

You

I'm writing python code for a text to speech app in a google colab python notebook, here is my code:

Cell 1:

```
from google.cloud import storage
from google.colab import auth

def uploadAudio(Filepath):
    # Authenticate the user
    auth.authenticate_user()

    # Initialize the storage client
    storage_client = storage.Client()

    # Set your bucket name and file path
    bucket_name = 'speech-to-text-app1' # Replace with your bucket name
    audio_file_path = Filepath
    destination_blob_name = Filepath

    # Upload the file bucket = storage_client.bucket(bucket_name)
    blob = bucket.blob(destination_blob_name)
    blob.upload_from_filename(audio_file_path)

    # Get the GCS URI
    gcs_uri = f'gs://{bucket_name}/{destination_blob_name}'
    print(f'File uploaded to {gcs_uri}')
    return gcs_uri
```

Cell 2:

```
!pip install google-cloud-speech
from google.cloud import speech
from google.api_core.client_options import ClientOptions
import io

def processAudioFile(audioPath):
    # Set the correct project ID
    project_id = "bcft-429709" # Replace with your actual project ID

    # Set client options, including the project ID
    client_options = ClientOptions(quota_project_id=project_id)

    # Initialize the client with the client options
    client = speech.SpeechClient(client_options=client_options)

    gcs_uri = uploadAudio(audioPath)

    audio = speech.RecognitionAudio(uri=gcs_uri)
    config = speech.RecognitionConfig(
        encoding=speech.RecognitionConfig.AudioEncoding.LINEAR16,
        sample_rate_hertz=16000,
        language_code="en-US",
    )

    # Use long_running_recognize for longer audio files
    operation = client.long_running_recognize(config=config, audio=audio)
```

```
# Wait for the operation to complete
response = operation.result(timeout=900)
print(response)
return response
```

Cell 3:

```
# Authenticate the user
from google.colab import auth
auth.authenticate_user()

from google.colab import files

print("Please upload an audio file")
audio_upload = files.upload()

# Extract the filename from the uploaded file
audio_filename = next(iter(audio_upload.keys())) if audio_upload else None

if audio_filename and (audio_filename.lower().endswith('.mp3') or
audio_filename.lower().endswith('.wav') or audio_filename.lower().endswith('.m4a')):
    audio_path = audio_filename
    print(f"Audio file uploaded successfully: {audio_filename}")
else:
    print("Error: Please upload a Audio file (.mp3, .wav or .m4a)")

response = processAudioFile(audio_path)
try:
    print("The response is as follows:", response)

    # Check if there are any results
    if not response.results:
        print("No transcription results found.")
    else:
        for result in response.results:
            print("Transcript: {}".format(result.alternatives[0].transcript))

except Exception as e:
    print(f"Error during speech recognition: {e}")
```

I'm running cell 3 directly and the other 2 cells are run indirectly through function calls, however I'm not getting a transcript, this is my output:

```
Audio file uploaded successfully: (Audio) Screenrecorder-2024-09-02-09-37-06-336.m4a
File uploaded to gs://speech-to-text-app1/(Audio) Screenrecorder-2024-09-02-09-37-06-336.m4a
total_billed_time {
seconds: 147
}
request_id: 3443228437626341026

The response is as follows:
total_billed_time {
seconds: 147
}
request_id: 3443228437626341026
```

No transcription results found.

Why could this be and how can I fix it, do I have to use a different method maybe?

Copilot

It looks like your code is correctly uploading the audio file and making a request to the Google Cloud Speech-to-Text API, but the API isn't returning any transcription results. Here are a few things you can check and try to fix the issue:

1. **Audio Encoding and Format:** Ensure that the audio file is in the correct format and encoding. Your code specifies LINEAR16 encoding and a sample rate of 16000 Hz. If your audio file is in a different format, you might need to convert it. You can use tools like `ffmpeg` to convert your audio file to the required format.
2. **Language Code:** Verify that the language code (en-US) matches the language spoken in the audio file. If the audio is in a different language, you need to update the `language_code` parameter accordingly.
3. **Audio File Content:** Ensure that the audio file contains clear speech and is not too noisy. Background noise or unclear speech can affect the transcription accuracy.
4. **API Quotas and Limits:** Check if you have reached any API quotas or limits. You can monitor your usage in the Google Cloud Console.
5. **Debugging the Response:** Print out more details from the response to understand why no transcription results are found. For example, you can print the response object to see if there are any error messages or additional information.

```
print(response)
```

6. **Alternative Methods:** If the issue persists, you might want to try using the `recognize` method instead of `long_running_recognize` for shorter audio files. This method is synchronous and might give you more immediate feedback.

```
response = client.recognize(config=config, audio=audio)
```

Here's an updated version of your `processAudioFile` function with additional debugging:

```
def processAudioFile(audioPath):
    # Set the correct project ID
    project_id = "bcft-429709" # Replace with your actual project ID

    # Set client options, including the project ID
    client_options = ClientOptions(quota_project_id=project_id)

    # Initialize the client with the client options
    client = speech.SpeechClient(client_options=client_options)

    gcs_uri = uploadAudio(audioPath)

    audio = speech.RecognitionAudio(uri=gcs_uri)
    config = speech.RecognitionConfig(
        encoding=speech.RecognitionConfig.AudioEncoding.LINEAR16,
        sample_rate_hertz=16000,
        language_code="en-US",
    )
```

```
# Use long_running_recognize for longer audio files
operation = client.long_running_recognize(config=config, audio=audio)

# Wait for the operation to complete
response = operation.result(timeout=900)

# Print the full response for debugging
print(response)

return response
```

Try these steps and see if they help resolve the issue. Let me know if you need further assistance!

Learn more: [1. codelabs.developers.google.com](https://codelabs.developers.google.com) [2. colab.research.google.com](https://colab.research.google.com)
[3. codelabs.developers.google.com](https://codelabs.developers.google.com) [4. cloud.google.com](https://cloud.google.com) [5. cloud.google.com](https://cloud.google.com) [6. cloud.google.com](https://cloud.google.com)
[7. github.com](https://github.com) [8. github.com](https://github.com) [9. r9y9.github.io](https://r9y9.github.io) [10. github.com](https://github.com) [11. www.dropbox.com](https://www.dropbox.com)
[12. www.dropbox.com](https://www.dropbox.com)

You

How can I find out the audio encoding and format of my audio on the web>?

Copilot

To find out the audio encoding and format of your audio file, you can use various tools and websites. Here are a few methods:

1. Online Tools:

- Websites like MediaInfo allow you to upload your audio file and get detailed information about its encoding, format, bit rate, sample rate, and more.

2. Command Line Tools:

- If you have access to a command line interface, you can use tools like ffmpeg or sox to get detailed information about your audio file. For example, with ffmpeg, you can use the following command:

```
ffmpeg -i your_audio_file.m4a
```

- This will display detailed information about the audio file, including its encoding and format.

2. Audio Software:

- Software like Audacity or VLC Media Player can also provide detailed information about your audio files. In VLC, you can go to Tools > Media Information to see the codec and format details.

3. Google Cloud Documentation:

- Google Cloud's [Introduction to audio encoding for Speech-to-Text](#) provides detailed information about different audio encodings and formats¹.

Using these methods, you should be able to determine the encoding and format of your audio file. Let me know if you need any further assistance!

Learn more: [1. cloud.google.com](https://cloud.google.com) [2. en.wikipedia.org](https://en.wikipedia.org) [3. en.wikipedia.org](https://en.wikipedia.org) [4. support.serato.com](https://support.serato.com)

You

How can I insert into a new word document using python-docx?

Copilot

To insert content into a new Word document using the python-docx library, you can follow these steps:

1. **Install the python-docx library:** If you haven't already installed the library, you can do so using pip:

```
pip install python-docx
```

2. **Create a new Word document and add content:** Here's a basic example of how to create a new Word document and add a heading, a paragraph, and a table:

```
from docx import Document
from docx.shared import Inches

# Create a new Document
document = Document()

# Add a heading
document.add_heading('Document Title', level=1)

# Add a paragraph
document.add_paragraph('This is a paragraph in the document.')

# Add a table
table = document.add_table(rows=2, cols=2)
table.cell(0, 0).text = 'Cell 1,1'
table.cell(0, 1).text = 'Cell 1,2'
table.cell(1, 0).text = 'Cell 2,1'
table.cell(1, 1).text = 'Cell 2,2'

# Add an image
document.add_picture('path/to/image.png', width=Inches(1.25))

# Save the document
document.save('example.docx')
```

3. **Run the script:** Execute the script to generate the Word document with the specified content.

This example demonstrates the basic functionality of the python-docx library. You can add more complex elements like styled text, lists, and more by exploring the [python-docx documentation](#)¹.

Feel free to ask if you have any specific requirements or need further assistance!

Learn more: [1. python-docx.readthedocs.io](#) [2. www.youtube.com](#) [3. pythoninoffice.com](#) [4. www.youtube.com](#) [5. www.geeksforgeeks.org](#) [6. github.com](#) [7. python-docx.readthedocs.io](#) [8. blog.finxter.com](#) [9. geeksforgeeks.org](#)

You

I have an audio filename such as myAudio1.mp3 or myAudio2.m4a, how can I replace the bit after the dot with docx using string methods in python?

Copilot

You can easily replace the file extension in a filename using Python's string methods. Here's a simple way to do it:

```
# Example filenames
filename1 = "myAudio1.mp3"
filename2 = "myAudio2.m4a"

# Function to replace the extension
def replace_extension(filename, new_extension):
    # Split the filename into name and extension
    name, _ = filename.rsplit('.', 1)
    # Return the new filename with the new extension
    return f'{name}.{new_extension}'

# Replace the extensions
new_filename1 = replace_extension(filename1, "docx")
new_filename2 = replace_extension(filename2, "docx")

print(new_filename1) # Output: myAudio1.docx
print(new_filename2) # Output: myAudio2.docx
```

This function works by splitting the filename at the last dot (.) and then reconstructing the filename with the new extension. You can use this function to replace any file extension with .docx or any other extension you need.

Learn more: [1. stackoverflow.com](https://stackoverflow.com) [2. stackoverflow.com](https://stackoverflow.com) [3. discuss.python.org](https://discuss.python.org) [4. zetcode.com](https://zetcode.com)
[5. en.wikipedia.org](https://en.wikipedia.org) [6. en.wikipedia.org](https://en.wikipedia.org) [7. support.serato.com](https://support.serato.com) [8. cloud.google.com](https://cloud.google.com)

[Conversation by Fifa Utuber with Copilot on Tuesday, Sep 3 2024](#)