

Necesito implementar en código Java la API de OpenAI, como puedo hacerlo?

Para utilizar la API de OpenAI en Java, puedes seguir los siguientes pasos:

Regístrate en la página de OpenAI y obtén tus credenciales API.

Descarga la biblioteca cliente oficial de OpenAI para Java llamada `openai-java-sdk`. Puedes descargarlo desde su página de GitHub o agregarlo a tu proyecto a través de Maven.

Importa la biblioteca `openai-java-sdk` a tu proyecto Java y crea una instancia del cliente OpenAI utilizando tus credenciales API.

Utiliza los métodos proporcionados por la biblioteca `openai-java-sdk` para interactuar con la API de OpenAI. Por ejemplo, puedes llamar al método `completions()` para generar texto utilizando el modelo de lenguaje GPT-3.

A continuación se muestra un ejemplo de código que utiliza la API de OpenAI para generar texto:

java

Copy code

```
import ai.openai.api.*;
import ai.openai.api.models.*;

public class OpenAITest {

    public static void main(String[] args) throws Exception {
        String apiKey = "TU_API_KEY";
        OpenAI api = new OpenAI(apiKey);

        String prompt = "Escribe aquí tu prompt";
        CompletionRequest request = new CompletionRequest.Builder()
            .prompt(prompt)
            .model("text-davinci-002")
```

```
.temperature(0.5)

.maxTokens(60)

.build();
```

```
CompletionResponse response = api.completions(request);

String text = response.getChoices().get(0).getText();

System.out.println(text);

}

}
```

Este código utiliza la API de OpenAI para generar texto a partir del prompt proporcionado utilizando el modelo de lenguaje GPT-3. El texto generado se imprime en la consola. Recuerda reemplazar "TU_API_KEY" con tus credenciales API válidas.

Mikel Murugarren

Como sería ese código en caso de querer utilizar el modelo que han lanzado recientemente, ChatGPT?

Si quieres utilizar el modelo de lenguaje ChatGPT de OpenAI, puedes hacerlo siguiendo los mismos pasos que mencioné anteriormente. Solo necesitas cambiar el identificador del modelo en la solicitud de completado de la API de OpenAI.

Aquí hay un ejemplo de código Java que utiliza la API de OpenAI para generar texto utilizando el modelo de lenguaje ChatGPT:

java

Copy code

```
import ai.openai.api.*;

import ai.openai.api.models.*;

public class ChatGPTExample {
```

```

public static void main(String[] args) throws Exception {

    String apiKey = "TU_API_KEY";

    OpenAI api = new OpenAI(apiKey);


    String prompt = "Hola, ¿cómo estás?";

    CompletionRequest request = new CompletionRequest.Builder()

        .prompt(prompt)

        .model("text-davinci-002")

        .temperature(0.5)

        .maxTokens(60)

        .build();


    CompletionResponse response = api.completions(request);

    String text = response.getChoices().get(0).getText();

    System.out.println(text);

}

}

```

En este ejemplo, se ha especificado el identificador del modelo como "text-davinci-002", que es la última versión del modelo de lenguaje GPT-3. Para utilizar el modelo ChatGPT, solo necesitas cambiar el identificador del modelo a "text-davinci-002-chat". Por ejemplo:

java

Copy code

```

CompletionRequest request = new CompletionRequest.Builder()

    .prompt(prompt)

    .model("text-davinci-002-chat")

    .temperature(0.5)

    .maxTokens(60)

    .build();

```

Con esto deberías poder utilizar el modelo de lenguaje ChatGPT de OpenAI en tu aplicación Java. Recuerda reemplazar "TU_API_KEY" con tus credenciales API válidas.

Mikel Murugarren

Gracias! Podrías ahora darme un ejemplo donde ese código es una función llamada "peticionChatGpt" con el prompt parametrizado a la que se llama desde la función main?

¡Claro! Aquí te dejo un ejemplo de cómo podrías definir la función peticionChatGpt con el prompt como un parámetro:

java

Copy code

```
import ai.openai.api.*;
import ai.openai.api.models.*;

public class ChatGPTExample {

    public static void main(String[] args) throws Exception {

        String apiKey = "TU_API_KEY";
        OpenAI api = new OpenAI(apiKey);

        String prompt = "Hola, ¿cómo estás?";
        String response = peticionChatGpt(api, prompt);
        System.out.println(response);
    }

    public static String peticionChatGpt(OpenAI api, String prompt) throws Exception {
        CompletionRequest request = new CompletionRequest.Builder()
            .prompt(prompt)
            .model("text-davinci-002-chat")
```

```

        .temperature(0.5)

        .maxTokens(60)

        .build();

    CompletionResponse response = api.completions(request);

    String text = response.getChoices().get(0).getText();

    return text;
}

}

```

En este ejemplo, hemos definido la función `peticionChatGpt` que toma dos argumentos: una instancia del cliente de OpenAI `api` y el `prompt` para generar una respuesta de ChatGPT. La función utiliza la API de OpenAI para generar texto utilizando el modelo de lenguaje ChatGPT y devuelve la respuesta generada.

En la función `main`, hemos creado una instancia del cliente `api` y hemos llamado a la función `peticionChatGpt` con el `prompt` "Hola, ¿cómo estás?" como argumento. La respuesta generada se almacena en la variable `response` y se imprime en la consola.

Recuerda reemplazar "TU_API_KEY" con tus credenciales API válidas y ajustar los valores de los parámetros `temperature` y `maxTokens` según tus necesidades.

Mikel Murugarren

Gracias! Puedes darme una breve explicación de lo que son las variables `temperature` y `maxTokens`?
¡Por supuesto!

`temperature`: Es un parámetro que controla la "creatividad" de las respuestas generadas por el modelo de lenguaje. Valores más altos de temperatura producen respuestas más diversas y menos predecibles, mientras que valores más bajos producen respuestas más conservadoras y predecibles. La temperatura debe estar en el rango de 0 a 1, siendo 0 una respuesta determinística y 1 una respuesta altamente creativa. El valor predeterminado es 1.

maxTokens: Es un parámetro que controla la longitud máxima de la respuesta generada por el modelo de lenguaje. El modelo generará una respuesta de hasta maxTokens palabras. Si se establece un valor más alto, es más probable que el modelo genere respuestas más largas y detalladas, pero también pueden ser menos coherentes. Si se establece un valor más bajo, las respuestas serán más concisas y coherentes, pero también pueden ser menos detalladas. El valor predeterminado es 16.

Es importante experimentar con diferentes valores de temperatura y maxTokens para obtener las mejores respuestas para tu caso de uso específico.

Mikel Murugarren

El valor de alguna de estas variables tiene consecuencias en el rendimiento de la aplicación que queremos desarrollar?

Sí, el valor de estos parámetros puede tener un impacto en el rendimiento de tu aplicación, especialmente en términos de la velocidad de respuesta y la cantidad de datos que se envían a través de la red.

En general, cuanto más alta sea la temperatura y el valor de maxTokens, más tiempo tardará el modelo en generar una respuesta y más datos se enviarán a través de la red. Si estás utilizando la API de OpenAI en una aplicación en línea que tiene una gran cantidad de tráfico, es posible que desees ajustar estos valores para obtener un equilibrio adecuado entre la calidad de la respuesta y la velocidad de la respuesta.

Además, ten en cuenta que los modelos más grandes, como el modelo GPT-3, pueden tardar más tiempo en generar una respuesta en comparación con los modelos más pequeños, lo que puede afectar el rendimiento de tu aplicación. En general, es recomendable probar y ajustar estos parámetros para encontrar el equilibrio correcto entre la calidad de la respuesta y el rendimiento de tu aplicación.

Mikel Murugarren

Gracias! Podrías mostrarme ahora un ejemplo de código typescript, donde se haga una petición HttpClient a nuestra aplicación pasando como parametro el prompt? Teniendo en cuenta que la aplicación java que implementa la API de OpenAI esta en localhost:127.0.0.1:8080/chatgpt

¡Claro! Aquí te muestro un ejemplo de cómo hacer una petición HTTP POST a la aplicación Java que implementa la API de OpenAI, usando TypeScript y Angular HttpClient.

Para hacer la petición HTTP, debes tener en cuenta que la API de OpenAI requiere que el payload sea enviado en formato JSON en el cuerpo de la solicitud POST. Además, debes enviar tu clave de API de OpenAI como encabezado de autorización.

typescript

Copy code

```
import { HttpClient, HttpHeaders } from '@angular/common/http';
```

```
import { Injectable } from '@angular/core';
```

```
@Injectable({
```

```
  providedIn: 'root'
```

```
})
```

```
export class ChatGptService {
```

```
  private baseUrl = 'http://127.0.0.1:8080/chatgpt';
```

```
  constructor(private http: HttpClient) { }
```

```
  peticionChatGpt(prompt: string) {
```

```
    const body = {
```

```
      prompt: prompt,
```

```
      model: "text-davinci-002-chat",
```

```
      temperature: 0.5,
```

```
      max_tokens: 60
```

```
    };
```

```
    const headers = new HttpHeaders({
```

```
    "Content-Type": "application/json",  
    "Authorization": "Bearer " + "TU_API_KEY"  
  });  
  
  return this.http.post(this.baseUrl, body, { headers: headers });  
}  
}
```

En este ejemplo, hemos creado un servicio ChatGptService que tiene un método peticiónChatGpt que toma un argumento prompt y devuelve un observable que contiene la respuesta de la API de OpenAI.

En la función peticiónChatGpt, hemos construido el cuerpo de la solicitud POST como un objeto JavaScript que contiene el prompt, el modelo, la temperatura y el máximo de tokens. Luego, hemos creado un objeto HttpHeaders que contiene el encabezado de autorización con tu clave de API de OpenAI.

Finalmente, hemos hecho una llamada al método post del objeto HttpClient de Angular, pasando la URL de la API de OpenAI, el cuerpo de la solicitud y los encabezados como parámetros. La respuesta de la API de OpenAI se devuelve como un observable.

Recuerda reemplazar "TU_API_KEY" con tus credenciales API válidas y ajustar los valores de los parámetros temperature y maxTokens según tus necesidades. Además, debes asegurarte de que la aplicación Java que implementa la API de OpenAI esté en ejecución en localhost:127.0.0.1:8080/chatgpt.

Mikel Murugarren

Como puedo conseguir mi API_KEY?

Para obtener tu clave de API de OpenAI, debes registrarte en el sitio web de OpenAI y crear una cuenta. A continuación, debes crear una solicitud de acceso a la API a través del formulario de solicitud en el sitio web de OpenAI.

La solicitud de acceso a la API requerirá que proporciones algunos detalles sobre ti y sobre tu proyecto, así como una descripción de cómo planeas usar la API de OpenAI. Una vez que se apruebe

tu solicitud, se te proporcionará una clave de API que puedes usar para acceder a los servicios de OpenAI.

Ten en cuenta que el acceso a la API de OpenAI puede tener un costo, dependiendo del tipo de uso que le des y de la cantidad de solicitudes que realices. Asegúrate de leer la documentación de OpenAI para conocer los detalles sobre los precios y las limitaciones de uso de la API.

Una vez que tengas tu clave de API de OpenAI, podrás usarla para hacer solicitudes a la API de OpenAI desde tu aplicación, como se muestra en el ejemplo de código que te proporcioné anteriormente.

Mikel Murugarren

Puedo descargar esta conversación para revisar los consejos que me has dado mas tarde?

Por supuesto, puedes guardar esta conversación para revisar los consejos que te he dado más tarde.

Una forma sencilla de hacerlo es copiar y pegar la conversación en un archivo de texto o documento. Otra opción es hacer una captura de pantalla de la conversación y guardarla como una imagen.

También puedes utilizar la opción de "Exportar conversación" o "Guardar conversación" que ofrecen algunas plataformas de chat, si está disponible en la plataforma en la que estamos conversando.

Espero que los consejos que te he brindado sean útiles para ti. Si tienes más preguntas, no dudes en preguntar. ¡Estoy aquí para ayudarte!