

Google Cloud Speech-to-Text

最近、Google Cloud の Speech-to-Text API を試してみました。以下は、その文字起こしを行うために使用した Python 関数です。

```
import os
import json
import time
import argparse
from google.cloud import speech
from pydub import AudioSegment
import tempfile

# 固定の出力ディレクトリ
OUTPUT_DIRECTORY = "assets/transcriptions"

def speech_to_text(audio_file, output_filename):
    print(f" 生成中の文字起こし: {output_filename}")
    try:
        client = speech.SpeechClient()

        # pydub を使用してオーディオファイルを読み込み、パラメータを決定
        audio_segment = AudioSegment.from_file(audio_file)
        sample_rate = audio_segment.frame_rate
        channels = audio_segment.channels

        # ファイル拡張子に基づいてエンコーディングを決定
        file_extension = os.path.splitext(audio_file)[1].lower()
        if file_extension == '.mp3':
            encoding = speech.RecognitionConfig.AudioEncoding.MP3
        elif file_extension in ['.wav', '.wave']:
            encoding = speech.RecognitionConfig.AudioEncoding.LINEAR16
        elif file_extension == '.flac':
            encoding = speech.RecognitionConfig.AudioEncoding.FLAC
        else:
            print(f" サポートされていないファイル形式: {file_extension}")
            return

        # 認識設定
```

```

config = speech.RecognitionConfig(
    encoding=encoding,
    sample_rate_hertz=sample_rate,
    audio_channel_count=channels,
    language_code="en-US", # ロジックに基づいて設定
)

with open(audio_file, "rb") as f:
    audio_content = f.read()

audio = speech.RecognitionAudio(content=audio_content)

# 長時間の音声認識を実行
try:
    operation = client.long_running_recognize(config=config, audio=audio)
    response = operation.result(timeout=300) # 必要に応じてタイムアウトを調整
except Exception as e:
    print(f" 文字起こし中のエラー: {e}")
    return

print(response.results)

transcription = ""
for result in response.results:
    transcription += result.alternatives[0].transcript + "\n"

with open(output_filename, "w", encoding="utf-8") as f:
    f.write(transcription)
print(f" 文字起こしが {output_filename} に書き込まれました")

except Exception as e:
    print(f"{output_filename} の文字起こし生成中にエラーが発生しました: {e}")

def process_audio_files(input_dir, output_dir):
    os.makedirs(output_dir, exist_ok=True)

    all_audio_files = [f for f in os.listdir(input_dir) if f.endswith(('mp3', 'wav', 'm4a'))]
    total_files = len(all_audio_files)
    print(f" 処理するオーディオファイルの総数: {total_files}")

```

```

if total_files == 0:
    print(f"{input_dir}' ディレクトリにオーディオファイルが見つかりません。")
    return

files_processed = 0

for filename in all_audio_files:
    audio_file_path = os.path.join(input_dir, filename)
    output_filename = os.path.join(output_dir, f"{os.path.splitext(filename)[0]}.txt")
    if os.path.exists(output_filename):
        print(f"{filename} をスキップ: {output_filename} は既に存在します。")
        continue
    print(f"\n処理中 {files_processed + 1}/{total_files}: {filename}")
    try:
        # ファイル名のサフィックスに基づいて言語を決定
        if filename.endswith('-zh.mp3') or filename.endswith('-zh.wav') or filename.endswith('-zh.m4a'):
            language_code = "cmn-CN"
        else:
            language_code = "en-US"

        # 必要に応じて speech_to_text 内の config を更新
        # 簡略化のため、speech_to_text 内で language_code を設定します

        speech_to_text(
            audio_file=audio_file_path,
            output_filename=output_filename,
        )
        files_processed += 1
        print(f" ファイル {files_processed}/{total_files} を処理しました。 \n")
    except Exception as e:
        print(f"{filename} の処理に失敗しました: {e}")
        continue

print(f" 処理が完了しました！ {files_processed}/{total_files} ファイルを処理しました。")

if __name__ == "__main__":
    parser = argparse.ArgumentParser(description=" オーディオファイル进行处理して文字起こしを生成します。")
    parser.add_argument('--input_dir', type=str, default="assets/audios", help=" オーディオファイルの入力ディレクトリ")

```

```
args = parser.parse_args()
```

```
process_audio_files(  
    input_dir=args.input_dir,  
    output_dir=OUTPUT_DIRECTORY,  
)
```