

IES AUGUSTO GONZALES DE LINARES

CFGS DAM2º

APLICACIÓN DE CONSULTA

A API REST

Programación De Servicios Y Procesos

Daniel Espinosa García

2023/2024

Índice

[1. Elección de API 2](#_Toc157963874)

[1.1. DeepL / translator 2](#_Toc157963875)

[2. Dependencias Cliente API (org.json) 4](#_Toc157963876)

[3. Métodos Relevantes en la Aplicación 5](#_Toc157963877)

[4. Demostración de funcionamiento 7](#_Toc157963878)

[5. Investigación sobre la creación de servidores API REST 7](#_Toc157963879)

[5.1. ¿Qué lenguaje, framework o librería se pueden usar? 7](#_Toc157963880)

[5.2. ¿Qué elementos hardware y software necesitarías para desplegar la API REST? 8](#_Toc157963881)

[6. Bibliografía 9](#_Toc157963882)

[7. Tabla de ilustraciones 10](#_Toc157963883)

# Elección de API

Siguiendo los criterios requeridos para la práctica y despues de investigar dos repositorios de repositorios de API’s gratuitos [RapidApi](https://rapidapi.com/hub) y [PublicApi](https://github.com/public-apis/public-apis) me decante por la utilización de la API DeeplL/ Translator que permite traducir texto a varios idiomas. En su versión gratuita permite un uso de 500.000 caracteres.

Me decante de un diseño sencillo en el cual el usuario solo necesita introducir el texto y el idioma de destino, la propia API es capaz de detectar el idioma de origen. Replicando el diseño que utiliza DeepL en la web.

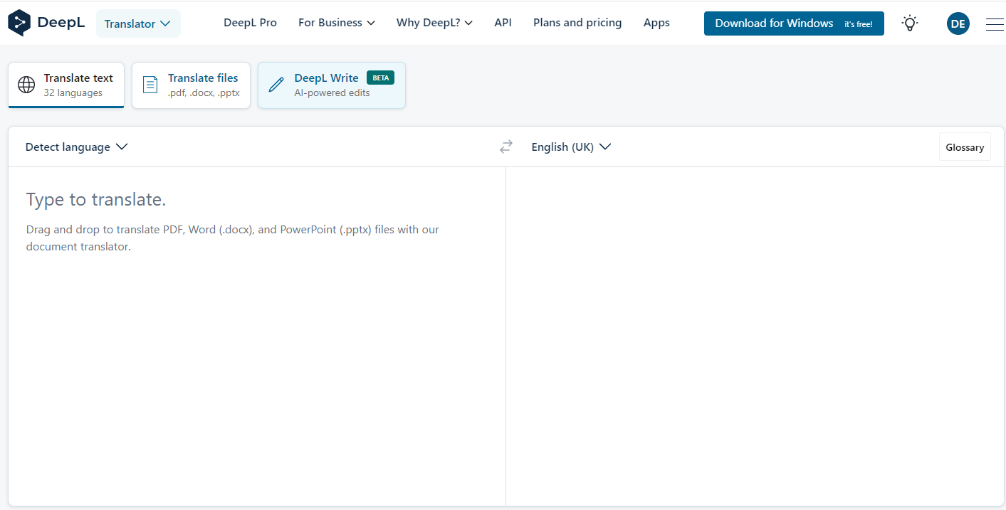


Ilustración 1 Interfaz web de DeepL / Translator

# DeepL / Translator

Lo primero que realice fue el registro en la Web para tener acceso a la API, la cual necesita de una ApiKey para autentificar al usuario.

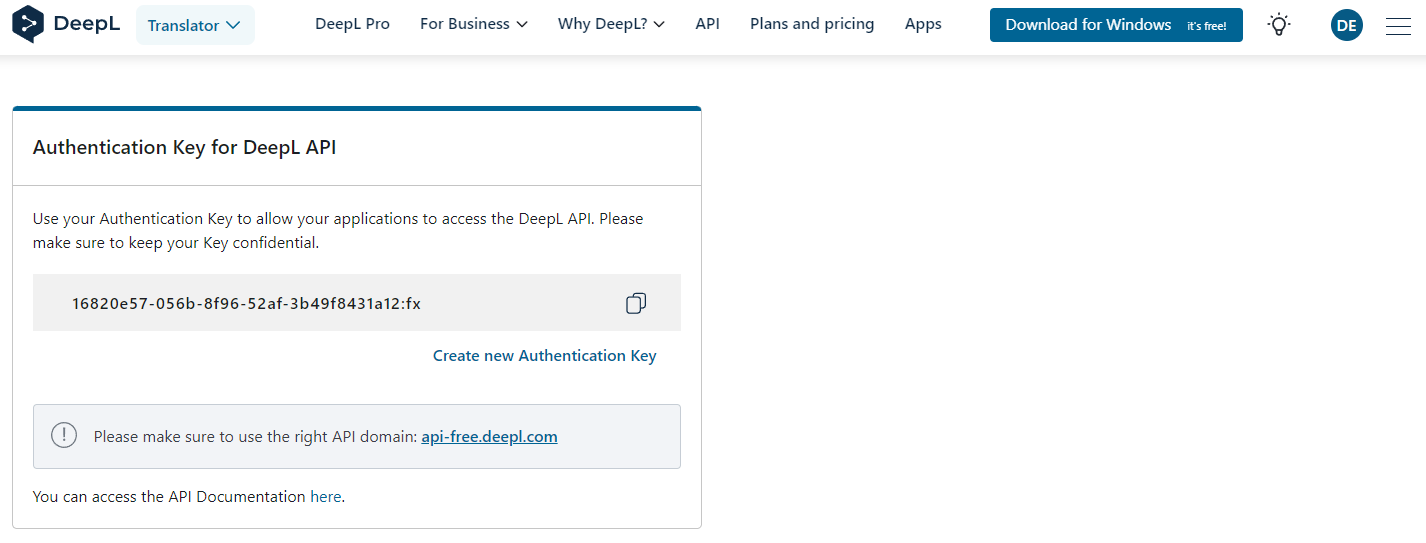


Ilustración 2 DeepL ApiKey

La API de DeepL cuenta con una [documentación](https://developers.deepl.com/docs/getting-started/readme) que explica su funcionamiento, en ella encontré los detalles de cómo utilizar los End Points que utilizo en la práctica.

**End Point /Translate**

Permite enviando un texto y un lenguaje de objetivo que la API lo traduzca. No es necesario introducir el lenguaje original del texto ya que ella misma se ocupa de averiguarlo.

Aun asi entre las opciones que permite la API se incluye:

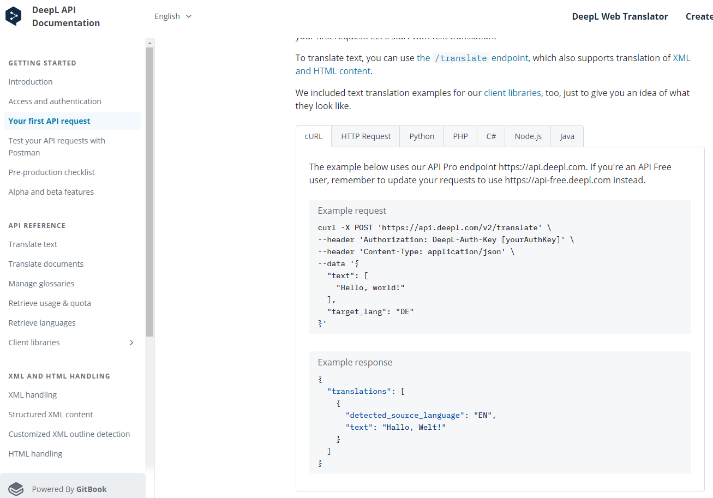


Ilustración 3 Documentación End Point Translate

- source\_lang: Indicar el idioma de origen del texto

- target\_lang: Indica el idioma al que tiene que ser traducido el texto

- formality: La formalidad del texto traducido, la cual tiene niveles de ajuste.

Devuelve la información en un JSON Array que contiene dentro un JSON Object con la traducción y los parámetros solicitados.

**End Point /Languages**

Este end point nos da acceso a todos los idiomas que permite la aplicación traducir.

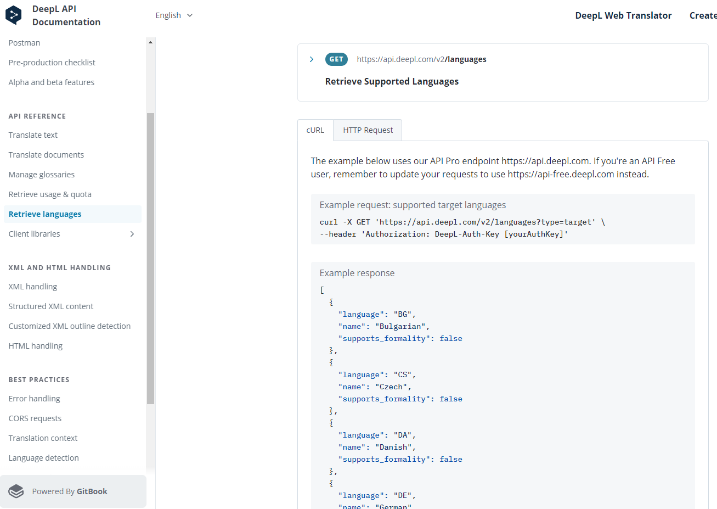


Ilustración 4 Documentación End Point Languages

Dándonos su nombre, una abreviatura que se utilizara para designar el idioma de destino u origen del texto a traducir y si permite que pueda ser pasado a un lenguaje más formal o no en modo de booleano.

Nos entrega la información en modo de un JSON Array que contiene múltiples JSON Objects uno por cada idioma.

**End Point /Usage**

Permite saber el uso de la aplicación y el límite que tiene el usuario, en el modo gratuito cuenta con 500.000 caracteres al mes para traducir.

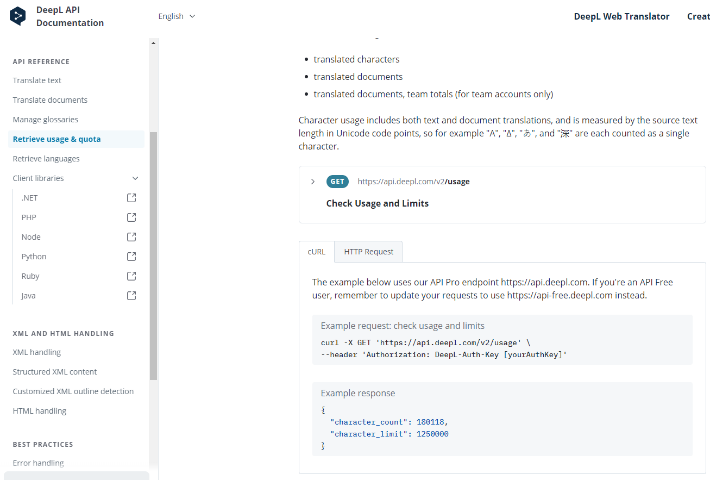


Ilustración 5 Documentación End Point Usage

Este end point solo requiere que se le envié la ApiKey del usuario y devolverá el total de caracteres utilizados y el límite de caracteres que tenga la cuenta.

Entrega la información en modo de un JSON Object.

# Dependencias Cliente API (org.json)

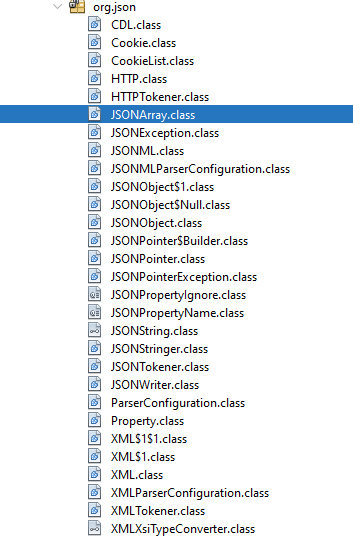


Ilustración 6 Clases de la Dependencia org.json

Para realizar la práctica seleccione la dependencia [org.json](https://mvnrepository.com/artifact/org.json/json/20231013) que consta de distintas clases con sus metodos que permiten realizar el manejo de la información recibida en formato JSON, de los cuales caben mencionar:

JSONArray: nos permite crear objetos de esta clase para almacenar la información recibida en un Array de Objetos y poder recorrerlo rescatando la información almacenada.

JSONObject: esta clase tambien nos permite crear objetos de esta, y almacena la información en mapas de Clave Valor, que es la manera que utiliza JSON para tratar la información, permitiendo la búsqueda de los datos de manera rápida utilizando los metodos que la clase contiene.

Utilizaremos estas clases y algunos de sus metodos para realizar la práctica.

# Métodos Relevantes en la Aplicación

La aplicación consta de cuatro metodos relevantes los cuales atacan a los End Points antes mencionados.

**Método traducir()**

Utilizará el End Point [/Translate](#_DeepL_/_translator) al cual enviara la información al servidor y devolverá un JSON con el texto traducido y el idioma de origen, trataremos la información utilizando la dependencia org.json para mostrarla en la interfaz gráfica.



Ilustración 7 Método traducir

**Método obtainLanguages()**

Este método utiliza el End Point [/Languages](#_DeepL_/_translator) el cual tras realizar la petición el servidor nos envía un JSONArray que contiene JSONObject con los lenguajes que permite traducir. Recojo la información que necesito, en este caso, name y language. Para almacenarlo en un mapa de Clave Valor. Mostrando por la interfaz gráfica la Clave para que el usuario seleccione el idioma al que quiere traducir el texto.



Ilustración 8 Método obtainLanguages

**Método usage()**

Ataca al End Point [/Usage](#_DeepL_/_translator) recibiendo como respuesta del servidor un JSONObject con un par de claves valor uno para el numero de caracteres utilizados y otro para el límite de caracteres mensual de la cuenta. Esta información se muestra en la interfaz gráfica.



Ilustración 9 Método usage

**Método statusService()**

Para comprobar que el servicio está activo al iniciar la aplicación ataco dos End Points de DeepL si todo es correcto devuelve True en caso contrario False. Tanto este como los metodos anteriores utilizan el método .timeout() que permite establecer un tiempo de espera de respuesta del servidor y si falla devuelve una excepción.



Ilustración 10 Método statusService

# Demostración de funcionamiento

Mediante la interfaz gráfica se nos permite:

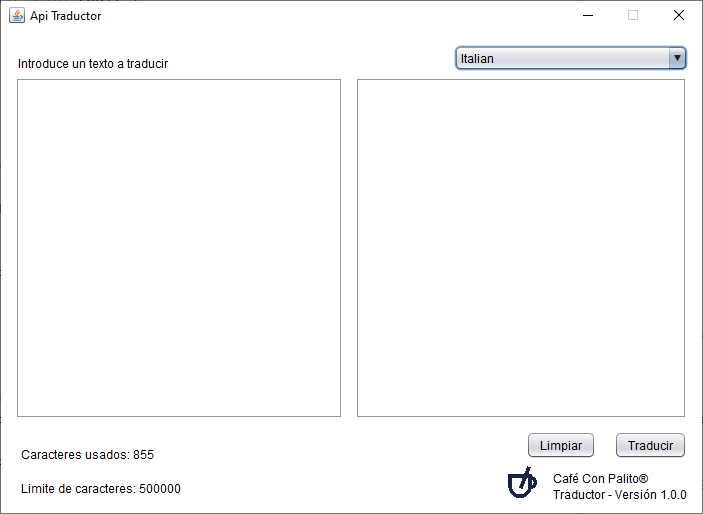


Ilustración 11 Aplicación Traductor

* Introducir un texto en el TextArea para su traducción.
* Seleccionar mediante un ComboBox el lenguaje al que se quiere traducir el texto, obtenidos desde la API
* Utilizando el botón de Traducir traduce el texto introducido al idioma seleccionado.
* El botón de limpiar limpia los TextArea.

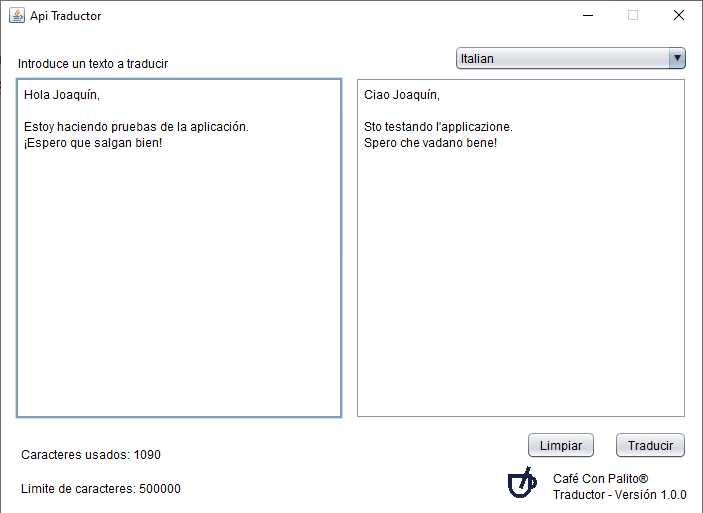


Ilustración 12 Demostración de uso de la aplicación

# Investigación sobre la creación de servidores API REST

Existen diversas maneras de crear Api Rest y cada lenguaje tiene su propia manera de trabajar con ello, investigue los tres lenguajes más usados Java, Python y C# en el siguiente apartado detallare un poco más su funcionamiento, pero todos permiten las conexiones concurrentes.

# ¿Qué lenguaje, framework o librería se pueden usar?

**JAVA**

* Framework: Spring
* Dependencias: Spring Boot, Spring Data, Spring Web, Swagger
* Ventaja: al ser un lenguaje muy estricto y muy estructurado el IDE detectara cualquier problema.
* Desventaja: la configuración inicial es muy larga y es necesario tener una buena base de programacion.

**PYTHON**

* Framework: FastApi
* Dependencias: SQLAlchemy, Alembic
* Ventaja: configuración inicial fácil y fácil de programar
* Desventaja: como no es un lenguaje fuertemente tipado y estructurado es mucho más facil cometer errores

**C#**

* Framework: .net
* Dependencias: asp.net, Swagger
* Ventaja: la configuración inicial es facil, lenguaje estructurado y tipado
* Desventaja: es necesario tener una buena base de programacion

# ¿Qué elementos hardware y software necesitarías para desplegar la API REST?

Como diría Jose Francisco Márquez, nuestro profesor de DB de 1º, ¡depende!

A nivel de hardware dependerá del número de conexiones concurrentes que recibirá el servicio ademas de tener en cuenta la cantidad de consultas y/o datos que tendrá que gestionar.

Estos servicios se pueden desplegar tanto en “local”, si se dispone de un servidor, o en la nube, esta última es la más utilizada en la actualidad, empresas como Google con Google Cloud, Amazon con AWS o Microsoft con Azure, permiten su despliegue de manera rápida y relativamente sencilla.

Según el tipo de lenguaje necesitaremos como requisitos mínimos de software el compilador o interprete y las dependencias en la máquina que correrá el servicio.

# Bibliografía

*ASP.net*. (s.f.). Obtenido de Microsoft: https://dotnet.microsoft.com/es-es/apps/aspnet

*DeepL API Documentación*. (s.f.). Obtenido de DeepL: https://developers.deepl.com/docs/getting-started/readme

Oracle. (s.f.). *Oracle JavaDoc 17*. Obtenido de https://docs.oracle.com/en/java/javase/17/docs/api/index.html

*org.json*. (s.f.). Obtenido de Maven Repository: https://mvnrepository.com/artifact/org.json/json/20231013

*org.json*. (s.f.). Obtenido de Git Hub: https://github.com/stleary/JSON-java

*Public Apis*. (s.f.). Obtenido de https://github.com/public-apis/public-apis

*Rapid Api*. (s.f.). Obtenido de https://rapidapi.com/hub

*Spring.io*. (s.f.). Obtenido de https://spring.io/

*SQLAlchemy*. (s.f.). Obtenido de SQLAlchemy: https://www.sqlalchemy.org/

# Tabla de ilustraciones

[Ilustración 1 Interfaz web de DeepL / Translator 2](file:///C:\Users\produccion\Desktop\DAM2\PSP\PSP%2004\Tareas\PSP_Tema4_Tarea2_API.docx#_Toc157963536)

[Ilustración 2 DeepL ApiKey 2](file:///C:\Users\produccion\Desktop\DAM2\PSP\PSP%2004\Tareas\PSP_Tema4_Tarea2_API.docx#_Toc157963537)

[Ilustración 3 Documentación End Point Translate 3](file:///C:\Users\produccion\Desktop\DAM2\PSP\PSP%2004\Tareas\PSP_Tema4_Tarea2_API.docx#_Toc157963538)

[Ilustración 4 Documentación End Point Languages 3](file:///C:\Users\produccion\Desktop\DAM2\PSP\PSP%2004\Tareas\PSP_Tema4_Tarea2_API.docx#_Toc157963539)

[Ilustración 5 Documentación End Point Usage 4](file:///C:\Users\produccion\Desktop\DAM2\PSP\PSP%2004\Tareas\PSP_Tema4_Tarea2_API.docx#_Toc157963540)

[Ilustración 6 Clases de la Dependencia org.json 4](file:///C:\Users\produccion\Desktop\DAM2\PSP\PSP%2004\Tareas\PSP_Tema4_Tarea2_API.docx#_Toc157963541)

[Ilustración 7 Método traducir 5](file:///C:\Users\produccion\Desktop\DAM2\PSP\PSP%2004\Tareas\PSP_Tema4_Tarea2_API.docx#_Toc157963542)

[Ilustración 8 Método obtainLanguages 5](file:///C:\Users\produccion\Desktop\DAM2\PSP\PSP%2004\Tareas\PSP_Tema4_Tarea2_API.docx#_Toc157963543)

[Ilustración 9 Método usage 6](file:///C:\Users\produccion\Desktop\DAM2\PSP\PSP%2004\Tareas\PSP_Tema4_Tarea2_API.docx#_Toc157963544)

[Ilustración 10 Método statusService 6](file:///C:\Users\produccion\Desktop\DAM2\PSP\PSP%2004\Tareas\PSP_Tema4_Tarea2_API.docx#_Toc157963545)

[Ilustración 11 Aplicación Traductor 7](file:///C:\Users\produccion\Desktop\DAM2\PSP\PSP%2004\Tareas\PSP_Tema4_Tarea2_API.docx#_Toc157963546)

[Ilustración 12 Demostración de uso de la aplicación 7](file:///C:\Users\produccion\Desktop\DAM2\PSP\PSP%2004\Tareas\PSP_Tema4_Tarea2_API.docx#_Toc157963547)