducciones creadas por el usuario. Proporciona las traducciones agrupadas por los parámetros uriPrefix, from, to, user, minRating y maxRating.

Sintaxis

```
UserTranslation[] GetUserTranslations (
    string appId,
    string uriPrefix,
    string from,
    string to,
    int? minRating,
    int? maxRating,
    string user,
    string category
    DateTime? minDateUtc,
    DateTime? maxDateUtc,
    int? skip,
    int? take);
```

Parámetros

PARÁMETRO	DESCRIPCIÓN
appId	Obligatorio Si se usa el encabezado de autorización, deje el campo appId vacío o incluya una cadena que contenga "Bearer" + " " + token de acceso.
uriPrefix	Opcional Una cadena que contiene el prefijo del URI de la traducción.
De	Opcional Una cadena que representa el código de idioma del texto de traducción.
to	Opcional Una cadena que representa el código de idioma al que se va a traducir el texto.
minRating	Opcional Un valor entero que representa la clasificación de calidad mínima para el texto traducido. El valor válido se encuentra entre -10 y 10. El valor predeterminado es 1.

PARÁMETRO	DESCRIPCIÓN
maxRating	Opcional Un valor entero que representa la clasificación de calidad máxima para el texto traducido. El valor válido se encuentra entre -10 y 10. El valor predeterminado es 1.
user	Opcional. Una cadena que se utiliza para filtrar los resultados según el autor del envío
category	Opcional Una cadena que contiene la categoría o el dominio de la traducción. Este parámetro admite solo la opción predeterminada general.
minDateUtc	Opcional La fecha desde la que desea recuperar las traducciones. La fecha debe tener el formato UTC.
maxDateUtc	Opcional La fecha hasta la que desea recuperar las traducciones. La fecha debe tener el formato UTC.
skip	Opcional El número de resultados que desea omitir en una página. Por ejemplo, si desea omitir las 20 primeras filas de los resultados y ver a partir del 21º registro de resultados, especifique 20 en este parámetro. El valor predeterminado para este parámetro es 0.
take	Opcional El número de resultados que desea recuperar. El número máximo de cada solicitud es 100. El valor predeterminado es 50.

NOTE

Los parámetros de solicitud skip y take habilitan la paginación para un gran número de registros de resultados.

Valor devuelto

El conjunto de resultados contiene la matriz de **UserTranslation**. Cada UserTranslation tiene los siguientes elementos:

CAMPO	DESCRIPCIÓN
CreatedDateUtc	La fecha de creación de la entrada mediante AddTranslation()
De	Idioma de origen
OriginalText	Texto de lenguaje de origen que se usa al enviar la solicitud
Rating	Clasificación que se aplica por el remitente en la llamada al método AddTranslation()
Para	Idioma de destino
TranslatedText	Traducción tal y como se envió en la llamada al método AddTranslation()
Identificador URI	URI que se aplica en la llamada al método AddTranslation()

CAMPO	DESCRIPCIÓN
Usuario	Nombre del usuario

Excepciones

EXCEPCIÓN	MESSAGE	CONDICIONES
ArgumentOutOfRangeException	El parámetro " maxDateUtc " debe ser mayor o igual que " minDateUtc ".	El valor del parámetro maxDateUtc es menor que el valor del parámetro minDateUtc .
TranslateApiException	IP está por encima de la cuota.	<ul style="list-style-type: none"> Se alcanzó el límite para el número de solicitudes por minuto. El tamaño de la solicitud sigue estando limitado a 10 000 caracteres. Una cuota por hora y por día limitan el número de caracteres que aceptará la API de Microsoft Translator.
TranslateApiException	AppId está por encima de la cuota.	El identificador de la aplicación superó la cuota por hora o por día.

NOTE

La cuota se ajustará para garantizar la equidad entre todos los usuarios del servicio.

Vea ejemplos de código en GitHub

- [C#](#)
- [PHP](#)

Devolución de las N mejores traducciones

13/01/2020 • 3 minutes to read • [Edit Online](#)

NOTE

Este método está obsoleto. No está disponible en la versión 3.0 de Translator Text API.

Los métodos `GetTranslations()` y `GetTranslationsArray()` de Microsoft Translator API incluyen una marca booleana opcional "IncludeMultipleMTAlternatives". El método devolverá hasta `maxTranslations` alternativas, donde la diferencia se proporciona a partir de la lista de N mejores del motor de traducción.

La firma es:

Sintaxis

C#

```
GetTranslationsResponse Microsoft.Translator.GetTranslations(appId, text, from, to, maxTranslations, options);
```

Parámetros

PARÁMETRO	DESCRIPCIÓN
appId	Obligatorio Si se usa el encabezado de autorización, deje el campo <code>appId</code> vacío o incluya una cadena que contenga "Bearer" + " " + token de acceso.
text	Obligatorio Una cadena que representa el texto que se va a traducir. El tamaño del texto no debe superar los 10 000 caracteres.
De	Obligatorio Una cadena que representa el código de idioma del texto que se va a traducir.
to	Obligatorio Una cadena que representa el código de idioma al que se va a traducir el texto.
maxTranslations	Obligatorio Un entero que representa el número máximo de traducciones que se van a devolver.
options	Opcional Un objeto <code>TranslateOptions</code> que contiene los valores que se enumeran a continuación. Son todos opcionales y toman como valor predeterminado las opciones de configuración más comunes.

- Categoría: La única opción admitida, así como la predeterminada, es "general".
- ContentType: La única opción admitida, así como la predeterminada, es "text/plain".
- Estado: Estado del usuario para ayudar a poner en correlación la solicitud y la respuesta. Se devolverá el mismo contenido en la respuesta.
- IncludeMultipleMTAlternatives: marca para determinar si se devuelve más de una alternativa desde el motor de MT. El valor predeterminado es `false` e incluye solo una alternativa.

Clasificaciones

Las clasificaciones se aplican como sigue: la mejor traducción automática tiene una clasificación de 5. Las alternativas de traducción (las N mejores) generadas automáticamente tienen una clasificación de 0 y un grado de coincidencia de 100.

Número de alternativas

El número de alternativas devueltas va hasta maxTranslations, pero puede ser menor.

Pares de idiomas

Esta funcionalidad no está disponible para las traducciones entre chino simplificado y tradicional, en ambas direcciones. Está disponible para el resto de pares de idiomas admitidos de Microsoft Translator.

Compatibilidad de idiomas y regiones para Translator Text API

13/01/2020 • 10 minutes to read • [Edit Online](#)

Translator Text API admite los siguientes idiomas para la conversión de texto a texto. La traducción automática neuronal (NMT) es el nuevo estándar de traducción automática de alta calidad con tecnologías de inteligencia artificial, y está disponible de forma predeterminada con la versión V3 de Translator Text API cuando hay un sistema neuronal disponible.

[Más información sobre cómo funciona la traducción automática](#)

Traducción

Translator API V2

NOTE

V2 quedó en desuso el 30 de abril de 2018. Migre sus aplicaciones a V3 para aprovechar la nueva funcionalidad disponible exclusivamente en V3.

- Solo estadísticas: no hay ningún sistema neuronal disponible para este idioma.
- Traducción neuronal disponible: hay un sistema neuronal disponible. Utilice el parámetro `category=general` para acceder al sistema neuronal.
- Neuronal como sistema predeterminado: el sistema de traducción neuronal es el predeterminado. Utilice el parámetro `category=smt` para acceder al sistema estadístico y utilizarlo con Microsoft Translator Hub.
- Solo neuronal: la traducción neuronal es la única que está disponible.

Translator API V3 Translator API V3 utiliza el sistema neuronal de forma predeterminada, por lo que los sistemas estadísticos solamente están disponibles cuando no existe un sistema neuronal.

NOTE

Actualmente, un subconjunto de los idiomas neuronales está disponible en Traductor personalizado y agregamos otros adicionales de forma gradual. [Vea los idiomas disponibles actualmente en Traductor personalizado.](#)

IDIOMA	CÓDIGO DE IDIOMA	API V2	API V3
Afrikaans	<code>af</code>	Solo estadísticas	Neuronal
Árabe	<code>ar</code>	Traducción neuronal disponible	Neuronal
Bengalí	<code>bn</code>	Traducción neuronal disponible	Neuronal
Bosnio (latino)	<code>bs</code>	Traducción neuronal disponible	Neuronal

IDIOMA	CÓDIGO DE IDIOMA	API V2	API V3
Búlgaro	bg	Traducción neuronal disponible	Neuronal
Cantonés (tradicional)	yue	Solo estadísticas	Estadística
Catalán	ca	Solo estadísticas	Estadística
Chino simplificado	zh-Hans	Neuronal como sistema predeterminado	Neuronal
Chino tradicional	zh-Hant	Neuronal como sistema predeterminado	Neuronal
Croata	hr	Traducción neuronal disponible	Neuronal
Checo	cs	Traducción neuronal disponible	Neuronal
Danés	da	Traducción neuronal disponible	Neuronal
Neerlandés	nl	Traducción neuronal disponible	Neuronal
English	en	Traducción neuronal disponible	Neuronal
Estonio	et	Traducción neuronal disponible	Neuronal
Fiyiano	fj	Solo estadísticas	Estadística
Filipino	fil	Solo estadísticas	Estadística
Finés	fi	Traducción neuronal disponible	Neuronal
Francés	fr	Traducción neuronal disponible	Neuronal
Alemán	de	Traducción neuronal disponible	Neuronal
Griego	el	Traducción neuronal disponible	Neuronal
Criollo haitiano	ht	Solo estadísticas	Estadística
Hebreo	he	Traducción neuronal disponible	Neuronal

IDIOMA	CÓDIGO DE IDIOMA	API V2	API V3
Hindi	hi	Neuronal como sistema predeterminado	Neuronal
Hmong Daw	mww	Solo estadísticas	Estadística
Húngaro	hu	Traducción neuronal disponible	Neuronal
Islandés	is	Solo neuronal	Neuronal
Indonesio	id	Solo estadísticas	Estadística
Italiano	it	Traducción neuronal disponible	Neuronal
Japonés	ja	Traducción neuronal disponible	Neuronal
Kiswahili	sw	Solo estadísticas	Estadística
Klingon	tlh	Solo estadísticas	Estadística
Klingon (plqaD)	tlh-Qaak	Solo estadísticas	Estadística
Coreano	ko	Traducción neuronal disponible	Neuronal
Letón	lv	Traducción neuronal disponible	Neuronal
Lituano	lt	Traducción neuronal disponible	Neuronal
Malgache	mg	Solo estadísticas	Estadística
Malayo	ms	Solo estadísticas	Estadística
Maltés	mt	Solo estadísticas	Estadística
Maori	mi	Solo neuronal	Neuronal
Noruego	nb	Traducción neuronal disponible	Neuronal
Persa	fa	Traducción neuronal disponible	Neuronal
Polaco	pl	Traducción neuronal disponible	Neuronal

IDIOMA	CÓDIGO DE IDIOMA	API V2	API V3
Portugués	pt	Traducción neuronal disponible	Neuronal
Otomí Querétaro	otq	Solo estadísticas	Estadística
Rumano	ro	Traducción neuronal disponible	Neuronal
Ruso	ru	Traducción neuronal disponible	Neuronal
Samoano	sm	Solo estadísticas	Estadística
Serbio (cirílico)	sr-Cyr1	Solo estadísticas	Estadística
Serbio (latino)	sr-Latn	Solo estadísticas	Estadística
Eslovaco	sk	Traducción neuronal disponible	Neuronal
Esloveno	sl	Traducción neuronal disponible	Neuronal
Español	es	Traducción neuronal disponible	Neuronal
Sueco	sv	Traducción neuronal disponible	Neuronal
Tahitiano	ty	Solo estadísticas	Estadística
Tamil	ta	Traducción neuronal disponible	Neuronal
Telugu	te	Solo neuronal	Neuronal
Tailandés	th	Traducción neuronal disponible	Neuronal
Tongano	to	Solo estadísticas	Estadística
Turco	tr	Traducción neuronal disponible	Neuronal
Ucraniano	uk	Traducción neuronal disponible	Neuronal
Urdu	ur	Solo estadísticas	Estadística
Vietnamita	vi	Traducción neuronal disponible	Neuronal

IDIOMA	CÓDIGO DE IDIOMA	API V2	API V3
Galés	cy	Traducción neuronal disponible	Neuronal
Maya Yucateco	yua	Solo estadísticas	Estadística

Transliteración

El método Transliterate admite los siguientes idiomas. En "Hacia/Desde", "<-->" indica que el idioma se puede transliterar hacia o desde cualquiera de los alfabetos enumerados. "-->" indica que el idioma solo se puede transliterar de un idioma al otro.

IDIOMA	CÓDIGO DE IDIOMA	SCRIPT	HACIA/DESDE	SCRIPT
Árabe	ar	Árabe Arab	<-->	Latino Latn
Bengalí	bn	Bengalí Beng	<-->	Latino Latn
Chino (simplificado)	zh-Hans	Chino simplificado Hans	<-->	Latino Latn
Chino (simplificado)	zh-Hans	Chino simplificado Hans	<-->	Chino tradicional Hant
Chino (tradicional)	zh-Hant	Chino tradicional Hant	<-->	Latino Latn
Chino (tradicional)	zh-Hant	Chino tradicional Hant	<-->	Chino simplificado Hans
Gujarati	gu	Guyaratí Gujr	-->	Latino Latn
Hebreo	he	Hebreo Hebr	<-->	Latino Latn
Hindi	hi	Devanagari Deva	<-->	Latino Latn
Japonés	ja	Japonés Jpan	<-->	Latino Latn
Canarés	kn	Canarés Knda	-->	Latino Latn
Malayalam	ml	Malayalam Mlym	-->	Latino Latn
Maratí	mr	Devanagari Deva	-->	Latino Latn
Odia	or	Odia Orya	<-->	Latino Latn
Punjabi	pa	Gurmukhi Guru	<-->	Latino Latn
Serbio (cirílico)	sr-Cyrl	Cirílico Cyrl	-->	Latino Latn

IDIOMA	CÓDIGO DE IDIOMA	SCRIPT	HACIA/DESDE	SCRIPT
Serbio (latino)	sr-Latn	Latino Latn	-->	Cirílico Cyr1
Tamil	ta	Tamil Tam1	-->	Latino Latn
Telugu	te	Telugu Telu	-->	Latino Latn
Tailandés	th	Tailandés Thai	<-->	Latino Latn

Diccionario

El diccionario admite los siguientes idiomas desde o hacia el inglés con los métodos Lookup y Examples.

IDIOMA	CÓDIGO DE IDIOMA
Afrikaans	af
Árabe	ar
Bengalí	bn
Bosnio (latino)	bs
Búlgaro	bg
Catalán	ca
Chino simplificado	zh-Hans
Croata	hr
Checo	cs
Danés	da
Neerlandés	nl
Estonio	et
Finés	fi
Francés	fr
Alemán	de
Griego	el
Criollo haitiano	ht

IDIOMA	CÓDIGO DE IDIOMA
Hebreo	he
Hindi	hi
Hmong Daw	mww
Húngaro	hu
Islandés	is
Indonesio	id
Italiano	it
Japonés	ja
Kiswahili	sw
Klingon	tlh
Coreano	ko
Letón	lv
Lituano	lt
Malayo	ms
Maltés	mt
Noruego	nb
Persa	fa
Polaco	pl
Portugués	pt
Rumano	ro
Ruso	ru
Serbio (latino)	sr-Latn
Eslovaco	sk
Esloveno	sl

IDIOMA	CÓDIGO DE IDIOMA
Español	es
Sueco	sv
Tamil	ta
Tailandés	th
Turco	tr
Ucraniano	uk
Urdu	ur
Vietnamita	vi
Galés	cy

Detect

Translator Text API detecta todos los idiomas disponibles para traducción y transliteración.

Acceso a la lista de idiomas de Translator Text API mediante programación

Puede recuperar una lista de los idiomas admitidos por Translator Text API v3.0 mediante el método `Languages`. Puede ver la lista por característica, código de idioma o por nombre del idioma en inglés o en cualquier otro idioma admitido. El servicio Microsoft Translator actualiza esta lista automáticamente cuando hay nuevos idiomas disponibles.

[Ver la documentación de referencia de la operación `Languages`](#)

Personalización

Los siguientes idiomas están disponibles para personalización al y del inglés mediante [Traductor personalizado](#).

IDIOMA	CÓDIGO DE IDIOMA
Árabe	ar
Bengalí	bn
Bosnio (latino)	bs
Búlgaro	bg
Chino simplificado	zh-Hans
Chino tradicional	zh-Hant

IDIOMA	CÓDIGO DE IDIOMA
Croata	hr
Checo	cs
Danés	da
Neerlandés	nl
English	en
Estonio	et
Finés	fi
Francés	fr
Alemán	de
Griego	el
Hebreo	he
Hindi	hi
Húngaro	hu
Islandés	is
Indonesio	id
Irlandés	ga
Italiano	it
Japonés	ja
Kiswahili	sw
Coreano	ko
Letón	lv
Lituano	lt
Malgache	mg
Maori	mi

IDIOMA	CÓDIGO DE IDIOMA
Noruego	nb
Persa	fa
Polaco	pl
Portugués	pt
Rumano	ro
Ruso	ru
Samoano	sm
Serbio (latino)	sr-Latn
Eslovaco	sk
Esloveno	sl
Español	es
Sueco	sv
Tailandés	th
Turco	tr
Ucraniano	uk
Vietnamita	vi
Galés	cy

Acceso a la lista en el sitio web de Microsoft Translator

Para echar un vistazo rápido a los idiomas, el sitio web de Microsoft Translator muestra todos los idiomas admitidos por Translator Text API y Speech API. Esta lista no incluye información específica para desarrolladores, por ejemplo, los códigos de idioma.

[Ver la lista de idiomas](#)

Límites de solicitudes de Translator Text

13/01/2020 • 4 minutes to read • [Edit Online](#)

En este artículo se proporcionan los valores de limitación de Translator Text API. Los servicios incluyen traducción, transliteración, detección de la longitud de oraciones, detección de idiomas y traducciones alternativas.

Límites de caracteres y matriz por solicitud

Cada solicitud Translate está limitada a 5000 caracteres. Se le cobrará por cada carácter, no por el número de solicitudes. Se recomienda enviar solicitudes más cortas.

En la siguiente tabla se muestran los límites de caracteres y elementos de matriz correspondientes a cada una de las operaciones de Translator Text API.

OPERACIÓN	TAMAÑO MÁXIMO DEL ELEMENTO DE MATRIZ	NÚMERO MÁXIMO DE ELEMENTOS DE MATRIZ	TAMAÑO MÁXIMO DE LA SOLICITUD (CARACTERES)
Translate	5.000	100	5.000
Transliterar	5.000	10	5.000
Detect	10 000	100	50.000
BreakSentence	10 000	100	50.000
Búsqueda en diccionario	100	10	1000
Ejemplos de diccionario	100 para texto y 100 para traducción (200 en total)	10	2.000

Límites de caracteres por hora

El límite de caracteres por hora se basa en el nivel de suscripción de Translator Text.

La cuota por hora debe consumirse de forma uniforme a lo largo de la hora. Por ejemplo, en el límite del nivel F0 de 2 millones de caracteres por hora, los caracteres deben consumirse a una velocidad no superior a unos 33 300 caracteres por ventana deslizante de un minuto (2 millones de caracteres divididos por 60 minutos).

Si alcanza o sobrepasa estos límites, o bien envía una parte demasiado grande de la cuota en un breve período de tiempo, es probable que reciba una respuesta de superación de cuota. No hay ningún límite en las solicitudes simultáneas.

NIVEL	LÍMITE DE CARACTERES
F0	2 millones de caracteres por hora
S1	40 millones de caracteres por hora
S2 / C2	40 millones de caracteres por hora

NIVEL	LÍMITE DE CARACTERES
S3 / C3	120 millones de caracteres por hora
S4 / C4	200 millones de caracteres por hora

Los límites de las [suscripciones a varios servicios](#) son los mismos que los del nivel S1.

Estos límites están restringidos a modelos de traducción estándar de Microsoft. Los modelos de traducción personalizada que usan Traductor personalizado están limitados a 1800 caracteres por segundo.

Latencia

La API Translator Text tiene una latencia máxima de 15 segundos con los modelos estándar y 120 segundos al usar modelos personalizados. Normalmente, las respuestas para el *texto de 100 caracteres* tardan entre 150 milisegundos y 300 milisegundos en devolverse. Los modelos de traductor personalizados tienen características de latencia similares en la velocidad de solicitudes sostenidas y pueden tener una latencia mayor cuando la tasa de solicitudes es intermitente. Los tiempos de respuesta variarán según el tamaño de la solicitud y el par de idiomas. Si no recibe una traducción o una [respuesta de error](#) en ese período de tiempo, compruebe el código, la conexión de red y vuelva a intentarlo.

Límites de longitud de oraciones

Cuando se usa la función [BreakSentence](#), la longitud de la oración se limita a 275 caracteres. Existen excepciones para estos idiomas:

IDIOMA	CÓDIGO	LÍMITE DE CARACTERES
Chino	zh	132
Alemán	de	290
Italiano	it	280
Japonés	ja	150
Portugués	pt	290
Español	es	280
Italiano	it	280
Tailandés	th	258

NOTE

Este límite no se aplica a las traducciones.

Pasos siguientes

- [Precios](#)
- [Disponibilidad regional](#)

- [Referencia de Translator Text API v3](#)

¿Qué es Custom Translator?

16/01/2020 • 6 minutes to read • [Edit Online](#)

[Custom Translator](#) es una característica del servicio Microsoft Translator que permite a las empresas de traducción, los desarrolladores de aplicaciones y los proveedores de servicios de lenguaje compilar sistemas de traducción automática neuronal (NMT) personalizados. Los sistemas de traducción personalizados se integran fácilmente en las aplicaciones, flujos de trabajo y sitios web existentes. [Custom Translator](#) ofrece unas funcionalidades parecidas a las que presenta [Microsoft Translator Hub](#) para la traducción automática estadística (SMT), pero exclusivamente para sistemas de traducción automática neuronal (NMT).

Los sistemas de traducción compilados con [Custom Translator](#) están disponibles en la misma Microsoft Translator [Text API V3](#) que se basa en la nube, es [segura](#), de alto rendimiento y altamente escalable, que realiza miles de millones de traducciones todos los días.

Custom Translator admite más de 36 idiomas y los asigna directamente a los idiomas disponibles para NMT. Para obtener una lista completa, consulte [Idiomas de Microsoft Translator](#).

Características

Custom Translator proporciona diferentes características a la hora de compilar un sistema de traducción personalizado y posteriormente acceder a él.

CARACTERÍSTICA	DESCRIPCIÓN
Aprovechamiento de la tecnología de traducción automática neuronal	Mejora la traducción aprovechando la traducción automática neuronal (NMT) que proporciona Custom Translator.
Creación de sistemas que conocen la terminología del negocio	Personalice y compile sistemas de traducción mediante documentos paralelos que comprendan la terminología usada en su propia empresa y sector.
Uso de un diccionario para compilar los modelos	Si no tiene un conjunto de datos de aprendizaje, puede entrenar un modelo solo con datos de diccionario.
Colaboración con otros usuarios	Colabore con su equipo compartiendo su trabajo con diferentes personas.
Acceso al modelo de traducción personalizado	Se puede acceder al modelo de traducción personalizado en cualquier momento mediante las aplicaciones y programas existentes a través de Microsoft Translator Text API V3.

Obtención de mejores traducciones

Microsoft Translator lanzó la [traducción automática neuronal \(NMT\)](#) en 2016. La NMT proporciona avances importantes en lo que se refiere a la calidad de la traducción con respecto a la tecnología estándar de [traducción automática estadística \(SMT\)](#). Como la NMT captura mejor el contexto de oraciones completas antes de traducirlas, proporciona unas traducciones de mayor calidad, un tono más natural y más fluidas. [Custom Translator](#) proporciona NMT para sus modelos personalizados lo cual da como resultado una traducción de mejor calidad.

Puede usar los documentos traducidos anteriormente para crear un sistema de traducción. Estos documentos incluyen la terminología y el estilo específicos de un dominio de mejor forma que lo hace un sistema de traducción

estándar. Los usuarios pueden cargar documentos ALIGN, PDF, LCL, HTML, HTM, XLF, TMX, XLIFF, TXT, DOCX y XLSX.

Custom Translator acepta también datos paralelos en el nivel de documento para que la recopilación de datos y su preparación sea más eficaz. Si los usuarios tienen acceso a versiones del mismo contenido en varios idiomas, pero en documentos independientes Custom Translator podrá hacer concordar automáticamente las frases de los distintos documentos.

Si se proporciona el tipo y la cantidad adecuados de datos de aprendizaje, no es raro observar una mejora de la [puntuación BLEU](#) de entre 5 y 10 puntos mediante el uso de Custom Translator.

Productivo y rentable

Con [Custom Translator](#), el aprendizaje e implementación de un sistema personalizado no requiere ningún conocimiento de programación.

Mediante el portal seguro de [Custom Translator](#), los usuarios pueden cargar datos de aprendizaje y entrenar sistemas, probarlos e implementarlos en un entorno de producción mediante una interfaz de usuario intuitiva. El sistema estará disponible para su uso a escala en unas pocas horas (el tiempo real depende del tamaño de los datos de aprendizaje).

También se puede acceder a [Custom Translator](#) mediante programación a través de una [API dedicada](#) (actualmente en versión preliminar). La API permite a los usuarios administrar la creación o actualización del aprendizaje de forma periódica a través de su propia aplicación o servicio web.

El costo de usar un modelo personalizado para traducir contenido se basa en el plan de tarifa de Translator Text API del usuario. Consulte la [página web de precios de Translator Text API](#) para Cognitive Services para más información.

Traduzca de forma segura siempre y en cualquier lugar en todas sus aplicaciones y servicios.

Se puede acceder a los sistemas personalizados fácilmente e integrarlos en cualquier producto o flujo de trabajo empresarial, o en cualquier dispositivo, a través de Microsoft Translator Text API mediante tecnología REST estándar.

Pasos siguientes

- Lea acerca de los [detalles de precios](#).
- Con la [guía de inicio rápido](#) aprenderá a compilar un modelo de traducción en Custom Translator.

Inicio rápido: Compilación, implementación y uso de un modelo personalizado para la traducción

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

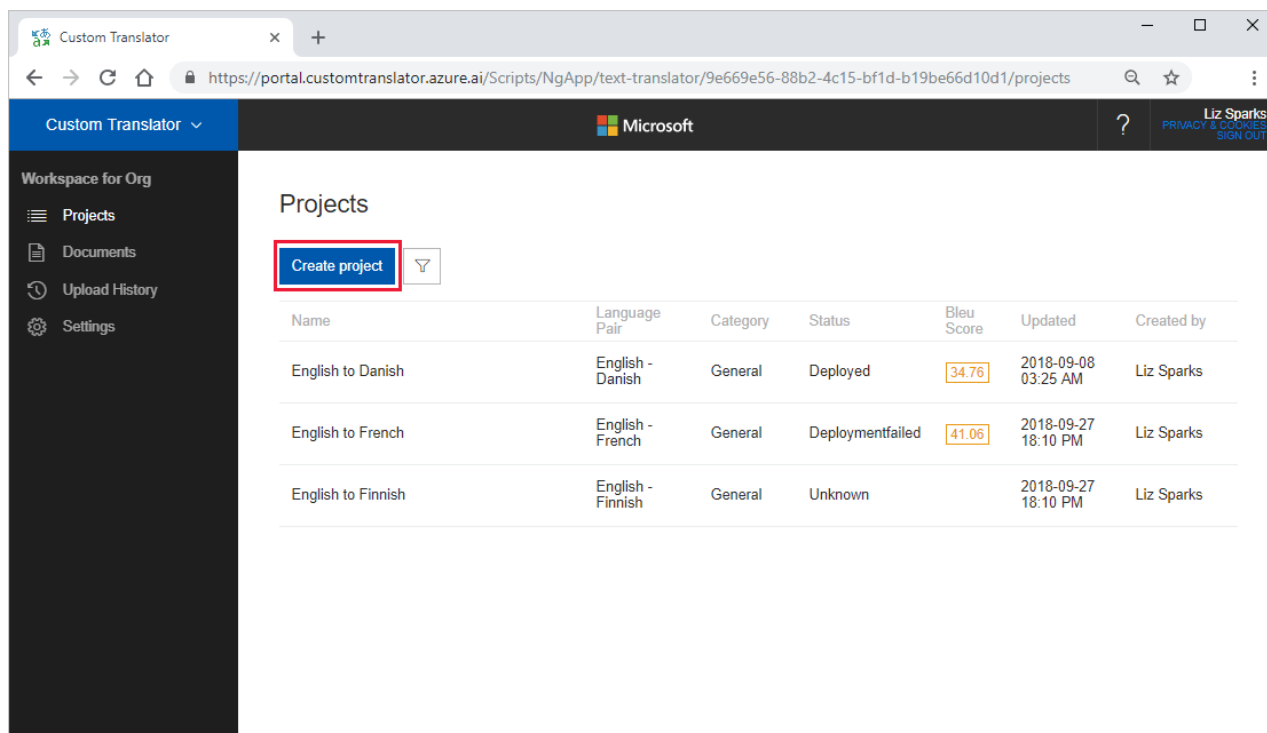
En este artículo se proporcionan instrucciones detalladas para compilar un sistema de traducción con Custom Translator.

Prerequisites

1. Para usar el portal de [Custom Translator](#), necesitará una [cuenta Microsoft](#) o [cuenta de Azure AD](#) (cuenta de organización hospedada en Azure) para iniciar sesión.
2. Una suscripción a Translator Text API a través de Azure Portal. Necesitará la clave de suscripción de Translator Text API para asociarla con el área de trabajo de Custom Translator. Consulte [Cómo suscribirse a Translator Text API](#).
3. Cuando disponga de todo lo anterior, inicie sesión en el portal de [Custom Translator](#). Una vez allí, vaya a la página Configuración en la que podrá asociar la clave de suscripción de Microsoft Translator Text API con el área de trabajo.

Crear un proyecto

En la página de aterrizaje del portal de Custom Translator, haga clic en Nuevo proyecto. En el cuadro de diálogo puede especificar el nombre de proyecto deseado, el par de idiomas y la categoría, así como otros campos importantes. Después, guarde el proyecto. Para más información, visite [Creación de proyectos](#).



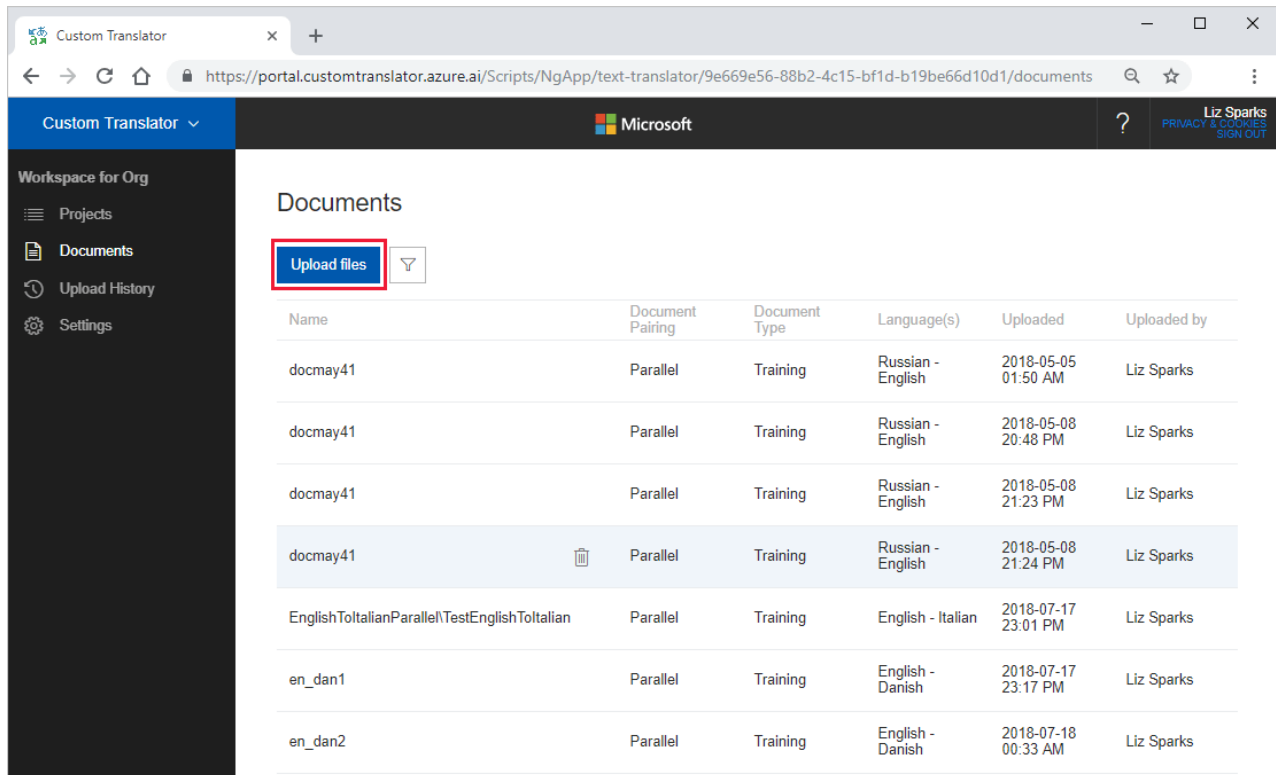
The screenshot shows the Custom Translator portal interface. On the left is a sidebar with navigation options: Projects, Documents, Upload History, and Settings. The main area is titled 'Projects' and contains a 'Create project' button (highlighted with a red box) and a table of existing projects.

Name	Language Pair	Category	Status	Bleu Score	Updated	Created by
English to Danish	English - Danish	General	Deployed	34.76	2018-09-08 03:25 AM	Liz Sparks
English to French	English - French	General	Deployment failed	41.06	2018-09-27 18:10 PM	Liz Sparks
English to Finnish	English - Finnish	General	Unknown		2018-09-27 18:10 PM	Liz Sparks

Cargar documentos

A continuación, cargue los conjuntos de documentos de [aprendizaje](#), [optimización](#) y [pruebas](#). Puede cargar documentos [paralelos](#) y documentos de cuadro combinado. También puede cargar un [diccionario](#).

Puede cargar documentos desde la pestaña de documentos o desde la página de un proyecto concreto.



Custom Translator

Microsoft

Workspace for Org

- Projects
- Documents
- Upload History
- Settings

Documents

[Upload files](#)

Name	Document Pairing	Document Type	Language(s)	Uploaded	Uploaded by
docmay41	Parallel	Training	Russian - English	2018-05-05 01:50 AM	Liz Sparks
docmay41	Parallel	Training	Russian - English	2018-05-08 20:48 PM	Liz Sparks
docmay41	Parallel	Training	Russian - English	2018-05-08 21:23 PM	Liz Sparks
docmay41	Parallel	Training	Russian - English	2018-05-08 21:24 PM	Liz Sparks
EnglishToItalianParallelTestEnglishToItalian	Parallel	Training	English - Italian	2018-07-17 23:01 PM	Liz Sparks
en_dan1	Parallel	Training	English - Danish	2018-07-17 23:17 PM	Liz Sparks
en_dan2	Parallel	Training	English - Danish	2018-07-18 00:33 AM	Liz Sparks

Al cargar los documentos, elija el tipo de documento (aprendizaje, optimización o pruebas) y el par de idiomas. Si carga documentos paralelos tendrá que especificar además un nombre de documento. Para más información, visite [Carga de documentos](#).

Crear un modelo

Una vez cargados todos los documentos necesarios el siguiente paso es crear el modelo.

Seleccione el proyecto que ha creado. Verá todos los documentos que ha cargado que compartan el par de idiomas con este proyecto. Seleccione los documentos que desea incluir en el modelo. Puede seleccionar datos de [aprendizaje](#), [optimización](#) y [prueba](#) o solo seleccionar datos de aprendizaje y dejar que Custom Translator compile automáticamente los conjuntos de datos de optimización y prueba del modelo.

Custom Translator

Microsoft

Workspace for Org

- Projects
- Documents
- Upload History
- Settings

Projects > English to Danish

Category ID: 00000000-0000-0000-0000-000000000000-GENERAL
Language Pair: English - Danish
Category: General
[Edit project](#)

Data Models

Create model Upload files

6 Documents

<input type="checkbox"/>	Name	Document Pairing	Document Type	Language(s)	English Sentences	Danish Sentences
<input type="checkbox"/>	en_dan1	Parallel	Training	English - Danish	60,109	60,109
<input type="checkbox"/>	en_dan2	Parallel	Training	English - Danish	86,980	86,980
<input type="checkbox"/>	en_dan3	Parallel	Training	English - Danish	79,487	79,487
<input type="checkbox"/>	English to Danish Tuning 2	Parallel	Tuning	English - Danish	3,500	3,500
<input type="checkbox"/>	EU_15900000	Parallel	Tuning	Danish - English	4,355	4,355
<input type="checkbox"/>	English to Danish Testing 2 - Para	Parallel	Testing	English - Danish	3,050	3,050

Cuando haya acabado de seleccionar los documentos que desee, haga clic en el botón Crear modelo para crear el modelo e iniciar el aprendizaje. Puede ver el estado del aprendizaje y los detalles de todos los modelos entrenados, en la pestaña Modelos.

Para más información, visite [Creación de un modelo](#).

Análisis del modelo

Una vez que el aprendizaje se ha completado correctamente, inspeccione los resultados. La puntuación BLEU es una métrica que indica la calidad de la traducción. También puede comparar manualmente las traducciones realizadas con el modelo personalizado con las traducciones proporcionadas en el conjunto de pruebas. Para ello, vaya a la pestaña "Prueba" y haga clic en "Resultados del sistema". Si inspecciona manualmente varias de estas traducciones podrá hacerse una buena idea de la calidad de la traducción que su sistema ha producido. Para más información, visite la página de [resultados de pruebas del sistema](#).

Implementación de un modelo entrenado

Cuando esté listo para implementar el modelo entrenado, haga clic en el botón "Implementar". Puede tener un modelo implementado por proyecto y puede ver el estado de la implementación en la columna Estado. Para más información, visite [Model Deployment](#) (Implementación de un modelo)

Custom Translator

Microsoft

Workspace for Org

- Projects
- Documents
- Upload History
- Settings

Projects > English to Danish

Category ID: 00000000-0000-0000-0000-000000000000-GENERAL
Language Pair: English - Danish
Category: General
[Edit project](#)

Data Models

4 models

Name	Status	Bleu Score	Baseline Bleu Score	Training	Dictionary	Tuning	Test	Model Action
English >> Danish 1st model	Undeployed	34.76	34.27	52,015	0	2,500	2,500	Deploy
English >> Danish 2nd model	Succeeded	34.75	34.52	52,015	0	2,500	2,500	Deploy
English >> Danish 3rd model	Succeeded	31	30.88	57,015	0	4,139	2,868	Deploy
Second model for EN to DAN	Succeeded	34.58	34.4	52,015	0	2,500	2,500	Deploy

Uso de un modelo implementado

Se puede acceder a los modelos implementados a través de Microsoft Translator [Text API V3](#) especificando [CategoryID](#). Puede encontrar más información sobre Translator Text API en la página web de [referencia de API](#).

Pasos siguientes

- Vaya a [Custom Translator workspace and manage your projects](#) (Área de trabajo de Custom Translator y administración de proyectos).

¿Qué es un área de trabajo de Custom Translator?

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

Un área de trabajo es un área para crear y compilar el sistema de traducción personalizada. Un área de trabajo puede contener varios proyectos, modelos y documentos. Todo el trabajo que realice en Custom Translator se encuentra dentro de un área de trabajo específica.

Las áreas de trabajo son privadas y solo pueden acceder a ellas usted mismo y las personas a las que invite. Las personas a las que no invite no podrán acceder al contenido del área de trabajo. Puede invitar a tantas personas como quiera al área de trabajo y modificar o quitar el acceso en cualquier momento. También puede crear una nueva área de trabajo. De forma predeterminada, un área de trabajo no contendrá ninguno de los proyectos o documentos que están dentro de las otras áreas de trabajo.

¿Qué es un proyecto de Custom Translator?

Un proyecto es un contenedor para un modelo, documentos y pruebas. Cada proyecto incluye automáticamente todos los documentos que se cargan en esa área de trabajo con el par de idiomas correcto. Por ejemplo, si tiene proyectos en inglés y en español, y un proyecto de inglés a español, se incluirán los mismos documentos en todos los proyectos. Cada proyecto tiene un id. de categoría asociado que se usa en las consultas a la [API V3](#) para recibir las traducciones. El parámetro id. de categoría se utiliza para obtener las traducciones de un sistema personalizado creado con Custom Translator.

Categorías de proyecto

La categoría identifica el dominio (el área de terminología y estilo que quiere usar) para el proyecto. Elija la que mejor se ajuste a los documentos. En algunos casos, su elección de categoría influye directamente en el comportamiento de Custom Translator.

Tenemos dos conjuntos de modelos de línea base. Son General y Tecnología. Si la categoría **Tecnología** está seleccionada, se utilizarán los modelos de línea de base técnicos. Para cualquier otra selección de categoría, se utilizan los modelos de línea de base generales. El modelo de línea de base tecnológico se desempeña bien en el ámbito de la tecnología, pero muestra una calidad inferior, si las frases que se utilizan para la traducción no se encuentran dentro del dominio tecnológico. Se recomienda que los clientes seleccionen la categoría Tecnología solo si las frases se encuentran estrictamente dentro del dominio tecnológico.

En la misma área de trabajo, puede crear proyectos para el mismo par de idiomas en distintas categorías. Custom Translator no permite la creación de un proyecto duplicado con el mismo par de idiomas y categoría. Si aplica una etiqueta al proyecto, podrá evitar esta restricción. No use las etiquetas a menos que esté creando sistemas de traducción para varios clientes, ya que la adición de una etiqueta única al proyecto se verá reflejada en el id. de categoría del proyecto.

Etiquetas de proyecto

Custom Translator le permite asignar una etiqueta de proyecto al proyecto. La etiqueta de proyecto distingue entre varios proyectos con el mismo par de idiomas y categoría. Como práctica recomendada, evite usar etiquetas de proyecto a menos que sea necesario.

La etiqueta de proyecto se usa como parte del id. de categoría. Si la etiqueta del proyecto se deja sin establecer o se establece de forma idéntica entre proyectos, los proyectos con la misma categoría y *diferentes* pares de idiomas compartirán el mismo id. de categoría. Este enfoque resulta ventajoso porque permite que usted o su cliente cambien entre idiomas cuando usen la Text Translator API sin preocuparse por que un id. de categoría sea único.

para cada proyecto.

Por ejemplo, si quisiera permitir traducciones en el dominio Tecnología del inglés al francés y del francés al inglés, crearía dos proyectos: uno para inglés -> francés y otro para francés -> inglés. Especificaría la misma categoría para ambos (Tecnología) y dejaría en blanco la etiqueta del proyecto. El id. de categoría para ambos proyectos coincidiría, por lo que podría consultar la API para las traducciones al inglés y al francés sin tener que modificar el valor del id. de categoría.

Si es un proveedor de servicios de lenguaje y quiere prestar servicios a varios clientes con diferentes modelos que conserven la misma categoría y par de idiomas, utilizar una etiqueta de proyecto para diferenciar clientes sería una decisión adecuada.

Pasos siguientes

- Obtenga información sobre [entrenamientos y modelos](#) para saber cómo crear un modelo de traducción de forma eficiente.

¿Qué son los documentos paralelos?

13/01/2020 • 4 minutes to read • [Edit Online](#)

Los documentos paralelos son pares de documentos en los que uno es la traducción del otro. Un documento en el par contiene frases en el idioma de origen y el otro documento contiene estas mismas frases traducidas al idioma de destino. No importa qué idioma está marcado como "origen" y qué idioma como "destino": un documento paralelo se puede usar para entrenar un sistema de traducción en cualquier dirección.

Requisitos

Necesita un mínimo de 10 000 frases paralelas alineadas para entrenar a un sistema. Esta limitación es una red de seguridad para asegurarse de que las frases paralelas contienen suficiente vocabulario único para entrenar correctamente un modelo de traducción. Como procedimiento recomendado, agregue continuamente más contenido paralelo y entrene el sistema de nuevo para mejorar la calidad de la traducción. Consulte [Alineación de frases](#).

Microsoft requiere que los documentos cargados en Custom Translator no infrinjan los derechos de propiedad intelectual o de copyright de terceros. Para más información, consulte los [términos de uso](#). Cargar un documento mediante el portal no modifica la propiedad intelectual del documento en sí.

Uso de documentos paralelos

El sistema usa los documentos paralelos para:

1. Conocer cómo se asignan normalmente las palabras, expresiones y frases entre los dos idiomas.
2. Saber cómo procesar el contexto adecuado según las frases que lo rodean. Puede que una palabra no se traduzca siempre exactamente como otra palabra del otro idioma.

Como procedimiento recomendado, asegúrese de que hay una correspondencia completa entre las frases en las versiones de los documentos de origen y de destino.

Si el proyecto es de un dominio (o categoría) específico, los documentos deben contener una terminología coherente con esa categoría. La calidad del sistema de traducción resultante depende del número de frases del conjunto de documentos y de la calidad de las mismas. Cuantos más ejemplos contengan los documentos con usos diversos de una palabra específica de la categoría, mejor realizará el sistema la traducción.

Los documentos que se cargan son privados para cada área de trabajo y se pueden usar en tantos proyectos o aprendizajes como desee. Las frases extraídas de los documentos se almacenan de forma independiente en el repositorio como archivos de texto Unicode sin formato y están disponibles para su eliminación cuando sea necesario. No use Custom Translator como repositorio de documentos ya que no podrá descargar los documentos que cargó previamente en el mismo formato que los cargó.

Pasos siguientes

- Aprenda a usar un [diccionario](#) en Custom Translator.

¿Qué es un diccionario?

13/01/2020 • 8 minutes to read • [Edit Online](#)

Un diccionario es un par alineado de documentos que especifica una lista de frases u oraciones y sus traducciones correspondientes. Use un diccionario en el entrenamiento cuando quiera que Microsoft Translator siempre traduzca todas las instancias de la frase u oración de origen de la forma que especifique el diccionario. A veces, los diccionarios se denominan glosarios o bases terminológicas. Puede pensar en el diccionario como un "copiar y reemplazar" a la fuerza para todos los términos incluidos. Además, el servicio Traductor personalizado de Microsoft crea y usa sus propios diccionarios de uso general para mejorar la calidad de su traducción. Sin embargo, los diccionarios proporcionados por el cliente tienen prioridad y serán los primeros en usarse al buscar palabras o frases.

Los diccionarios solo funcionan en los proyectos en los pares de idiomas que tengan detrás un modelo de red neuronal general de Microsoft totalmente compatible. [Consulte la lista completa de idiomas.](#)

Diccionario de frases

Al incluir un diccionario de frases durante el entrenamiento de un modelo, todas las palabras y frases incluidas se traducen de la forma especificada. El resto de la oración se traduce como de costumbre. Puede usar un diccionario de frases para especificar las frases que no se deben traducir si proporciona la misma frase sin traducir en los archivos de origen y de destino del diccionario.

Diccionario de oraciones

El diccionario de oraciones le permite especificar una traducción exacta de destino para una oración de origen. Para que exista una coincidencia con el diccionario de oraciones, la oración completa que se envíe debe coincidir con la entrada de origen del diccionario. Si solo coincide una parte de la oración, la entrada será una coincidencia. Cuando se detecta una coincidencia, se devuelve la entrada de destino del diccionario de oraciones.

Entrenamientos de solo diccionario

Puede entrenar un modelo usando solo los datos de un diccionario. Para ello, seleccione solo el documento de diccionario (o los documentos de diccionario) que quiere incluir y pulse en Crear modelo. Ya que se trata de un entrenamiento de solo diccionario, no hay un número mínimo de oraciones de aprendizaje necesarias. Normalmente, el modelo completará el entrenamiento mucho más rápido que un entrenamiento estándar. Los modelos resultantes utilizarán los modelos base de Microsoft para la traducción, con la adición de los diccionarios que ha agregado. No obtendrá un informe de pruebas.

NOTE

Custom Translator no alinea las oraciones en los archivos de diccionario, por lo que es importante que los documentos de diccionario incluyan el mismo número de frases u oraciones tanto de origen como de destino, y que estén perfectamente alineados.

Recomendaciones

- Los diccionarios no reemplazan al entrenamiento de modelos mediante datos de aprendizaje. Se recomienda evitarlos y dejar que el sistema aprenda de los datos de aprendizaje. Sin embargo, cuando las oraciones o los nombres compuestos deban representarse tal cual, se debe usar un diccionario.

- El diccionario de frases debe usarse con moderación. Por tanto, debe saber que cuando se reemplaza una frase dentro de una oración, el contexto de esa oración se pierde o se ve limitado para continuar con la traducción del resto de la oración. Como resultado, aunque la frase o palabra en la oración se traducirá según el diccionario proporcionado, la calidad general de la traducción de la oración a menudo se verá afectada.
- El diccionario de frases funciona bien para los nombres compuestos, como nombres de producto ("Microsoft SQL Server"), nombres propios ("Ciudad de Hamburgo") o las características del producto ("tabla dinámica"). No funciona igual de bien para los verbos o adjetivos, ya que normalmente se declinan en el idioma de origen o en el de destino. El procedimiento recomendado es agregar al diccionario de frases solamente nombres compuestos.
- Cuando se usa un diccionario de frases, las mayúsculas y los signos de puntuación son importantes. Las entradas del diccionario solo coincidirán con las palabras y frases de la oración de entrada que utilicen exactamente las mismas mayúsculas y minúsculas que se especifican en el archivo de diccionario de origen. Además, las traducciones reflejarán tanto las mayúsculas y minúsculas como los signos de puntuación proporcionados en el archivo de diccionario de destino. Por ejemplo, si ha entrenado un sistema de inglés a español que usa un diccionario de frases que especifica "US" en el archivo de código fuente y "EE. UU." en el archivo de destino. Cuando solicite la traducción de una frase que incluya la palabra "us" (no en mayúsculas), NO coincidirá con el diccionario. Sin embargo, si solicita la traducción de una frase que contiene la palabra "US" (en mayúsculas), coincidiría con el diccionario y la traducción contendría "EE. UU." Tenga en cuenta que es posible que las mayúsculas y minúsculas, y la puntuación de la traducción sean diferentes tanto de las especificadas en el archivo de destino del diccionario como de las del origen. Sigue las reglas del idioma de destino.
- Si una palabra aparece varias veces en un archivo de diccionario, el sistema usará siempre la última entrada proporcionada. Por consiguiente, el diccionario no debe contener varias traducciones de la misma palabra.

Pasos siguientes

- Obtenga más información en relación a las [directrices sobre los formatos de documento](#).

Guía sobre formatos y convenciones de nomenclatura para documentos

13/01/2020 • 3 minutes to read • [Edit Online](#)

Cualquier archivo que se utilice para la traducción personalizada debe tener al menos **cuatro** caracteres de longitud.

Esta tabla incluye todos los formatos de archivo compatibles que puede usar para crear el sistema de traducción:

FORMATO	EXTENSIONES	DESCRIPCIÓN
XLIFF	.XLF, .XLIFF	Formato de documento paralelo que se exporta desde los sistemas de memoria de traducción. Los idiomas utilizados se definen dentro del archivo.
TMX	.TMX	Formato de documento paralelo que se exporta desde los sistemas de memoria de traducción. Los idiomas utilizados se definen dentro del archivo.
ZIP	.ZIP	ZIP es un formato de archivo.
Locstudio	.LCL	Formato de Microsoft para los documentos paralelos
Microsoft Word	.DOCX	Documento de Microsoft Word
Adobe Acrobat	.PDF	Documento portátil de Adobe Acrobat
HTML	.HTML, .HTM	Documento HTML
Archivo de texto	.TXT	Archivos de texto codificados en UTF-16 o UTF-8. El nombre de archivo no debe contener caracteres japoneses.
Archivo de texto alineado	.ALIGN	La extensión <code>.ALIGN</code> es una extensión especial que puede usar si sabe que las oraciones del par de documentos están perfectamente alineadas. Si proporciona un archivo <code>.ALIGN</code> , Custom Translator no alineará las oraciones por usted.
Archivo de Excel	.XLSX	Archivo de Excel (2013 o posterior). La primera línea o fila de la hoja de cálculo debe ser el código de idioma.

Formatos de diccionario

Para los diccionarios, Traductor personalizado es compatible con todos los formatos de archivo que se admiten para los conjuntos de entrenamiento. Si está utilizando un diccionario de Excel, la primera línea o fila de la hoja de

cálculo debe contener códigos de idioma.

Formatos de archivo ZIP

Los documentos se pueden agrupar en un único archivo ZIP y cargarse. Custom Translator es compatible con los formatos de archivo ZIP (ZIP, GZ y TGZ).

Todos los documentos del archivo zip con la extensión TXT, HTM, HTML, PDF, DOCX o ALIGN deben seguir esta convención de nomenclatura:

{nombre del documento}_{código de idioma} donde {nombre del documento} es el nombre del documento y {código de idioma} es el identificador de idioma ISO (dos caracteres), que indica que el documento contiene oraciones en ese idioma. Debe haber un guion bajo () *antes del código de idioma*.

Por ejemplo, para cargar dos documentos paralelos dentro de un archivo ZIP para un sistema de inglés a español, los archivos deben tener los nombres "data_en" y "data_es".

No es necesario que los archivos de la memoria de traducción (TMX, XLF, XLIFF, LCL y XLSX) sigan la convención de nomenclatura de un idioma específico.

Pasos siguientes

- Obtenga información sobre los [proyectos](#) para crearlos y administrarlos.

Emparejamiento y alineación de oraciones en documentos paralelos

13/01/2020 • 3 minutes to read • [Edit Online](#)

Durante el entrenamiento, las oraciones disponibles en los documentos paralelos se emparejan o se alinean. Custom Translator notifica el número de oraciones que pudo emparejar con el nombre de Oraciones alineadas en cada uno de los conjuntos de datos.

Proceso de emparejamiento y alineación

Custom Translator aprende la traducción de las oraciones una a la vez. Primero, lee una oración de origen y, después, la traducción de esa misma oración en el idioma de destino. A continuación, alinea entre sí las palabras y frases presentes en ambas oraciones. Este proceso le permite asignar las palabras y frases en una oración a las palabras y frases equivalentes en la traducción de esa oración. La alineación intenta asegurarse de que el sistema se entrena con oraciones que se corresponden entre sí como traducciones.

Documentos alineados previamente

Si sabe que tiene documentos paralelos, puede invalidar la alineación de oraciones al proporcionar los archivos de texto alineados previamente. Puede extraer todas las oraciones de ambos documentos en un archivo de texto, organizar una oración por línea y cargarlas con la extensión `.align`. La extensión `.align` indica a Custom Translator que debe omitir la alineación de oraciones.

Para obtener mejores resultados, intente asegurarse de que hay una oración por línea en sus archivos. No incluya caracteres de nueva línea en una oración, ya que esto ocasionará alineaciones deficientes.

Número mínimo de frases sugerido

Para que un entrenamiento se realice correctamente, en la tabla siguiente se muestra el número mínimo de frases necesarias en cada tipo de documento. Esta limitación es una red de seguridad para asegurarse de que las frases paralelas contienen suficiente vocabulario único para entrenar correctamente un modelo de traducción. La directriz general consiste en tener más frases paralelas en el dominio de calidad de traducción humana y generar modelos de mayor calidad.

TIPO DE DOCUMENTO	NÚMERO MÍNIMO SUGERIDO DE FRASES	NÚMERO MÁXIMO DE FRASES
Cursos	10 000	No hay límite superior
Ajuste	5.000	2500
Prueba	5.000	2500
Diccionario	0	No hay límite superior

NOTE

- El entrenamiento no se iniciará y se producirá un error si no se cumple el número mínimo de 10 000 frases para el entrenamiento.
- La optimización y las pruebas son opcionales. Si no las proporciona, el sistema quitará un porcentaje adecuado del entrenamiento que se usará para la validación y las pruebas.
- Puede entrenar un modelo usando solo los datos de un diccionario. Consulte [Qué es el diccionario](#).

Pasos siguientes

- Aprenda a usar un [diccionario](#) en Custom Translator.

Filtrado de datos

13/01/2020 • 4 minutes to read • [Edit Online](#)

Al enviar documentos que se usarán para entrenar un sistema personalizado, dichos documentos se someten a una serie de pasos de procesamiento y filtrado para prepararse para el aprendizaje. Estos pasos se explican aquí.

Conocer el filtrado puede ayudarle a entender el recuento de oraciones que aparece en el traductor personalizado, así como los pasos que puede seguir para preparar los documentos para el entrenamiento de Traductor personalizado.

Alineación de frases

Si el documento no está en formato XLIFF, TMX o ALIGN, Traductor personalizado alinea las oraciones de los documentos de origen y de destino entre sí, oración por oración. Traductor no realiza la alineación del documento, sigue la nomenclatura de los documentos para buscar el documento coincidente del otro idioma. Dentro del documento, Traductor personalizado intenta encontrar la frase correspondiente en el otro idioma. Usa el marcado de documento, como etiquetas HTML incrustadas, para ayudar con la alineación.

Si ve una discrepancia importante entre el número de frases en el lado de origen y de destino de los documentos, es posible que el documento no fuese paralelo en primer lugar, o que no se pudiera alinear bien por otros motivos. Si el documento se empareja con una gran diferencia (> 10 %) de oraciones en cada lado, deberá asegurarse de que están realmente en paralelo. Traductor personalizado muestra una advertencia junto al documento si el recuento de frases es sospechosamente diferente.

Desduplicación

Traductor personalizado quita las frases que están presentes en los documentos de prueba y optimización de los datos de aprendizaje. La eliminación se realiza de forma dinámica dentro de la ejecución de aprendizaje, no en el paso de procesamiento de datos. Traductor personalizado le notifica el recuento de oraciones en la información general del proyecto antes de esa eliminación.

Filtro de longitud

- Quite las frases con solo una palabra a ambos lados.
- Quite las frases con más de 100 palabras a ambos lados. Las frases en chino, japonés o coreano están exentas.
- Quite las frases con menos de 3 caracteres. Las frases en chino, japonés o coreano están exentas.
- Quite las frases con más de 2000 caracteres en chino, japonés o coreano.
- Quite las frases con menos de un 1 % de caracteres alfabéticos.
- Quite las entradas de diccionario que contengan más de 50 palabras.

Espacio en blanco

- Reemplace cualquier secuencia de caracteres de espacios en blanco, incluyendo las tabulaciones y secuencias de Retorno de carro/Avance de línea con un carácter de espacio único.
- Quitar espacio inicial o final de la oración

Puntuación de final de frase

Reemplace varios caracteres de puntuación de final de frase con una sola instancia.

Normalización de caracteres japoneses

Convierta las letras y dígitos de ancho completo en caracteres de ancho medio.

Etiquetas XML sin escape

El filtrado transforma las etiquetas sin escape en etiquetas con escape:

- `<` se convierte en `&lt;`
- `>` se convierte en `&gt;`
- `&` se convierte en `&amp;`

Caracteres no válidos

Traductor personalizado quita frases que contengan el carácter Unicode U+FFFD. El carácter U+FFFD indica una conversión de codificación incorrecta.

Pasos siguientes

- [Entrenamiento de un modelo](#) en Traductor personalizado.

¿Qué son los entrenamientos y los modelos?

14/01/2020 • 9 minutes to read • [Edit Online](#)

Un modelo es el sistema que proporciona una traducción para pares de idiomas específicos. El resultado de un entrenamiento correcto es un modelo. Cuando se entrena un modelo, se requieren tres tipos de documentos mutuamente exclusivos: aprendizaje, ajuste y pruebas. También se puede proporcionar el tipo de documento de diccionario. Consulte [Alineación de frases](#).

Si solo se proporcionan datos de aprendizaje al poner en cola un entrenamiento, Custom Translator reunirá automáticamente los datos de pruebas y de ajuste. Usará un subconjunto aleatorio de frases de los documentos de aprendizaje y excluirá estas frases de los propios datos de aprendizaje.

Tipo de documento de aprendizaje para Custom Translator

Custom Translator usa los documentos incluidos en el conjunto de aprendizaje como base para generar el modelo. Durante la ejecución del entrenamiento, se alinean (o emparejan) las oraciones que están presentes en estos documentos. Puede tomarse libertades durante la creación del conjunto de documentos de aprendizaje. Puede incluir documentos que crea que son de importancia tangencial en un modelo. Vuelva a incluirlos en otro entrenamiento para ver el efecto que esto tiene en la puntuación [BLEU \(suplente de evaluación bilingüe\)](#). Siempre que el conjunto de ajuste y el de pruebas sean constantes, no dude en experimentar con la creación del conjunto de aprendizaje. Este enfoque es una forma eficaz de modificar la calidad de su sistema de traducción.

Puede ejecutar varios entrenamientos dentro de un proyecto y comparar las [puntuaciones BLEU](#) en todas las ejecuciones del entrenamiento. Cuando ejecute varios entrenamientos para compararlos, asegúrese de usar los mismos datos de prueba o de ajuste cada vez. También asegúrese de inspeccionar los resultados manualmente en la pestaña de [pruebas](#).

Tipo de documento de ajuste para Custom Translator

Custom Translator utiliza los documentos paralelos incluidos en este conjunto para optimizar el sistema de traducción y así obtener resultados óptimos.

Los datos de ajuste se usan durante el entrenamiento para ajustar todos los parámetros y las ponderaciones del sistema de traducción según los valores óptimos. Elija sus datos de ajuste cuidadosamente: deben ser representativos del contenido de los documentos que va a traducir en el futuro. Los datos de ajuste tienen una importante influencia en la calidad de las traducciones generadas. El ajuste permite al sistema de traducción proporcionar traducciones más cercanas a las muestras proporcionadas en los datos de ajuste. No se necesitan más que 2500 frases para los datos de ajuste. Para una calidad de traducción óptima, se recomienda que seleccione manualmente al conjunto de ajuste mediante la selección del grupo de oraciones más representativas.

Durante la creación del conjunto de ajuste, seleccione las oraciones que tengan una longitud representativa de las oraciones que espera traducir en el futuro. También debe elegir las oraciones que contengan palabras y frases que pretende traducir, con la distribución aproximada que espera encontrar en futuras traducciones. En la práctica, las oraciones con una longitud de entre 7 y 10 palabras generarán los mejores resultados, dado que estas oraciones contienen suficiente contexto para mostrar inflexiones y proporcionan una longitud de frase que es significativa sin ser demasiado compleja.

Una buena descripción del tipo de oraciones que debe usar en el conjunto de ajuste es la prosa: oraciones reales y fluidas. No celdas de una tabla, ni poemas, ni listas de cosas, ni solo puntuación o números: idioma común.

Si selecciona manualmente los datos de ajuste, en ellos no debe repetirse ninguna de las oraciones incluidas en los

datos de aprendizaje o de pruebas. Los datos de ajuste tienen un impacto importante en la calidad de las traducciones: elija atentamente las oraciones.

Si no está seguro de qué opciones elegir para los datos de ajuste, simplemente, seleccione los datos de aprendizaje y permita que Custom Translator seleccione los datos de ajuste automáticamente. Cuando Custom Translator elige automáticamente los datos de ajuste, usa un subconjunto aleatorio de oraciones tomadas de los documentos bilingües de aprendizaje y excluye a las oraciones que están en el material de aprendizaje.

Conjunto de datos de pruebas para Custom Translator

Los documentos paralelos que se incluyen en el conjunto de pruebas se utilizan para calcular la puntuación BLEU (suplente de evaluación bilingüe). Esta puntuación indica la calidad de su sistema de traducción. Realmente, esta puntuación le indica qué tanto se acercan las traducciones del sistema de traducción creado con este entrenamiento a las oraciones de referencia incluidas en el conjunto de datos de pruebas.

La puntuación BLEU es una medida de la delta entre la traducción automática y la traducción de referencia. Su valor varía entre 0 y 100. Una puntuación de 0 indica que en la traducción no aparece ni una sola palabra de la referencia. Una puntuación de 100 indica que la traducción automática coincide exactamente con la referencia: las mismas palabras están en exactamente la misma posición. La puntuación que recibe es la puntuación BLEU promedio de todas las oraciones en los datos de pruebas.

Los datos de prueba deben incluir documentos paralelos en los que las oraciones del idioma de destino sean las traducciones más deseables de las oraciones del idioma de origen correspondientes para el par de origen y destino. Es posible que quiera usar los mismos criterios que utilizó para crear los datos de ajuste. Sin embargo, los datos de prueba no influyen en la calidad del sistema de traducción. Se utiliza exclusivamente para generar la puntuación BLEU.

No se necesitan más de 2500 oraciones para los datos de pruebas. Cuando el sistema elige automáticamente al conjunto de pruebas, usa un subconjunto aleatorio de oraciones tomadas de los documentos bilingües de aprendizaje y excluye a las oraciones que están en el material de aprendizaje.

Puede ver las traducciones personalizadas del conjunto de pruebas y compararlas con las traducciones proporcionadas en el conjunto de pruebas. Para ello, vaya a la pestaña de pruebas dentro de un modelo.

¿Qué es una puntuación BLEU?

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

BLEU (Bilingual Evaluation Understudy o suplente de evaluación bilingüe) es una medida de las diferencias existentes entre una traducción automática y una o varias traducciones humanas de referencia de una misma frase de origen.

Proceso de puntuación

El algoritmo de BLEU compara expresiones consecutivas de la traducción automática con las expresiones consecutivas que encuentra en la traducción de referencia y cuenta el número de coincidencias en un modo ponderado. Estas coincidencias son independientes de la posición. Un mayor grado de coincidencia indica un mayor grado de similitud con la traducción de referencia y, por tanto, una puntuación más alta. La inteligibilidad y la corrección gramatical no se tienen en cuenta.

¿Cómo funciona BLEU?

La principal ventaja de BLEU es que se correlaciona bien con el criterio humano calculando el promedio de los errores de criterio de frases individuales de un corpus de prueba, en lugar de intentar averiguar el criterio humano exacto para cada frase.

[Aquí](#) puede encontrar un análisis más completo sobre la puntuación BLEU.

Los resultados de BLEU dependen en gran medida de la amplitud del dominio, la coherencia de los datos de prueba con los datos de aprendizaje y optimización, y la cantidad de datos disponible para el entrenamiento. Si los modelos se han entrenado en un dominio reducido, y los datos de aprendizaje son coherentes con los datos de prueba, es previsible una puntuación BLEU alta.

NOTE

Una comparación entre puntuaciones BLEU solo se puede justificar si los resultados de BLEU se comparan con el mismo conjunto de prueba, el mismo par de idiomas y el mismo motor de traducción automática. La puntuación BLEU de un conjunto de prueba diferente es obligatoriamente diferente.

Crear un proyecto

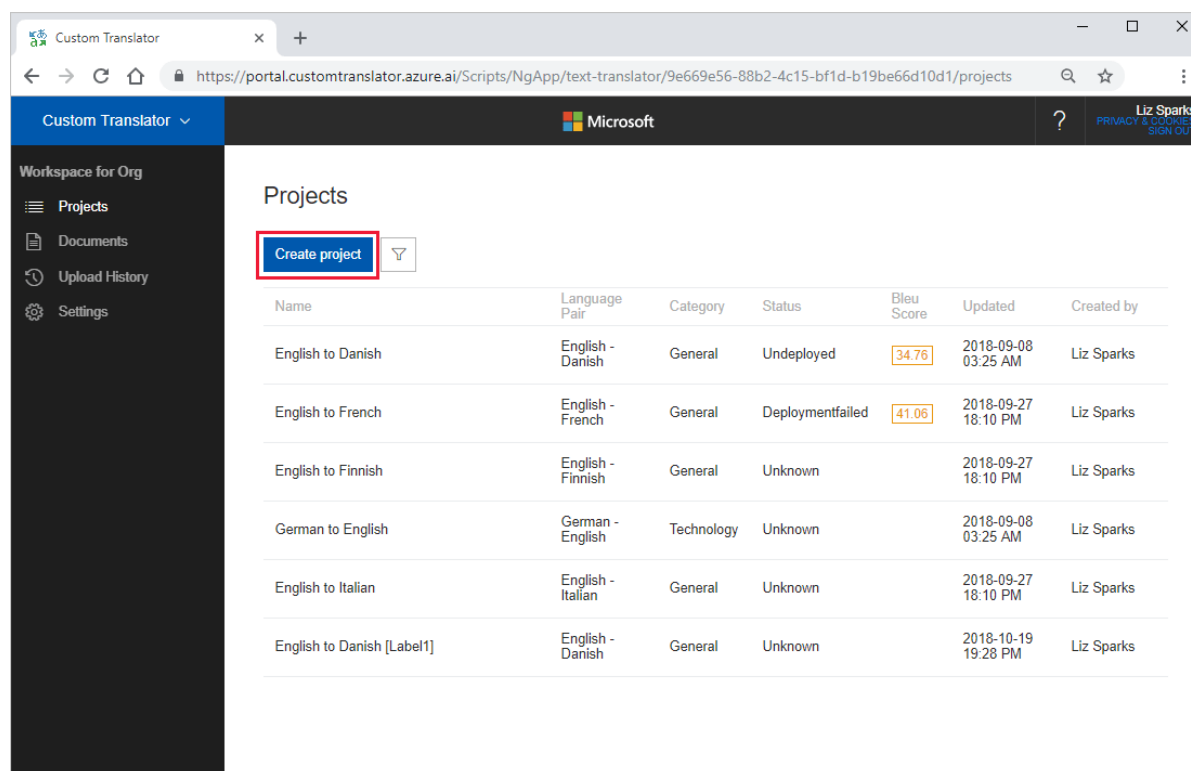
13/01/2020 • 4 minutes to read • [Edit Online](#)

Un proyecto es un contenedor para un modelo, documentos y pruebas. Cada proyecto incluye automáticamente todos los documentos que se cargan en esa área de trabajo con el par de idiomas correcto.

Crear un proyecto es el primer paso para crear un modelo.

Crear un proyecto:

1. En el portal de [Custom Translator](#), haga clic en Crear proyecto.



2. Escriba la siguiente información sobre el proyecto en el cuadro de diálogo:
 - a. Project name (Nombre del proyecto) (obligatorio): asigne al proyecto un nombre único y descriptivo. No es necesario mencionar los idiomas en el título.
 - b. Description: un breve resumen sobre el proyecto. Esta descripción no influye en el comportamiento de Custom Translator ni en el sistema personalizado resultante, pero puede ayudar a distinguir entre distintos proyectos.
 - c. Language pair (Par de idiomas) (obligatorio): seleccione los idiomas de origen y destino de la traducción.
 - d. Category (Categoría) (obligatorio): seleccione la categoría más adecuada para su proyecto. La categoría describe la terminología y el estilo de los documentos que se van a traducir.
 - e. Category description (Descripción de la categoría): use este campo para describir mejor el sector o la industria en concreto en los que está trabajando. Por ejemplo, si la categoría es "medicina", podría agregar un documento concreto, como cirugía o pediatría. Esta descripción no influye en el comportamiento de Custom Translator ni en el sistema personalizado resultante.
 - f. Project label (Etiqueta del proyecto): la [etiqueta de proyecto](#) distingue entre proyectos con el mismo par de

idiomas y categoría. Como práctica recomendada, use una etiqueta *solo* si piensa compilar varios proyectos para el mismo par de idiomas y la misma categoría y quiere tener acceso a estos proyectos con un id. de categoría distinto. No use este campo si está creando sistemas para una sola categoría. Las etiquetas de proyecto no son obligatorias y no ayudan a distinguir entre pares de idiomas. Puede usar la misma etiqueta para varios proyectos.

Create Project

Project name*

Name your project (max 256 chars)

Description

Add more information to your project (max 500 chars)

Language Pair*

Category*

Category descriptor

Add a category descriptor (max 75 chars)

Project label

Add a label to your project (max 20 chars)

Create

3. Haga clic en Crear.

Ver los detalles del proyecto

La página principal de Custom Translator muestra los primeros 10 proyectos en el área de trabajo. Muestra el nombre del proyecto, el par de idiomas, la categoría, el estado y la puntuación BLEU.

Después de seleccionar un proyecto, verá lo siguiente en la página del proyecto:

- **CategoryID** (Identificador de categoría): se crea al concatenar el identificador de área de trabajo, la etiqueta de proyecto y el código de categoría. Utilice el id. de categoría con Translator Text API para obtener traducciones personalizadas.
- **Botón Train** (Entrenar): use este botón para iniciar el [entrenamiento de un modelo](#).
- **Botón Add documents** (Agregar documentos): use este botón para [cargar documentos](#).
- **Botón Filter documents** (Filtrar documentos): use este botón para filtrar y buscar documentos específicos.

Projects > English to Danish

Category ID: 9e669e56-88b2-4c15-bf1d-b19be66d10d1-GENERAL

Language Pair: English - Danish

Category: General

[Edit project](#)

Data

Models

Create model

Upload files



6 Documents

<input type="checkbox"/>	Name	Document Pairing	Document Type	Language(s)	English Sentences	Danish Sentences
<input type="checkbox"/>	en_dan1	Parallel	Training	English - Danish	60,109	60,109
<input type="checkbox"/>	en_dan2	Parallel	Training	English - Danish	86,980	86,980
<input type="checkbox"/>	en_dan3	Parallel	Training	English - Danish	79,487	79,487
<input type="checkbox"/>	English to Danish Tuning 2	Parallel	Tuning	English - Danish	3,500	3,500
<input type="checkbox"/>	EU_15900000	Parallel	Tuning	Danish - English	4,355	4,355

Pasos siguientes

- Obtenga información sobre [cómo buscar, editar y eliminar el proyecto](#).
- Obtenga información sobre [cómo cargar un documento](#) para generar modelos de traducción.

Búsqueda, edición y eliminación de proyectos

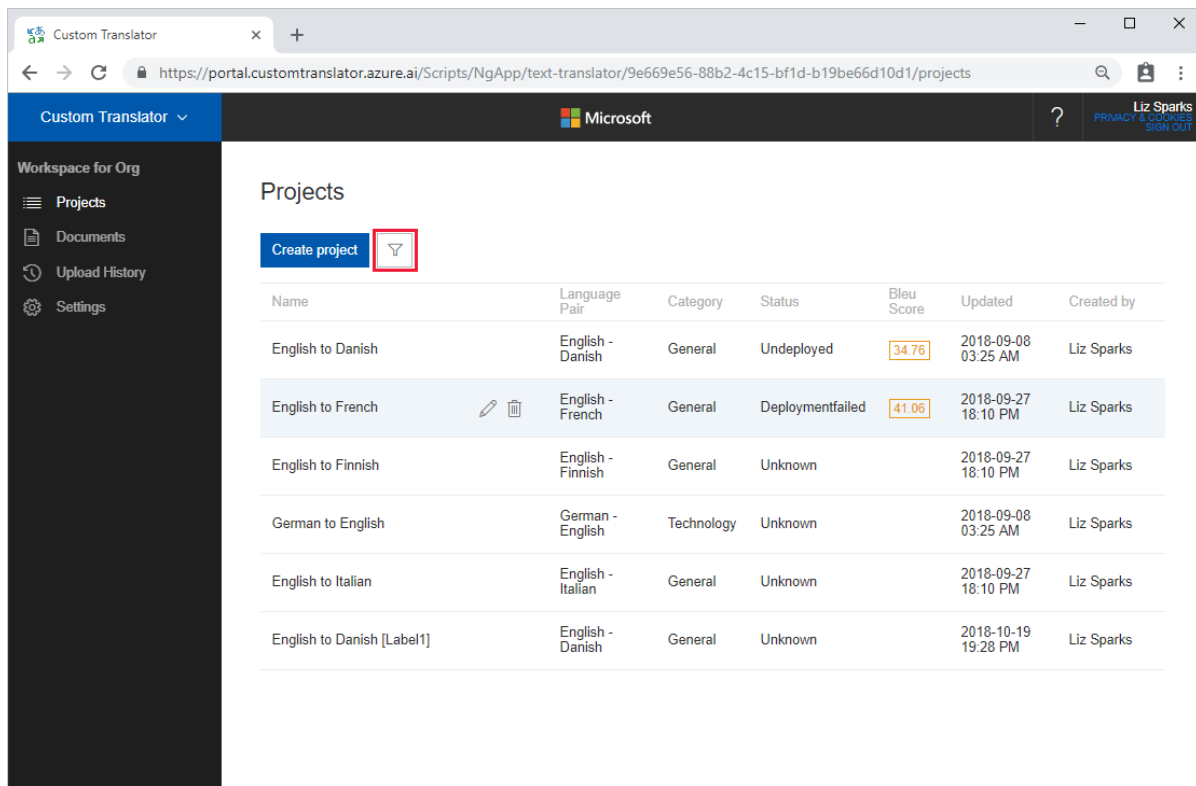
13/01/2020 • 2 minutes to read • [Edit Online](#)

Custom Translator proporciona varias formas de administrar los proyectos de manera eficaz. Puede crear muchos proyectos, realizar búsquedas según los criterios que elija y editar sus proyectos. También es posible eliminar proyectos de Custom Translator.

Búsqueda y filtrado de proyectos

La herramienta de filtro le permite buscar proyectos mediante condiciones de filtro diferentes. Se pueden utilizar como filtros el nombre del proyecto, el estado, los idiomas de origen y de destino y la categoría del proyecto.

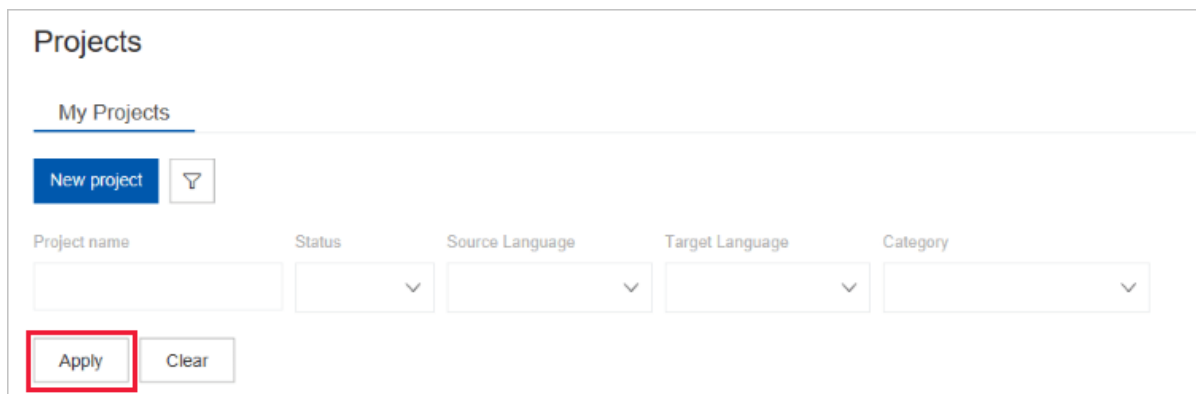
1. Haga clic en el botón de filtro.



The screenshot shows the 'Custom Translator' web interface. On the left is a sidebar with 'Workspace for Org' and options: 'Projects', 'Documents', 'Upload History', and 'Settings'. The main area is titled 'Projects' and contains a 'Create project' button and a filter icon (a funnel) highlighted with a red box. Below this is a table of projects:

Name	Language Pair	Category	Status	Bleu Score	Updated	Created by
English to Danish	English - Danish	General	Undeployed	34.76	2018-09-08 03:25 AM	Liz Sparks
English to French	English - French	General	Deployment failed	41.06	2018-09-27 18:10 PM	Liz Sparks
English to Finnish	English - Finnish	General	Unknown		2018-09-27 18:10 PM	Liz Sparks
German to English	German - English	Technology	Unknown		2018-09-08 03:25 AM	Liz Sparks
English to Italian	English - Italian	General	Unknown		2018-09-27 18:10 PM	Liz Sparks
English to Danish [Label1]	English - Danish	General	Unknown		2018-10-19 19:28 PM	Liz Sparks

2. Puede filtrar por cualquiera de los siguientes campos (o usarlos todos): nombre, estado, idioma de origen, idioma de destino y categoría.
3. Haga clic en Aplicar.



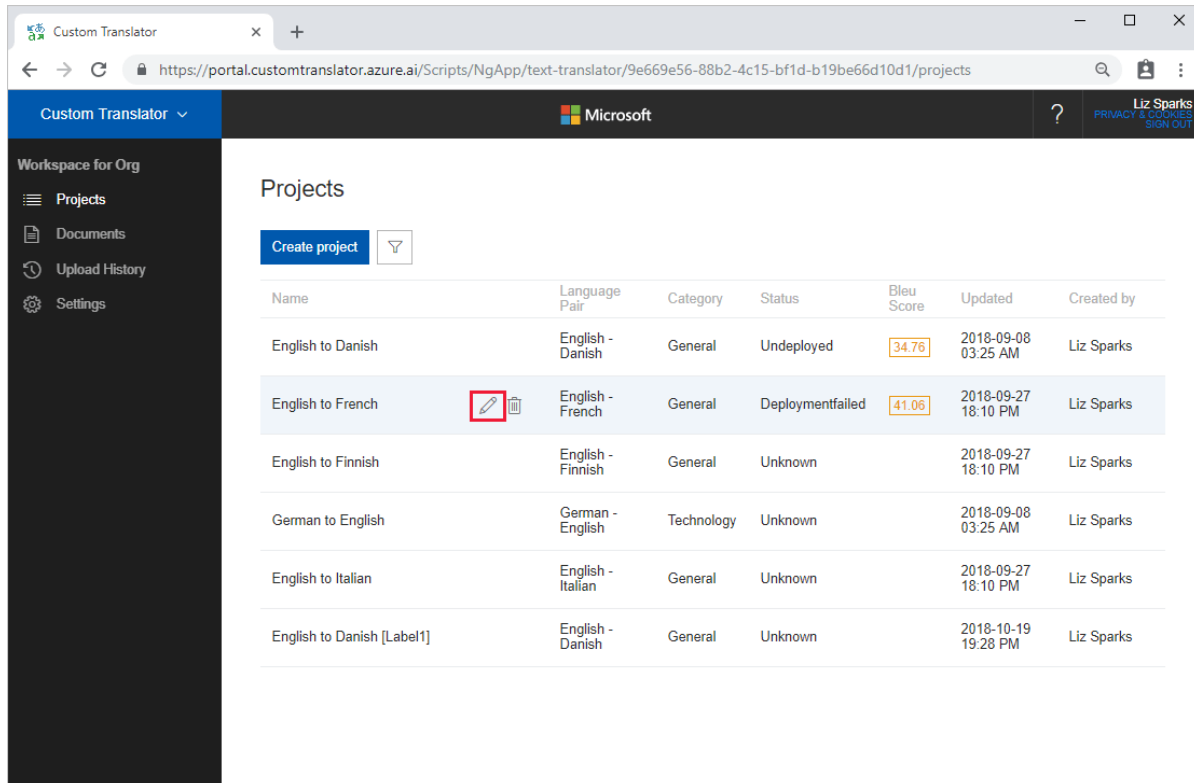
The screenshot shows the 'My Projects' filter form. It includes a 'New project' button and a filter icon. Below these are five input fields: 'Project name', 'Status', 'Source Language', 'Target Language', and 'Category'. At the bottom, there are two buttons: 'Apply' (highlighted with a red box) and 'Clear'.

4. Puede borrar el filtro para ver todos los proyectos si pulsa "Borrar".

Edición de un proyecto

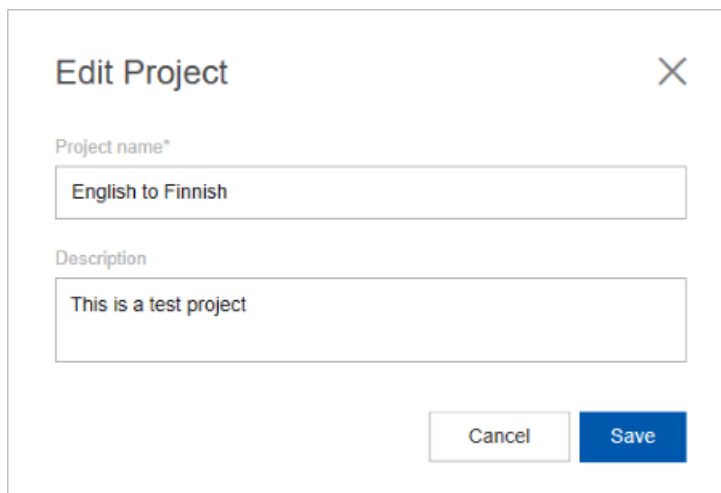
El traductor personalizado le ofrece la posibilidad de editar el nombre y la descripción de un proyecto. Otros metadatos del proyecto, como la categoría o los idiomas de origen y de destino no están disponibles para su edición. Los pasos siguientes describen cómo editar un proyecto.

1. Haga clic en el icono de lápiz que aparece cuando mantiene el mouse sobre un proyecto.



Name	Language Pair	Category	Status	Bleu Score	Updated	Created by
English to Danish	English - Danish	General	Undeployed	34.76	2018-09-08 03:25 AM	Liz Sparks
English to French	English - French	General	Deploymentfailed	41.06	2018-09-27 18:10 PM	Liz Sparks
English to Finnish	English - Finnish	General	Unknown		2018-09-27 18:10 PM	Liz Sparks
German to English	German - English	Technology	Unknown		2018-09-08 03:25 AM	Liz Sparks
English to Italian	English - Italian	General	Unknown		2018-09-27 18:10 PM	Liz Sparks
English to Danish [Label1]	English - Danish	General	Unknown		2018-10-19 19:28 PM	Liz Sparks

2. En el cuadro de diálogo, puede modificar el nombre o la descripción del proyecto, pero no puede modificar el par de idiomas, la categoría ni la etiqueta de proyecto.



Edit Project [Close]

Project name*

English to Finnish

Description

This is a test project

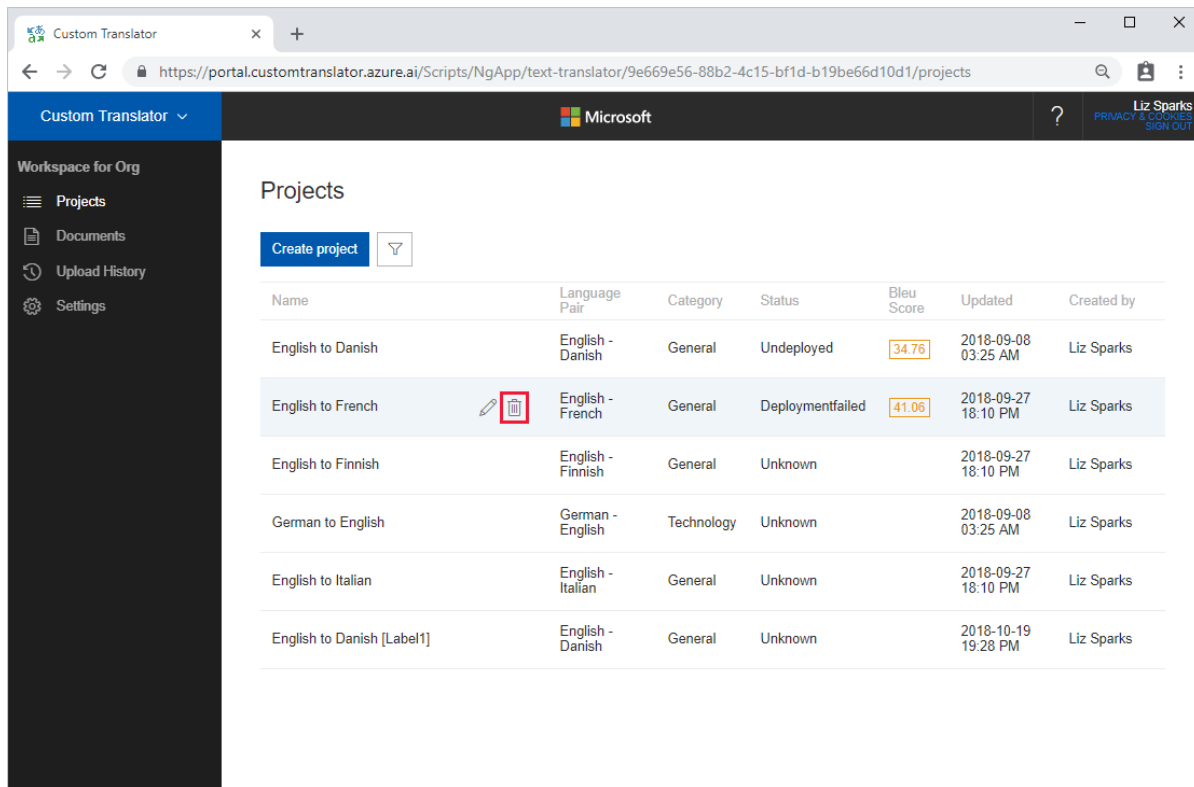
Cancel Save

3. Haga clic en el botón de filtro.

Eliminación de un proyecto

Puede eliminar un proyecto cuando ya no lo necesite. Los pasos siguientes describen cómo eliminar un proyecto.

1. Mantenga el mouse sobre cualquier registro de proyecto y haga clic en el icono de papelera.



Custom Translator

Microsoft

Workspace for Org

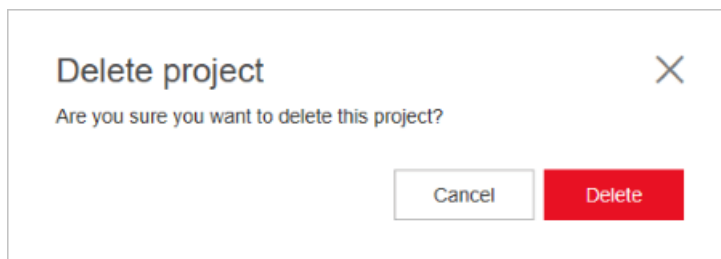
- Projects
- Documents
- Upload History
- Settings

Projects

Create project

Name	Language Pair	Category	Status	Bleu Score	Updated	Created by
English to Danish	English - Danish	General	Undeployed	34.76	2018-09-08 03:25 AM	Liz Sparks
English to French	English - French	General	Deployment failed	41.06	2018-09-27 18:10 PM	Liz Sparks
English to Finnish	English - Finnish	General	Unknown		2018-09-27 18:10 PM	Liz Sparks
German to English	German - English	Technology	Unknown		2018-09-08 03:25 AM	Liz Sparks
English to Italian	English - Italian	General	Unknown		2018-09-27 18:10 PM	Liz Sparks
English to Danish [Label1]	English - Danish	General	Unknown		2018-10-19 19:28 PM	Liz Sparks

2. Confirme la eliminación. Al eliminar un proyecto, se eliminarán todos los modelos que se crearon dentro de ese proyecto. La eliminación de un proyecto no afectará los documentos.



Delete project

Are you sure you want to delete this project?

Cancel Delete

Pasos siguientes

- Cargar documentos para comenzar a crear el modelo de traducción personalizada.

Cargar un documento

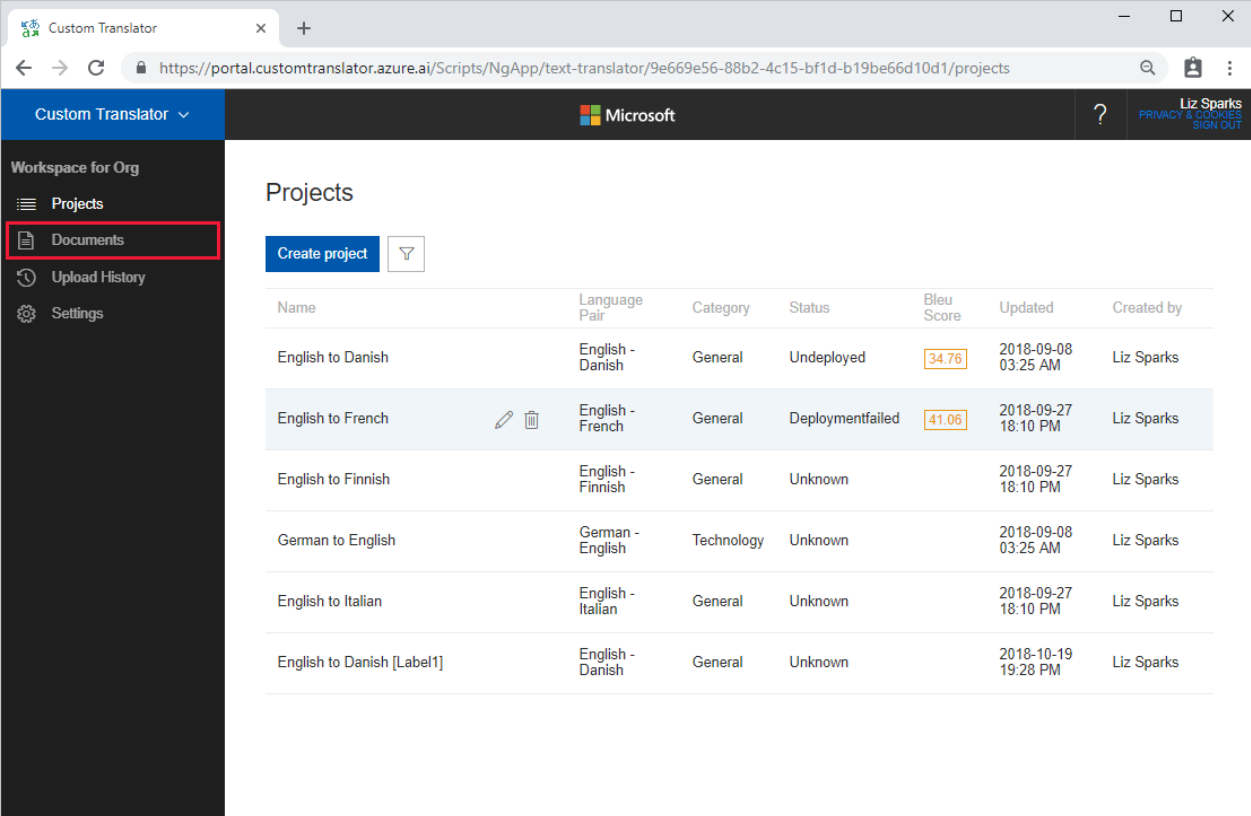
13/01/2020 • 5 minutes to read • [Edit Online](#)

En [Custom Translator](#), puede cargar documentos paralelos para entrenar los modelos de traducción. Los [documentos paralelos](#) son pares de documentos en los que uno es la traducción del otro. Un documento en el par contiene frases en el idioma de origen y el otro documento contiene estas mismas frases traducidas al idioma de destino.

Antes de cargar los documentos, revise la [Guía sobre formatos y convenciones de nomenclatura para documentos](#) para asegurarse de que el formato de archivo es compatible con Custom Translator.

¿Cómo se carga un documento?

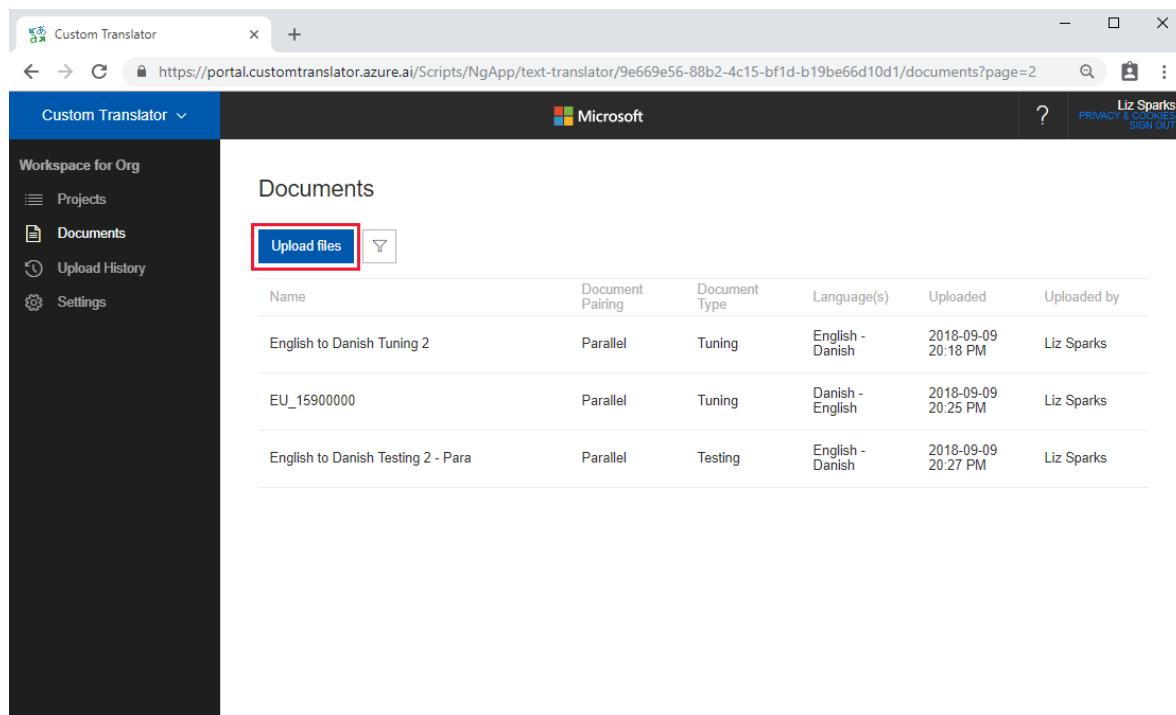
Desde el portal de [Custom Translator](#), haga clic en la pestaña "Documentos" para ir a la página de documentos.



The screenshot shows the Custom Translator portal interface. The sidebar on the left has a 'Documents' tab highlighted with a red box. The main content area is titled 'Projects' and contains a table of projects. The table has columns: Name, Language Pair, Category, Status, Bleu Score, Updated, and Created by. The 'English to French' project is highlighted in blue.

Name	Language Pair	Category	Status	Bleu Score	Updated	Created by
English to Danish	English - Danish	General	Undeployed	34.76	2018-09-08 03:25 AM	Liz Sparks
English to French	English - French	General	Deployment failed	41.06	2018-09-27 18:10 PM	Liz Sparks
English to Finnish	English - Finnish	General	Unknown		2018-09-27 18:10 PM	Liz Sparks
German to English	German - English	Technology	Unknown		2018-09-08 03:25 AM	Liz Sparks
English to Italian	English - Italian	General	Unknown		2018-09-27 18:10 PM	Liz Sparks
English to Danish [Label1]	English - Danish	General	Unknown		2018-10-19 19:28 PM	Liz Sparks

1. Haga clic en el botón Cargar archivos en la página de documentos.



2. En el cuadro de diálogo, rellene la información siguiente:

a. Tipo de documento:

- Aprendizaje: estos documentos se usarán para el conjunto de aprendizaje.
- Optimización: estos documentos se usarán para el conjunto de optimización.
- Prueba: estos documentos se usarán para el conjunto de pruebas.
- Diccionario de frases: estos documentos se usarán para el diccionario de frases.
- Diccionario de oraciones: estos documentos se usarán para el diccionario de oraciones.

b. Par de idiomas

c. Override document if exists (Reemplazar el documento si existe): active esta casilla si quiere sobrescribir los documentos existentes con el mismo nombre.

d. Rellene la sección correspondiente para los datos en paralelo o los datos combinados.

- Datos en paralelo:
 - Archivo de origen: seleccione el archivo en el idioma de origen del equipo local.
 - Archivo de destino: seleccione el archivo en el idioma de destino del equipo local.
 - Nombre del documento: solo se usa si está cargando archivos paralelos.
- Datos combinados:
 - Archivo combinado: seleccione el archivo combinado del equipo local. El archivo combinado tiene las oraciones tanto en el idioma de origen como en el de destino. La [convención de nomenclatura](#) es importante para los archivos combinados.

e. Haga clic en Cargar

Add Documents

*Maximum file size allowed is 100MB

Document Type:

Training

Language Pair

English -> German

☐ Override document if it exists

Parallel Data

Source (en) file:

Browse files...

1813n6_en.txt

Target (da) file:

Browse files...

1813n6_de.txt

.TXT|.HTML|.HTM|.PDF|.DOCX|.ALIGN file required.

Document Name:

New document

Archive or Combo File

Combo file:

Browse files...

.TMX|.XLF|.XLIFF|.LCL|.XLSX|.ZIP file required.

Cancel

Upload

- En este momento, estamos procesando sus documentos e intentando extraer las oraciones. Puede hacer clic en "Ver progreso de carga" para comprobar el estado de los documentos conforme se procesan.

Documents processing

Your files have been received and are being processed. It may take a while for large files to finish processing.

View upload progress

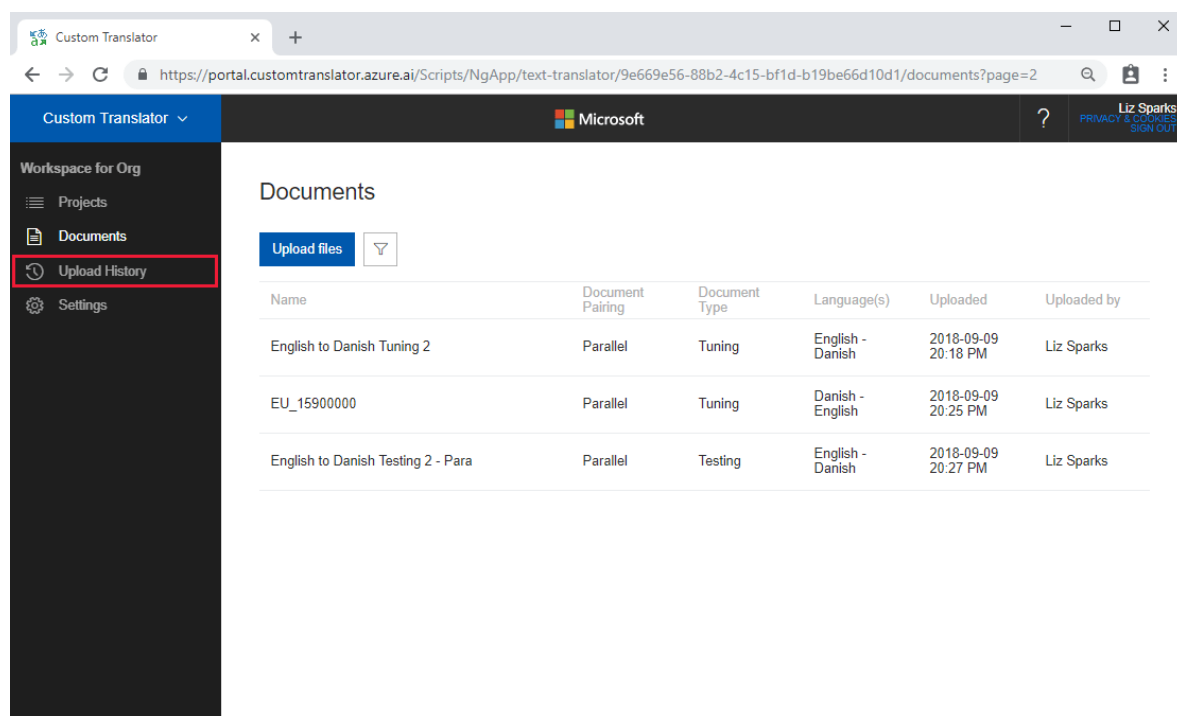
- En esta página se mostrarán el estado y los errores de cada archivo durante la carga. Puede ver los estado de carga anteriores en cualquier momento si hace clic en la pestaña "Historial de carga".

Upload History > English to Danish Testing 2 - Para			
Filter			
Name	Status	Language	Error
EU_15900000_enu_en.txt	✔ Succeeded	English	
EU_15900000_dan_da.txt	✔ Succeeded	Danish	

Ver el historial de carga

En la página del historial de carga puede ver los detalles de todas las cargas de documentos en el historial, como el tipo de documento, el par de idiomas, el estado de carga, etc.

- Desde el portal de [Custom Translator](#), haga clic en la pestaña del historial de carga para ver el historial.



- En esta página se muestra el estado de todas las cargas anteriores. Se muestran las cargas desde la más reciente hasta la más antigua. Para cada carga, se muestra el nombre del documento, el estado de la carga, la fecha de carga, el número de archivos cargados, el tipo de los archivos cargados y el par de idiomas del archivo.

Upload History						
Filter						
Name	Status	Upload Date	Number of Files	Type	Language Pair	Uploaded By
English to Danish Testing 2 - Para	✓ Succeeded	2018-09-09 20:27 PM	2	Testing	Danish, English	Liz Sparks
EU_15900000_tuning.zip	✓ Succeeded	2018-09-09 20:24 PM	2	Tuning	English, Danish	Liz Sparks
EU_15900000_Tuning.zip	❌ 1 error	2018-09-09 20:23 PM	1	Tuning		Liz Sparks
English to Danish Tuning 2	✓ Succeeded	2018-09-09 20:17 PM	2	Tuning	Danish, English	Liz Sparks
doc aug 30 single files end de	✓ Succeeded	2018-08-30 23:46 PM	2	Training	English, German	Liz Sparks
doc aug 30 single files end de	✓ Succeeded	2018-08-30 20:15 PM	2	Training	English, German	Liz Sparks

- Haga clic en cualquier registro del historial de carga. En la página de detalles del historial de carga puede ver los archivos cargados como parte del proceso de carga, el estado de carga de los archivos, el idioma del archivo y los mensajes de error (si hay algún error en la carga).

Pasos siguientes

- Use la [página de detalles del documento](#) para revisar la lista de las oraciones extraídas.
- [Cómo entrenar un modelo](#).

Ver detalles del documento

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

En la página de la lista de documentos se muestran los 10 primeros documentos en el área de trabajo. Para cada uno de los documentos, se muestra el nombre, emparejamiento, tipo, idioma, marca de tiempo de carga y la dirección de correo electrónico del usuario que ha cargado el documento.

Haga clic en un documento individual para ver la página de detalles del documento. La página de detalles del documento muestra la lista de las oraciones extraídas del documento.

- De forma predeterminada, está seleccionado el idioma de "origen" en el campo con el menú desplegable, pero puede cambiar la selección para ver las oraciones del idioma de destino.
- Se muestran 20 oraciones por página de forma predeterminada. Puede usar el control de paginación para desplazarse entre las páginas.

Documents > English to Danish Tuning 2

File:

English ▾

20 sentences

En

I am convinced that competitiveness will have greatly increased within 5-10 years and that we will have vehicles with no emissions.

This would be a decisive environmental victory.

In Amendment No 10, something of the utmost importance to all opponents of atomic energy is being put to the vote.

Let the Euratom Treaty be abrogated in its present form but its safety aspects incorporated into the Treaty on European Union.

If that amendment should actually be adopted tomorrow - as I hope - then it would probably be the first time in the history of this Parliament that an anti-atomic amendment reaches the conciliation procedure.

That would mean that, for the first time, the Council would have to come to terms with criticism and rejection of its atomic policy.

There was a similar amendment once before, but that one started from the extremely naive assumption that all Europe's atomic power stations would be shut down by the year 2002.

Anyone who realizes that 34 % of all the electrical power consumed in the EU comes from atomic power stations - in France, the figure is 70 % - will appreciate that this is not an option.

Also, the safety and health provisions must be retained at all costs -if only, of course, because eastward enlargement will bring certain States into the Union whose safety standards are absolutely different from our own.

In that case, of course, they would have to observe those safety provisions.

If all environmentally aware Members, across party boundaries and across national boundaries, too, can find it in their hearts to vote together in tomorrow's vote, then we have a chance to get to the conciliation procedure.

Eliminar un documento


El usuario debe ser propietario del área de trabajo para eliminar un documento. Además, si un modelo está usando el documento, ya sea durante el proceso de entrenamiento o durante el proceso de implementación, dicho documento no podrá eliminarse.


1. Vaya a la página del documento

2. Mantenga el mouse sobre cualquier registro de documento y haga clic en el icono de papelera.

Documents

Upload files



Name		Document Pairing	Document Type	Language(s)	Uploaded
docmay41		Parallel	Training	Russian - English	2018-05-05 01:50 AM
docmay41		Parallel	Training	Russian - English	2018-05-08 20:48 PM
docmay41		Parallel	Training	Russian - English	2018-05-08 21:23 PM
docmay41		Parallel	Training	Russian - English	2018-05-08 21:24 PM
EnglishToItalianParallel\TestEnglishToItalian		Parallel	Training	English - Italian	2018-07-17 23:01 PM
en_dan1		Parallel	Training	English - Danish	2018-07-17 23:17 PM
en_dan2		Parallel	Training	English - Danish	2018-07-18 00:33 AM

3. Confirme la eliminación.

Delete document

Are you sure you want to delete this document?

Cancel

Delete

Pasos siguientes

- Obtenga información sobre [cómo entrenar un modelo](#).

Entrenamiento de un modelo

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

Entrenar un modelo es un paso importante al crear un modelo de traducción, ya que sin ese entrenamiento el modelo no se puede generar. El entrenamiento se produce según los documentos que seleccione para ello.

Para entrenar un modelo:

1. Seleccione el proyecto en el que desea crear un modelo.
2. La pestaña Datos del proyecto mostrará todos los documentos pertinentes para el par de idiomas del proyecto. Seleccione manualmente los documentos que desea usar para entrenar el modelo. Puede seleccionar documentos de aprendizaje, optimización y prueba desde esta pantalla. También basta con seleccionar el conjunto de aprendizaje y que Custom Translator cree automáticamente los conjuntos de optimización y prueba.
 - Nombre del documento: el nombre asignado.
 - Emparejamiento: indica si se trata de un documento paralelo o monolingüe. En este momento, no se admiten los documentos monolingües para el entrenamiento.
 - Tipo de documento: puede ser de entrenamiento, ajuste, prueba o diccionario.
 - Par de idiomas: muestra los idiomas de origen y de destino para el proyecto.
 - Frases de origen: muestra el número de frases extraídas del archivo de origen.
 - Frases de destino: muestra el número de frases extraídas del archivo de destino.

Projects > English to Danish

Category ID: 9e669e56-88b2-4c15-bf1d-b19be66d10d1-GENERAL
Language Pair: English - Danish
Category: General
[Edit project](#)

Data

Models

Create model

Upload files

✓ 2 documents selected, 139596 training sentences selected

<input type="checkbox"/>	Name	Document Pairing	Document Type	Language(s)	English Sentences	Danish Sentences
<input checked="" type="checkbox"/>	en_dan1	Parallel	Training	English - Danish	60,109	60,109
<input type="checkbox"/>	en_dan2	Parallel	Training	English - Danish	86,980	86,980
<input checked="" type="checkbox"/>	en_dan3	Parallel	Training	English - Danish	79,487	79,487
<input type="checkbox"/>	English to Danish Tuning 2	Parallel	Tuning	English - Danish	3,500	3,500
<input type="checkbox"/>	EU_15900000	Parallel	Tuning	Danish - English	4,355	4,355
<input type="checkbox"/>	English to Danish Testing 2 - Para	Parallel	Testing	English - Danish	3,050	3,050

3. Haga clic en el botón Entrenar.
4. En el cuadro de diálogo, especifique un nombre para el modelo.
5. Haga clic en Entrenar modelo.

Train New Model

Model name*

Second model for EN to DAN

Cancel

Train model

6. Custom Translator enviará el entrenamiento y mostrará el estado del mismo en la pestaña de modelos.

Projects > English to Danish

Category ID: 9e669e56-88b2-4c15-bf1d-b19be66d10d1-GENERAL

Language Pair: English - Danish

Category: General

Edit project

Data

Models

5 models

Name	Status	Bleu Score	Baseline Bleu Score	Training	Dictionary	Tuning	Test	Model Action
English >> Danish 1st model	Undeployed	34.76	34.27	52,015	0	2,500	2,500	Deploy
English >> Danish 2nd model	Succeeded	34.75	34.52	52,015	0	2,500	2,500	Deploy
English >> Danish 3rd model	Succeeded	31	30.88	57,015	0	4,139	2,868	Deploy
Second model for EN to DAN	Succeeded	34.58	34.4	52,015	0	2,500	2,500	Deploy
Model 3 for English to Danish	Submitted			0	0	0	0	

NOTE

El traductor personalizado es compatible con 10 modelos de aprendizaje de un área de trabajo en cualquier momento dado.

Edición de un modelo

Puede editar un modelo mediante el vínculo Editar de la página Detalles del modelo.

1. Haga clic en el icono de lápiz.

Projects > English to Danish > English >> Danish 2nd model

Bleu score : 34.75

Baseline Bleu score : 34.52

English - Danish

Edit model

Training Details

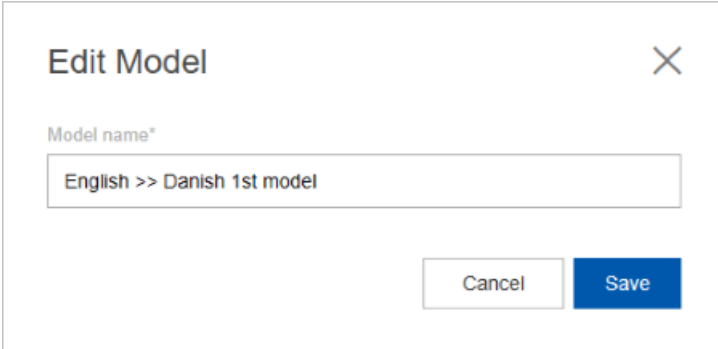
Test

1 documents

Name	Type	English Sentences	Danish Sentences	Aligned Sentences	Used Sentences
<input type="checkbox"/> en_dan1	Training - Parallel	60,109	60,109	57,015	52,015

2. En el cuadro de diálogo cambie,

- a. Nombre del modelo (obligatorio): asigne un nombre descriptivo al modelo.



The image shows a modal dialog box titled "Edit Model" with a close button (X) in the top right corner. Inside the dialog, there is a label "Model name*" followed by a text input field containing the text "English >> Danish 1st model". At the bottom right of the dialog, there are two buttons: "Cancel" and "Save".

3. Haga clic en Guardar.

Pasos siguientes

- Aprenda a [ver los detalles del modelo](#).

Ver detalles de los modelos

13/01/2020 • 7 minutes to read • [Edit Online](#)

La pestaña de modelos en el proyecto muestra todos los modelos del proyecto. Todos los modelos entrenados para ese proyecto se muestran en esta pestaña.

Estos detalles se muestran para todos los modelos del proyecto.

1. Nombre del modelo: muestra el nombre de un modelo determinado.
2. Estado: muestra el estado de un modelo determinado. El nuevo entrenamiento tendrá el estado Enviado hasta que se acepte. El estado cambiará a Procesamiento de datos mientras el servicio evalúa el contenido de los documentos. Una vez completada la evaluación de los documentos, el estado cambiará a En ejecución y podrá consultar el número de oraciones que forman parte del entrenamiento, incluidos los conjuntos de ajuste y de pruebas que se crearon automáticamente. A continuación se muestra una lista de estados del modelo que describe dichos estados.
 - Enviado: especifica que el back-end está procesando los documentos para ese modelo.
 - Entrenamiento en cola: especifica que el entrenamiento se pondrá en la cola del sistema de traducción automática para ese modelo.
 - En ejecución: especifica que el entrenamiento se está ejecutando en el sistema de traducción automática para ese modelo.
 - Correcto: especifica que el entrenamiento se realizó correctamente en el sistema de traducción automática y que hay un modelo disponible. En este estado, se muestra una puntuación BLEU para ese modelo.
 - Implementado: especifica que el modelo entrenado correctamente se envió al sistema de traducción automática para su implementación.
 - Anulando implementación: especifica que se está anulando la implementación del modelo implementado.
 - Implementación anulada: especifica que el proceso de anulación de la implementación de un modelo se ha completado correctamente.
 - Error de entrenamiento: especifica que no se pudo realizar el entrenamiento. Si se produce un error de entrenamiento, vuelva a ejecutar el trabajo de entrenamiento. Si el error continúa, póngase en contacto con nosotros. No elimine el modelo con errores.
 - Error de procesamiento de datos: especifica que hubo un error en el procesamiento de datos en uno o varios documentos que pertenecen al modelo.
 - Error de implementación: especifica que la implementación de modelos ha devuelto un error.
 - Borrador migrado: especifica que el modelo está en estado de borrador después de la migración del centro de conectividad al Traductor personalizado.
3. Puntuación BLEU: muestra la puntuación BLEU (suplente de evaluación bilingüe) del modelo, que indica la calidad de su sistema de traducción. Esta puntuación le indica qué tanto se acercan las traducciones del sistema de traducción creado con este entrenamiento a las oraciones de referencia incluidas en el conjunto de datos de prueba. La puntuación BLEU aparece si se ha completado correctamente el entrenamiento. Si el entrenamiento no se completa o devuelve errores, no verá ninguna puntuación BLEU.

- Recuento de oraciones de entrenamiento: muestra el número total de oraciones usadas como conjunto de entrenamiento.
- Recuento de oraciones de optimización: muestra el número total de oraciones usadas como conjunto de optimización.
- Recuento de oraciones de entrenamiento: muestra el número total de oraciones usadas como conjunto de pruebas.
- Recuento de oraciones monolingües: muestra el número total de oraciones usadas como conjunto monolingüe.
- Botón Implementar acción: para cada modelo entrenado correctamente, se muestra un botón "Implementar" si aún no está implementado. Si el modelo se ha implementado, se muestra un botón "Anular la implementación".
- Eliminar: puede usar este botón si desea eliminar el modelo. La eliminación de un modelo no elimina ninguno de los documentos que se utilizaron para crearlo.

Projects > English to Danish

Category ID: 9e669e56-88b2-4c15-bf1d-b19be66d10d1-GENERAL
Language Pair: English - Danish
Category: General
[Edit project](#)

Data
Models

5 models

Name	Status	Bleu Score	Baseline Bleu Score	Training	Dictionary	Tuning	Test	Model Action
English >> Danish 1st model	Undeployed	34.76	34.27	52,015	0	2,500	2,500	Deploy
English >> Danish 2nd model	Succeeded	34.75	34.52	52,015	0	2,500	2,500	Deploy
English >> Danish 3rd model	Succeeded	31	30.88	57,015	0	4,139	2,868	Deploy
Second model for EN to DAN	Succeeded	34.58	34.4	52,015	0	2,500	2,500	Deploy
Model 3 for English to Danish	Dataprocessing			0	0	0	0	

NOTE

Para comparar los entrenamientos consecutivos para los mismos sistemas, es importante mantener constantes los conjuntos de ajuste y de pruebas.

Ver detalles del entrenamiento de los modelos

Una vez completado el entrenamiento, puede revisar sus detalles desde la página de detalles. Seleccione un proyecto, busque y seleccione la pestaña de modelos y elija un modelo.

La página del modelo tiene dos pestañas: Detalles de entrenamiento y Prueba.

- Detalles de entrenamiento:** en esta pestaña se muestra la lista de documentos que se usaron en el entrenamiento:
 - Nombre del documento: en este campo se muestra el nombre del documento.
 - Tipo de documento: en este campo se muestra si este documento es paralelo o monolingüe.

- Recuento de oraciones en el idioma de origen: en este campo se muestra el número de oraciones que son parte del idioma de origen.
- Recuento de oraciones en el idioma de destino: en este campo se muestra el número de oraciones que son parte del idioma de destino.
- Oraciones alineadas: en este campo se muestra el número de oraciones que el Traductor personalizado ha alineado durante el proceso de alineación.
- Oraciones usadas: en este campo se muestra el número de oraciones que el Traductor personalizado ha usado durante el entrenamiento.

Projects > English to Danish > English >> Danish 3rd model

Bleu score : 31

Baseline Bleu score : 30.88

English - Danish

Edit model

Training Details

Test

3 documents

	Name	Type	English Sentences	Danish Sentences	Aligned Sentences	Used Sentences
<input type="checkbox"/>	en_dan1	Training - Parallel	60,109	60,109	57,015	57,015
<input type="checkbox"/>	EU_15900000	Tuning - Parallel	4,355	4,355	4,139	4,139
<input type="checkbox"/>	English to Danish Testing 2 - Para	Testing - Parallel	3,050	3,050	2,868	2,868

2. **Prueba:** en esta pestaña se muestran los detalles de prueba para un entrenamiento realizado correctamente.

Pasos siguientes

- Revise los [resultados de pruebas](#) y analice los resultados del entrenamiento.

Ver resultados de pruebas del sistema

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

Cuando el entrenamiento se realice correctamente, revise las pruebas del sistema para analizar los resultados del entrenamiento. Si está satisfecho con los resultados del entrenamiento, envíe una solicitud de implementación para el modelo entrenado.

Página de resultados de pruebas del sistema

Seleccione un proyecto y, a continuación, seleccione la pestaña de modelos de ese proyecto, busque el modelo que quiere usar y, por último, seleccione la pestaña de pruebas.

En la pestaña de pruebas se muestra lo siguiente:

1. **Resultados de pruebas del sistema:** el resultado del proceso de pruebas en los entrenamientos. El proceso de pruebas genera la puntuación BLEU.

Recuento de oraciones: cuántas oraciones paralelas se usaron en el conjunto de pruebas.

Puntuación BLEU: puntuación BLEU generada para un modelo tras la finalización del entrenamiento.

Estado: indica si se ha completado el proceso de prueba o si está en curso.

Projects > English to Danish > English >> Danish 1st model

Bleu score : 34.76

Baseline Bleu score : 34.27

English - Danish

[Edit model](#)

Training Details

Test

1 tests

Name	Sentence Count	Bleu Score	Status
System Test Results	2500	34.76	Complete

2. Haga clic en los resultados de pruebas del sistema y se mostrará la página de detalles del resultado de las pruebas. En esta página se muestra la traducción automática de las oraciones que formaban parte del conjunto de datos de prueba.

3. La tabla en la página de detalles del resultado de pruebas tiene dos columnas: una para cada idioma en el par. La columna del idioma de origen muestra la oración que se va a traducir. La columna para el idioma de destino contiene dos oraciones por cada fila.

Ref: esta oración es la traducción de referencia para la oración de origen tal como se especificó en el conjunto de datos de prueba.

MT: esta oración es la traducción automática de la oración de origen realizada por el modelo creado después de llevar a cabo el entrenamiento.

Projects > English to Danish > English >> Danish 1st model > System Test Results

English Sentences
Sentence Count: 2500

2,500 test results

English	Danish
The Serbs are not Milosevic, just as the Germans were not Hitler.	Ref: Serberne er ikke Milosevic, lige så lidt som tyskerne var Hitler. MT: Serberne er ikke Milosevic, ligesom tyskerne ikke var Hitler.
When we talk about identification and security services of this type, I hope that we do not limit ourselves and think that it is just a question of numbers.	Ref: Når vi taler om denne type identificerings- og sikkerhedstjenester, håber jeg, at vi ikke begrænser os og synes, at det bare drejer sig om tal. MT: Når vi taler om identifikations- og sikkerhedstjenester af denne type, håber jeg, at vi ikke begrænser os selv og mener, at det kun er et spørgsmål om tal.
It ought only to be permissible for animal feed to contain components of vegetable origin;	Ref: Hvorfor må foder ikke kun bestå af vegetabiliske bestanddele? MT: Det bør kun være tilladt, at foderstoffer indeholder bestanddele af vegetabilisk oprindelse;
Subject: Safe food National campaigns on food safety and health are being conducted in all 15 Member States during the winter and spring.	Ref: Om: Sikker mad I løbet af vinteren og foråret gennemføres der i alle EU's 15 medlemsstater kampagner om fødevaresikkerhed og sundhed. MT: Om: sikre fødevarer nationale kampagner for fødevaresikkerhed og-sundhed gennemføres i alle 15 medlemsstater i vinter-og forårsperioden.
This proposal is also currently before the Council and Parliament.	Ref: Dette forslag er også på nuværende tidspunkt til behandling i Rådet og Parlamentet. MT: Dette forslag er også i øjeblikket for Rådet og Parlamentet.
I would hope that their names can be added to whatever resolution comes out of today's debate.	Ref: Jeg håber, at deres navne kan blive tilføjet listen over underskrivere af det forslag til beslutning, som måtte blive resultatet af dagens forhandling. MT: Jeg håber, at man kan tilføje deres navne til enhver beslutning, der kommer fra dagens forhandling.

Descarga de la prueba

Haga clic en el vínculo Download Translations (Descargar traducciones) para descargar un archivo zip. El archivo zip contiene las traducciones automáticas de las oraciones de origen en el conjunto de datos de pruebas.


Projects > English to Danish > English >> Danish 1st model

Bleu score : 34.76
Baseline Bleu score : 34.27
English - Danish
[Edit model](#)

Training Details

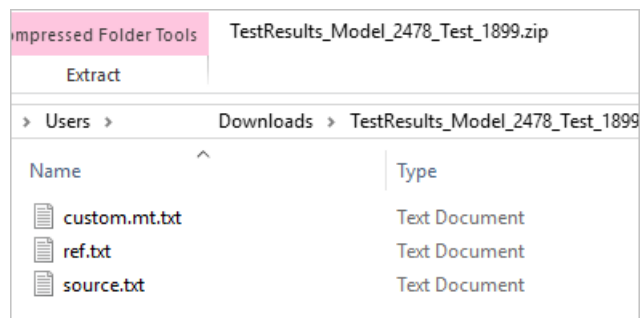
Test

1 tests

Name		Sentence Count	Bleu Score	Status
System Test Results		2500	34.76	Complete

Este archivo zip descargado contiene tres archivos.

1. **custom.mt.txt:** contiene la traducción automática de las oraciones del idioma de origen al idioma de destino que realizó el modelo entrenado con los datos del usuario.
2. **ref.txt:** contiene las traducciones de las oraciones del idioma de origen proporcionadas por el usuario en el idioma de destino.
3. **source.txt:** contiene las oraciones en el idioma de origen.



Implementar un modelo

Para solicitar una implementación, haga lo siguiente:

1. Seleccione un proyecto y vaya a la pestaña de modelos.
2. Para cada modelo entrenado correctamente, se muestra un botón "Implementar" si aún no se implementa.

Projects > English to Danish								
Category ID: 9e669e56-88b2-4c15-bf1d-b19be66d10d1-GENERAL								
Language Pair: English - Danish								
Category: General								
Edit project								
Data Models								
5 models								
Name	Status	Bleu Score	Baseline Bleu Score	Training	Dictionary	Tuning	Test	Model Action
English >> Danish 1st model	Undeployed	34.76	34.27	52,015	0	2,500	2,500	Deploy
English >> Danish 2nd model	Succeeded	34.75	34.52	52,015	0	2,500	2,500	Deploy
English >> Danish 3rd model	Succeeded	31	30.88	57,015	0	4,139	2,868	Deploy
Second model for EN to DAN	Succeeded	34.58	34.4	52,015	0	2,500	2,500	Deploy
Model 3 for English to Danish	Dataprocessing			0	0	0	0	

3. Haga clic en Implementar.
4. Seleccione **Implementado** para las regiones donde quiere que se implemente el modelo y haga clic en Guardar. Puede seleccionar **Implementado** para varias regiones.

Deploy or undeploy model

Currently all models will be deployed to all regions; however in a future release your model will only be deployed in the regions you have selected.

North America: Undeployed

Europe: Undeployed

Asia Pacific: Undeployed

Cancel Save

5. Puede ver el estado del modelo en la columna "Estado".

NOTE

Traductor personalizado es compatible con 10 modelos implementados dentro de un área de trabajo en cualquier momento dado.

Actualizar la configuración de implementación

Para actualizar la configuración de implementación:

1. Seleccione un proyecto y vaya a la pestaña **Modelos**.
2. En el caso de un modelo implementado correctamente, se muestra un botón **Actualizar**.

Projects > English to Danish

Category ID: 9e669e56-88b2-4c15-bf1d-b19be66d10d1-GENERAL
Language Pair: English - Danish
Category: General
[Edit project](#)

Data		Models						
Name	Status	Bleu Score	Baseline Bleu Score	Training	Dictionary	Tuning	Test	Model Action
English >> Danish 1st model	Deployed	34.76	34.27	52,015	0	2,500	2,500	Update
English >> Danish 2nd model	Succeeded	34.75	34.52	52,015	0	2,500	2,500	
English >> Danish 3rd model	Succeeded	31	30.88	57,015	0	4,139	2,868	
Second model for EN to DAN	Succeeded	34.58	34.4	52,015	0	2,500	2,500	
Model 3 for English to Danish	Trainingfailed			52,421	0	2,500	2,500	

3. Seleccione **Actualizar**.
4. Seleccione **Implementado** o **No implementado** para las regiones donde quiere que se implemente el modelo o se anule su implementación y, a continuación, haga clic en **Guardar**.

Deploy or undeploy model

Currently all models will be deployed to all regions; however in a future release your model will only be deployed in the regions you have selected.

North America: Deployed

Europe: Deployed

Asia Pacific: Deployed

Cancel Save

NOTE

Si selecciona **No implementado** para todas las regiones, se anula la implementación del modelo en todas las regiones y se coloca en un estado no implementado. Ahora está deshabilitado para su uso.

Pasos siguientes

- Comience a utilizar el modelo de traducción personalizado implementado mediante [Microsoft Translator Text API V3](#).
- Obtenga información sobre [cómo administrar la configuración](#) para compartir el área de trabajo y administrar la clave de suscripción.
- Obtenga información sobre [cómo migrar áreas de trabajo y proyectos](#) desde [Microsoft Translator Hub](#)

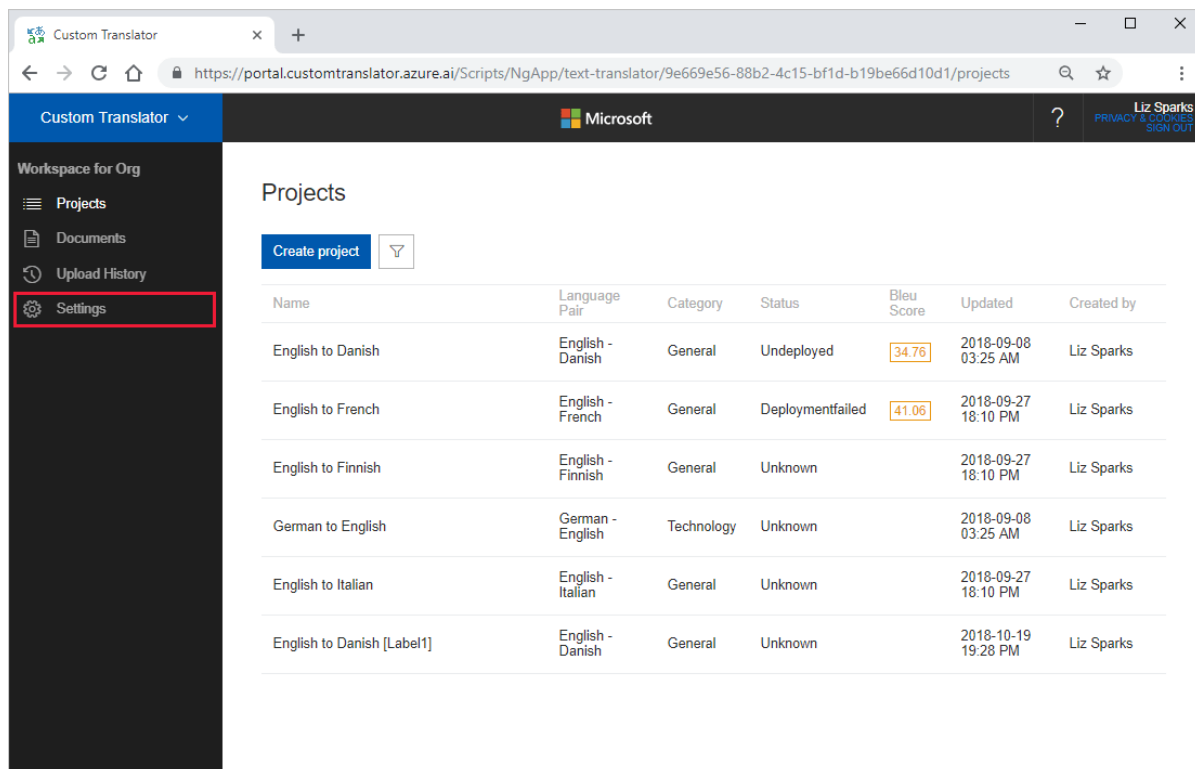
Cómo administrar la configuración

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

En la página de configuración de Custom Translator, puede crear una nueva área de trabajo, compartir el área de trabajo y agregar o modificar su clave de suscripción de Microsoft Translation.

Para acceder a la página de configuración, realice lo siguiente:

1. Inicie sesión en el portal de [Custom Translator](#).
2. En el portal de Custom Translator, haga clic en el icono de engranaje de la barra lateral.



Asociación de la suscripción de Microsoft Translator

Debe tener una clave de suscripción de Microsoft Translator Text API asociada con el área de trabajo para entrenar o implementar modelos.

Si no tiene una suscripción, siga los pasos a continuación:

1. Suscríbase a Microsoft Translator Text API. En este artículo se muestra cómo suscribirse a Microsoft Translator Text API.
2. Apunte la clave para la suscripción a Translator Text. Tanto la clave Key1 como la clave Key2 son aceptables.
3. Navegue de vuelta al portal de Custom Translator.

Adición de una clave existente

1. Vaya a la página "Configuración" de su área de trabajo.
2. Haga clic en Agregar clave.

Subscription Key

Your Microsoft Translator Text API subscription key is used for billing. Don't have a key? [Create one here.](#)

+ Add key

3. En el cuadro de diálogo, escriba la clave de la suscripción del traductor y, a continuación, haga clic en el botón "Agregar".

Add Existing Key



Paste your key(s) below

Cancel

Add

4. Después de agregar una clave, puede modificarla o eliminarla en cualquier momento.

Subscription Key

Your Microsoft Translator Text API subscription key is used for billing.

HubTextStandardS3: *****16ee4

Change key

Delete key

Administración del área de trabajo

Un área de trabajo es un área de trabajo para crear y compilar el sistema de traducción personalizada. Un área de trabajo puede contener varios proyectos, modelos y documentos.

Si una parte distinta del trabajo debe compartirse con personas distintas, puede resultar útil crear varias áreas de trabajo.

Crear un área de trabajo

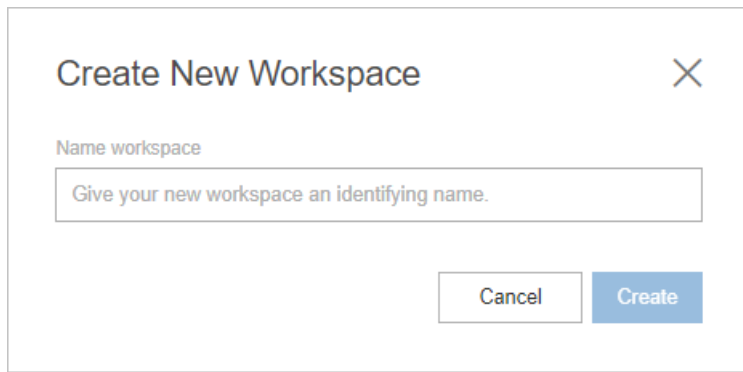
1. Vaya a la página "Configuración" del área de trabajo.
2. Haga clic en el botón "Nueva área de trabajo" de la sección "Crear área de trabajo nueva".

Create New Workspace

Workspaces are a place for your projects, models, and documents. You can share a workspace with your teammates. If different parts of your work need to be shared with different people, then creating multiple workspaces may be useful.

+ New workspace

3. En el cuadro de diálogo, escriba el nombre de la nueva área de trabajo.
4. Haga clic en "Crear".

A dialog box titled "Create New Workspace" with a close button (X) in the top right corner. Below the title is a label "Name workspace" and a text input field containing the placeholder text "Give your new workspace an identifying name.". At the bottom of the dialog are two buttons: "Cancel" and "Create".

Create New Workspace

Name workspace

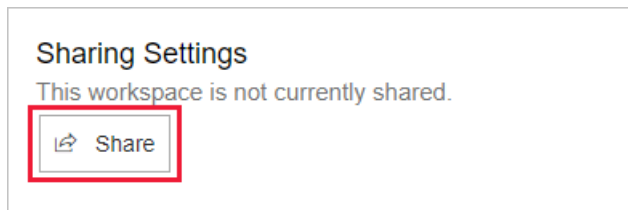
Give your new workspace an identifying name.

Cancel Create

Compartir el área de trabajo

En Custom Translator, puede compartir el área de trabajo con otros usuarios si una parte distinta del trabajo debe compartirse con personas diferentes.

1. Vaya a la página "Configuración" del área de trabajo.
2. Haga clic en el botón "Compartir" de la sección "Configuración de uso compartido".

A dialog box titled "Sharing Settings". Below the title is the text "This workspace is not currently shared.". At the bottom is a button with a share icon and the text "Share", which is highlighted with a red rectangular box.

Sharing Settings

This workspace is not currently shared.

Share

3. En el cuadro de diálogo, escriba una lista separada por comas de direcciones de correo electrónico con las que quiera compartir esta área de trabajo. Asegúrese de compartirla con la dirección de correo electrónico que la persona usa para iniciar sesión en Custom Translator. A continuación, seleccione el nivel de permisos de uso compartido adecuado.
4. Si el área de trabajo todavía tiene el nombre predeterminado "Mi área de trabajo", será necesario cambiarlo antes de compartir el área de trabajo.
5. Haga clic en "Guardar".

Permisos de uso compartido

1. **Lector:** un lector en el área de trabajo podrá ver toda la información del área de trabajo.
2. **Editor:** un editor en el área de trabajo podrá agregar documentos, entrenar modelos y eliminar documentos y proyectos. Puede agregar una clave de suscripción, pero no puede modificar con quién se comparte el área de trabajo, no puede eliminar el área de trabajo ni cambiarle el nombre.
3. **Propietario:** un propietario tiene permisos totales en el área de trabajo.


Cambiar los permisos de uso compartido

Cuando se comparte un área de trabajo, en la sección "Configuración de uso compartido" se muestran todas las direcciones de correo electrónico con las que se comparte esta área de trabajo. Puede cambiar los permisos de uso compartido existentes para cada dirección de correo electrónico si tiene acceso de propietario al área de trabajo.

1. En la sección "Configuración de uso compartido", se muestra un menú desplegable para cada correo electrónico donde se muestra el nivel de permisos actual.
2. Haga clic en el menú desplegable y seleccione el nuevo nivel de permisos que quiere asignarle a dicho correo electrónico.

Sharing Settings

This workspace is currently shared.

 Add people



Timmy.Forest@contoso.com

Editor ▼



Bret.Clegg@contoso.com

Reader ▼



Pasos siguientes

- Obtenga información sobre [cómo migrar áreas de trabajo y proyectos](#) desde [Microsoft Translator Hub](#)

Migración del área de trabajo y los proyectos de Microsoft Translator Hub a Custom Translator

13/01/2020 • 12 minutes to read • [Edit Online](#)

Puede migrar fácilmente el área de trabajo y los proyectos de [Microsoft Translator Hub](#) a Traductor personalizado. La migración se inicia desde Microsoft Hub al seleccionar un área de trabajo o proyecto; a continuación, se selecciona un área de trabajo en Traductor personalizado y, a continuación, se seleccionan los entrenamientos que desea transferir. Después de iniciar la migración, se transferirán las opciones de configuración de entrenamiento seleccionadas con todos los documentos pertinentes. Se entrenan los modelos implementados y se pueden implementar automáticamente tras la finalización.

Estas acciones se realizan durante la migración:

- Se transfieren los nombres de todos los documentos y las definiciones del proyecto con el prefijo "hub_" antes del nombre. La prueba generada automáticamente y los datos de optimización se llamarán `hub_systemtune_<modelid>` o `hub_systemtest_<modelid>`.
- Los entrenamientos que estaban en estado implementado en el momento de la migración se entrenarán automáticamente con los documentos de entrenamiento del centro. Este entrenamiento no se cargará en la suscripción. Si se ha seleccionado implementación automática para la migración, el modelo entrenado se implementará tras la finalización. Se aplicarán cargos de hospedaje normales.
- Los entrenamientos migrados que no estaban en estado implementado se colocan en estado de borrador migrado. En este estado, tendrá la opción de entrenar un modelo con la definición migrada, pero se aplicarán cargos de entrenamiento normales.
- En cualquier momento, la puntuación BLEU migrada desde el entrenamiento del centro se puede encontrar en la página TrainingDetails del modelo en el encabezado "Puntuación BLEU del centro de MT".

NOTE

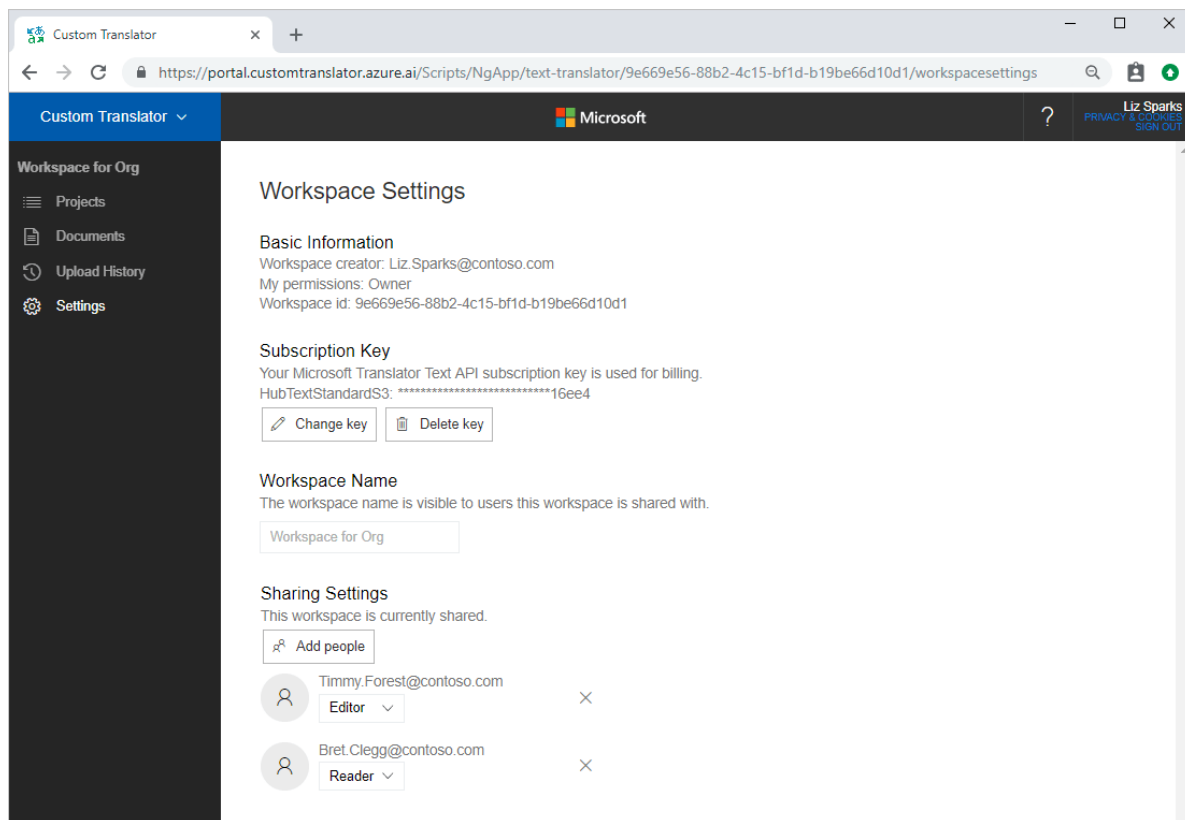
Para que un entrenamiento se realice correctamente, el Traductor personalizado necesita un mínimo de 10 000 frases extraídas únicas. El Traductor personalizado no puede realizar un entrenamiento con menos del [mínimo sugerido](#).

Búsqueda del identificador de área de trabajo de Traductor personalizado

Para migrar un área de trabajo de [Microsoft Translator Hub](#) necesita el identificador del área de trabajo de destino en Traductor personalizado. El área de trabajo de destino en Custom Translator es el lugar al que se migrarán todas las áreas de trabajo y proyectos de Microsoft Translator Hub.

Puede encontrar el identificador del área de trabajo de destino en la página de configuración de Traductor personalizado:

1. Vaya a la página "Configuración" en el portal de Custom Translator.
2. Encontrará el identificador del área de trabajo en la sección de información básica.



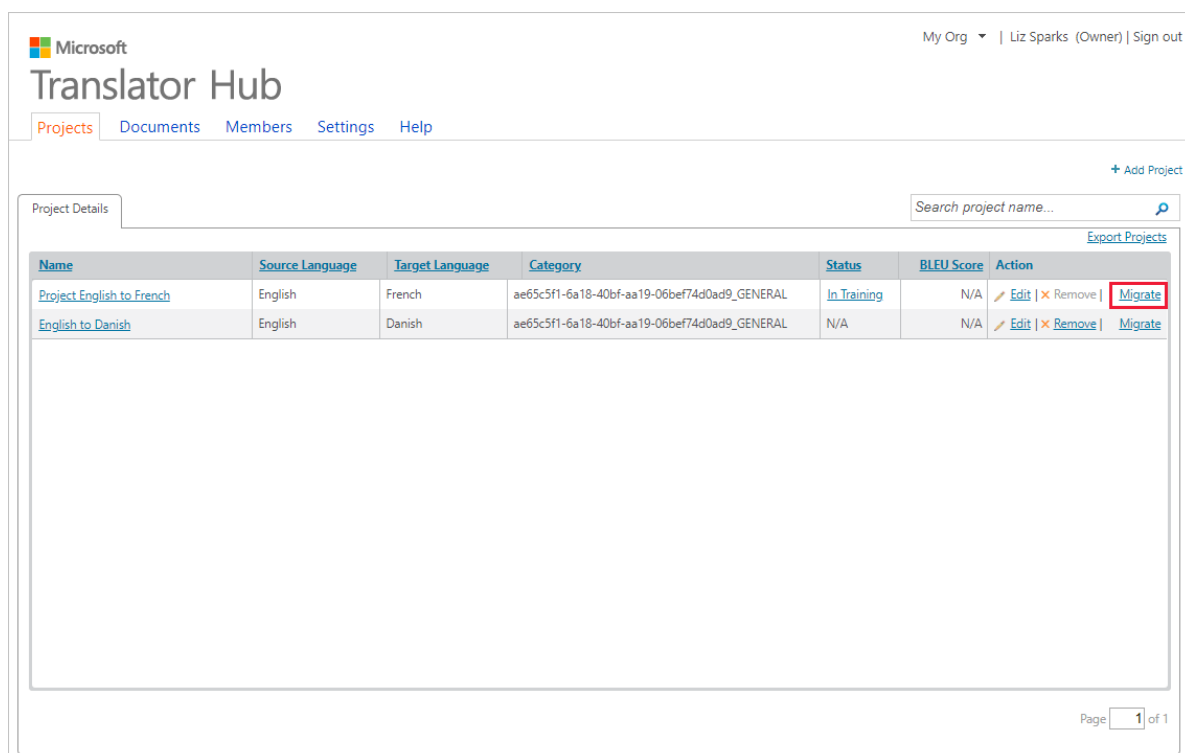
3. Conserve el identificador del área de trabajo de destino para consultarlo durante el proceso de migración.

Migración de un proyecto

Si desea migrar los proyectos de forma selectiva, Microsoft Translator Hub le ofrece esa posibilidad.

Para migrar un proyecto:

1. Inicie sesión en Microsoft Translator Hub.
2. Vaya a la página "Proyectos".
3. Haga clic en el vínculo "Migrar" del proyecto adecuado.




4. Al presionar el vínculo de migración, aparecerá un formulario que le permite:
 - Especificar el área de trabajo que desea transferir a Traductor personalizado
 - Indicar si desea transferir todos los entrenamientos con entrenamientos correctos o solo los entrenamientos implementados. De forma predeterminada, se transferirán todos los entrenamientos correctos.
 - Indicar si desea que los entrenamientos se implementen automáticamente cuando se complete el entrenamiento. De forma predeterminada, el entrenamiento no se implementará automáticamente cuando se complete.
5. Haga clic en "Enviar solicitud".

Migración de un área de trabajo

Además de migrar un único proyecto, también puede migrar todos los proyectos con entrenamientos correctos de un área de trabajo. Esto hará que cada proyecto del área de trabajo se evalúe como si se hubiera presionado el vínculo de migración. Esta característica es adecuada para los usuarios con muchos proyectos que desean migrar todos a Traductor personalizado con la misma configuración. La migración de un área de trabajo se puede iniciar en la página de configuración de Translator Hub.

Para migrar un área de trabajo:

1. Inicie sesión en Microsoft Translator Hub.
2. Vaya a la página Configuración.
3. En la página "Configuración", haga clic en "Migrate Workspace data to Custom Translator" (Migración de datos de área de trabajo a Custom Translator).



Translator Hub

[Projects](#) [Documents](#) [Members](#) [Settings](#) [Help](#)

Workspace Name:*

Dictionary only testing

Workspace ID:

a32f0eea-cd7b-44ca-9b36-50cb4fb10bf3

Organization:

Email:*

Liz.Sparks@contoso.com

?

Keep Data Private:

☐

?

Subscription Key:

?

*Required

Save Changes

Associate or Start a Microsoft Translator API subscription

Associating a Microsoft Translator subscription unlocks these features:

- Access to community and collaborative translations
- Import community and collaborative translations as training data
- Preauthorize higher levels of translation volumes


[Create new workspace / Join an existing workspace](#)

[Delete this workspace](#)

[Migrate Workspace data to Custom Translator](#)

4. En la siguiente página, seleccione cualquiera de estas dos opciones:

- Solo los entrenamientos implementados: si selecciona esta opción se migrarán solo los sistemas implementados y los documentos relacionados.
- Todos los entrenamientos correctos: si selecciona esta opción se migrarán todos los entrenamientos y los documentos relacionados.
- Escriba el identificador del área de trabajo de destino en Custom Translator.



Translator Hub

MIGRATE DATA TO CUSTOM TRANSLATOR

You have chosen to export your project(s) to Custom Translator.
Note: Once you migrate a project, you will not be able to do it again.

Select the trainings you want to export
You can choose to export only the currently deployed system or all successful systems.

☐ Deployed Trainings only.
☒ All Successful Trainings

Select the destination workspace on Custom Translator
Enter the Id of the destination workspace where you want this data to be migrated to :

You can find this in the settings page for the corresponding workspace in Custom Translator.

Submit Request

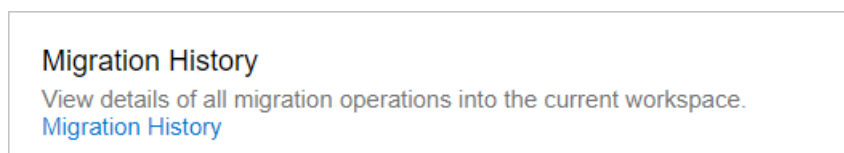
5. Haga clic en Enviar solicitud.

Historial de migraciones

Si solicita la migración del área de trabajo o de los proyectos de Microsoft Translator Hub, puede ver el historial de migraciones en la página Configuración de Custom Translator.

Para ver el historial de migraciones realice estos pasos:

1. Vaya a la página "Configuración" en el portal de Custom Translator.
2. En la sección del historial de migraciones de la página Configuración, haga clic en Historial de migraciones.



La página Historial de migraciones muestra la siguiente información como resumen para cualquier migración que solicite.

1. Migrado por: nombre y correo electrónico del usuario que envió esta solicitud de migración
2. Migrado el: marca de fecha y hora de la migración
3. Proyectos: número de proyectos para los que se solicitó la migración en comparación con el número de proyectos que se migraron correctamente.
4. Entrenamientos: número de entrenamientos para los que se solicitó la migración en comparación con el número de entrenamientos que se migraron correctamente.
5. Documentos: número de documentos para los que se solicitó la migración en comparación con el número de documentos que se migraron correctamente.

Workspace Settings > Migration History

Migrated By	Migrated On	Projects	Trainings	Documents	Details
Liz Sparks (Liz.Sparks@contoso.com)	2018-10-30 01:34 AM	Succeeded: 1 / 1	Succeeded: 9 / 9	Succeeded: 17 / 17	Export Details

Si desea obtener un informe de migración más detallado sobre los proyectos, aprendizajes y documentos, puede exportar los detalles en un archivo .csv.

Notas de implementación

- Los sistemas con pares de idiomas que todavía NO están disponibles en el Traductor personalizado solo podrán acceder a los datos o anular la implementación a través del Traductor personalizado. Estos proyectos se marcarán como "No disponible" en la página Proyectos. A medida que se habiliten nuevos pares de idiomas con el Traductor personalizado, los proyectos quedarán activos para su entrenamiento e implementación.
- La migración de un proyecto del centro a Traductor personalizado no tendrá ningún impacto en los entrenamientos o proyectos del centro. No se eliminan documentos ni proyectos del centro durante una migración y no se anula la implementación de modelos.
- Solo se permite migrar una vez por proyecto. Si necesita repetir una migración de un proyecto, póngase en contacto con nosotros.
- El Traductor personalizado admite los pares de idiomas de NMT desde y hacia el inglés. [Visualización de la lista completa de los idiomas admitidos](#). El centro no requiere modelos de referencia y, por tanto, admite varios miles de lenguajes. Puede migrar un par de idiomas no admitidos, pero solo se realizará la migración de los documentos y las definiciones del proyecto. No se podrá entrenar el modelo nuevo. Además, estos documentos y proyectos se mostrarán como inactivos para indicar que no se pueden utilizar en este momento. Si se agrega compatibilidad para estos proyectos y/o documentos, estarán activos y se podrán entrenar.
- Traductor personalizado no admite actualmente datos de entrenamiento monolingüe. Como con los pares de idiomas no admitidos, puede migrar documentos monolingües, pero se mostrarán como inactivos hasta que se admitan datos monolingües.
- Traductor personalizado requiere oraciones 10 000 frases en paralelo para el entrenamiento. Microsoft Hub podría entrenar sobre un conjunto más pequeño de datos. Si se migra un entrenamiento que no cumple este requisito, no será entrenado.

Custom Translator frente a Hub

En la siguiente tabla se comparan las características de Microsoft Translator Hub y Custom Translator.

	HUB	CUSTOM TRANSLATOR
Estado de la característica de personalización	Disponibilidad general	Disponibilidad general
Versión de Text API	V2	V3
Personalización de SMT	Sí	No
Personalización de NMT	No	Sí
Nueva personalización unificada de servicios de voz	No	Sí

	HUB	CUSTOM TRANSLATOR
Sin seguimiento	Sí	Sí

Nuevos idiomas

Si es una comunidad u organización que trabaja para crear un nuevo sistema de idioma para Microsoft Translator, vaya a custommt@microsoft.com para más información.

Pasos siguientes

- [Entrenamiento de un modelo.](#)
- Comience a utilizar el modelo de traducción personalizado implementado mediante [Microsoft Translator Text API V3](#).

Implementaciones en lenguajes no compatibles

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

Con la próxima retirada de Microsoft Translator Hub, Microsoft anulará la implementación de todos los modelos implementados actualmente a través de la instancia de Hub. Muchos de ustedes tienen modelos implementados en la instancia de Hub cuyos pares de idiomas no se admiten en Traductor personalizado. No queremos que los usuarios en esta situación carezcan de recursos para traducir su contenido.

Ahora tenemos un proceso que le permite implementar sus modelos no compatibles a través de Traductor personalizado. Este proceso le permite continuar traduciendo contenido mediante la API V3 más reciente. Estos modelos se hospedarán hasta que decida anular su implementación o el par de idiomas esté disponible en Traductor personalizado. En este artículo se explica el proceso de implementación de modelos con pares de idiomas no compatibles.

Requisitos previos

Para que sus modelos sean candidatos para la implementación, deben cumplir los siguientes criterios:

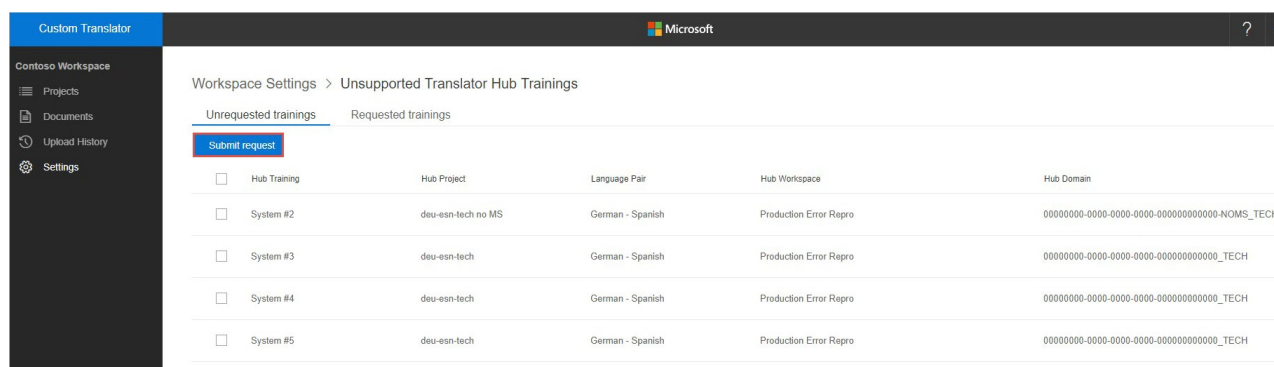
- El proyecto que incluye el modelo debe haber migrado de la instancia de Hub a Traductor personalizado mediante la herramienta de migración. [Aquí](#) encontrará el proceso de migración de proyectos y áreas de trabajo.
- El modelo debe tener el estado Implementado al producirse la migración.
- El par de idiomas del modelo debe ser un par de idiomas no compatible en Traductor personalizado. Los pares de idiomas en los que se admite un idioma, hacia o desde el inglés, pero el par en sí no incluye el inglés, son candidatos para las implementaciones de idiomas no compatibles. Por ejemplo, un modelo de Hub para un par de idiomas de francés a alemán se considera un par de idiomas no compatible, aunque francés a inglés e inglés a alemán son pares de idiomas compatibles.

Proceso

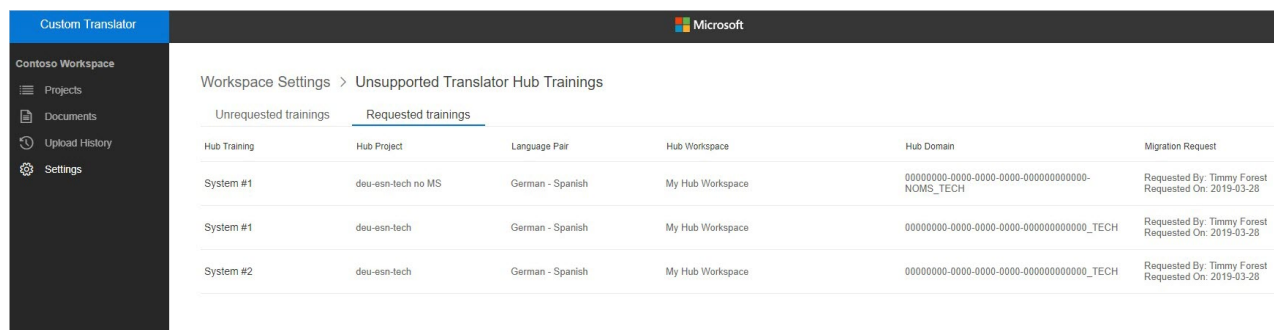
Una vez que haya migrado los modelos de la instancia de Hub que sean candidatos para la implementación, los encontrará yendo a la página **Configuración** para su área de trabajo y desplazamiento al final de la página donde verá la sección sobre **entrenamientos de Translator Hub no admitidos**. Solo aparecerá esta sección si tiene proyectos que cumplen los requisitos previos mencionados anteriormente.

The screenshot shows the 'Custom Translator' interface with the 'Workspace Settings' page. The left sidebar has a 'Settings' button highlighted with a red box. The main content area includes sections for 'Basic Information', 'Sharing Settings', 'Create New Workspace', and 'Migration History'. At the bottom, there is a section titled 'Unsupported Translator Hub Trainings' which is highlighted with a red box. This section contains text explaining that certain language pairs are not supported in Custom Translator and that users should select the appropriate Translator Hub trainings for migration.

En la página de selección de **entrenamientos de Translator Hub no admitidos**, la pestaña **Unrequested trainings** (Entrenamientos no solicitados) incluye modelos que cumplen los requisitos para la implementación. Seleccione los modelos que desea implementar y envíe una solicitud. Antes del 30 de abril, fecha límite de la implementación, podrá seleccionar tantos modelos como desee para la implementación.



Después del envío, el modelo dejará de estar disponible en la pestaña **Unrequested trainings** (Entrenamientos no solicitados) y aparecerá, en su lugar, en la pestaña **Requested trainings** (Entrenamientos solicitados). Podrá ver sus entrenamientos solicitados en cualquier momento.



Pasos siguientes

Los modelos que seleccionó para la implementación se guardarán una vez que se retire la instancia de Hub y se anule la implementación de todos los modelos. Hasta el 24 de mayo podrá enviar solicitudes de implementación de modelos no compatibles. Implementaremos dichos modelos el 15 de junio, momento en el que se podrá tener acceso a ellos a través de la API de Translator V3. Además, estarán disponibles a través de la API V2 hasta el 1 de julio.

Para obtener información adicional sobre fechas importantes relativas al desuso de la instancia de Hub, consulte [aquí](#). Una vez realizada la implementación, se aplicarán cargos de hospedaje normales. Consulte [Precios](#) para obtener detalles.

A diferencia de los modelos de Traductor personalizado estándar, los modelos de Hub solo estarán disponibles en una sola región, por lo que no se aplicarán cargos de hospedaje en varias regiones. Una vez realizada la implementación, podrá anular la implementación del modelo de Hub en cualquier momento a través del proyecto de Traductor personalizado migrado.

Pasos siguientes

- [Entrenamiento de un modelo](#).
- Comience a utilizar el modelo de traducción personalizado implementado mediante [Microsoft Translator Text API V3](#).

Glosario de Custom Translator

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

En el glosario de [Custom Translator](#) se explican los términos que puede encontrarse cuando trabaja con este.

PALABRA O EXPRESIÓN	DEFINICIÓN
Idioma de origen	El idioma de origen es el idioma del que parte y que desea convertir en otro idioma (el idioma de "destino").
Idioma de destino	El idioma de destino es el idioma que desea que la traducción automática proporcione después de recibir el idioma de origen.
Archivo monolingüe	Un archivo monolingüe es un archivo que solo tiene un idioma y que no está emparejado con otro archivo de un idioma diferente.
Archivos paralelos	Un archivo paralelo es una combinación de dos archivos con texto que se corresponde. Uno de ellos tiene el idioma de origen. El otro está en el idioma de destino.
Alineación de frases	El conjunto de datos paralelo debe tener las frases alineadas para aquellas frases que representan el mismo texto en ambos idiomas. Por ejemplo, en un archivo paralelo de origen la primera frase debería, en teoría, asignarse a la primera frase del archivo paralelo de destino.
Texto alineado	Uno de los pasos más importantes de la validación de archivos consiste en alinear las frases de los documentos paralelos. Las cosas se expresan de forma diferente en distintos idiomas. Además, los idiomas tienen un orden de las palabras diferente. Este paso permite realizar el trabajo de alineación de aquellas frases que tienen el mismo contenido para que se puedan usar en el aprendizaje. Un grado bajo de alineación de frases indica que hay algún error en uno de los archivos o en ambos.
División de palabras o anulación de la división	La división de palabras es la función encargada de marcar los límites entre las palabras. Muchos sistemas de escritura usan un espacio para denotar el límite entre las palabras. La anulación de la división hace referencia a la eliminación de cualquier marcador visible que se haya insertado entre las palabras en un paso anterior.
Delimitadores	Los delimitadores son las formas en que una frase se divide en segmentos o se delimita el margen entre las frases. Por ejemplo, en inglés los espacios delimitan las palabras, los dos puntos y el punto y coma delimitan oraciones, y el punto delimita frases.
Archivos de aprendizaje	Un archivo de aprendizaje se usa para enseñar al sistema de traducción automática como realizar asignaciones de un idioma (idioma de origen) a otro (idioma de destino). Cuantos más datos pueda proporcionar, mejor realizará el sistema la traducción.

PALABRA O EXPRESIÓN	DEFINICIÓN
Archivos de optimización	Estos archivos normalmente se derivan del conjunto de aprendizaje (si no ha seleccionado ningún conjunto de optimización). Las frases seleccionadas automáticamente se usan para optimizar el sistema y asegurarse de que funciona correctamente. Si decide crear sus propios archivos de optimización, asegúrese de que son un conjunto aleatorio de frases elegidas entre dominios si desea crear un modelo de traducción de uso general.
Archivos de prueba	Estos archivos son normalmente archivos derivados, seleccionados aleatoriamente del conjunto de aprendizaje (si no ha seleccionado ningún conjunto de prueba). El propósito de estas frases es evaluar la precisión del modelo de traducción. Son frases de las que desea asegurarse que el sistema las traduce correctamente. Por tanto, es posible que desee crear un conjunto de prueba y cargarlo al traductor para asegurarse de que estas frases se usan en la evaluación del sistema (generación de una puntuación BLEU).
Archivo combinado	Un tipo de archivo en el que se encuentran frases del idioma de origen y frases traducidas contenidas en el mismo archivo. Los formatos de archivo admitidos son: ".tmx", ".xliff", "XLF", ".lcl", ".xlsx".
Archivo de almacenamiento	Un archivo que contiene los demás archivos. Formatos de archivo admitidos: zip, gz, tgz.
Puntuación BLEU	BLEU es el método estándar del sector para evaluar la precisión del modelo de traducción. Aunque existen otros métodos de evaluación, Microsoft Translator se basa en el método BLEU para notificar la precisión a los propietarios de los proyectos.

Traductor personalizado: preguntas más frecuentes

13/01/2020 • 4 minutes to read • [Edit Online](#)

En este artículo se incluyen respuestas a preguntas frecuentes sobre [Custom Translator](#).

¿Cuáles son las restricciones actuales de Custom Translator?

Existen restricciones y límites con respecto al tamaño del archivo, el entrenamiento de los modelos y la implementación de los modelos. Recuerde estas restricciones cuando configure el entrenamiento para crear un modelo en Custom Translator.

- Los archivos enviados deben tener un tamaño inferior a 100 MB.
- No se admiten datos monolingües.

¿Cuándo debo solicitar la implementación de un sistema de traducción que ya se ha entrenado?

Crear el sistema de traducción óptimo para su proyecto puede requerir varios entrenamientos. Si la puntuación BLEU o los resultados de la prueba no son satisfactorios, es posible que quiera volver a intentarlo con más datos de aprendizaje o con datos filtrados más detenidamente. Debe ser estricto y cuidadoso mientras diseña sus conjuntos de ajuste y de pruebas, de forma que ambos estén completamente relacionados con la terminología y el estilo del material que quiere traducir. Puede ser más flexible en la creación de los datos de aprendizaje y experimentar con distintas opciones. Solicite una implementación de sistema cuando esté satisfecho con las traducciones en los resultados de prueba del sistema, cuando no tenga más datos que agregar al entrenamiento para mejorar el sistema, y cuando quiera acceder al modelo entrenado a través de las API.

¿Cuántos sistemas entrenados se pueden implementar en un proyecto?

Solo puede implementar un sistema entrenado por proyecto. Puede requerir de varios entrenamientos para crear un sistema de traducción adecuado para el proyecto, y recomendamos que solicite la implementación de un entrenamiento que le ofrezca el mejor resultado. Puede determinar la calidad del entrenamiento según la puntuación BLEU (mayor es mejor) y consultando con los revisores antes de decidir si la calidad de las traducciones es adecuada para la implementación.

¿Cuándo se implementarán mis entrenamientos?

La implementación normalmente tarda menos de una hora.

¿Cómo se tiene acceso a un sistema implementado?

Se puede acceder a los sistemas implementados a través de Microsoft Translator Text API V3 especificando el valor del id. de categoría. Puede encontrar más información sobre Translator Text API en la página web de [referencia de API](#).

¿Cómo omito la alineación y la separación de oraciones si mis datos ya están alineados por oración?

Custom Translator omite la alineación de oraciones y la separación de oraciones en los archivos TMX y en los archivos de texto con la extensión `.align`. Los archivos `.align` proporcionan a los usuarios una alternativa para

omitir el proceso de alineación y separación de oraciones de Traductor personalizado para los archivos que están perfectamente alineados y no necesitan de ningún procesamiento adicional. Se recomienda usar la extensión `.align` solo para los archivos que estén perfectamente alineados.

Si el número de oraciones extraídas no coincide con los dos archivos del mismo nombre base, Custom Translator ejecutará el proceso de alineación de oraciones en los archivos `.align` de todos modos.

He intentado cargar mi TMX, pero recibo el mensaje "Error de procesamiento de documento".

Asegúrese de que el TMX se ajusta a las especificaciones TMX 1.4b en <https://www.gala-global.org/tmx-14b>.