**Voice Assistant Script Documentation**

**Overview**

This Python script implements a simple voice assistant capable of recognizing spoken commands and performing tasks such as searching Wikipedia, opening websites, singing a song, and telling the current time. The assistant uses text-to-speech and speech recognition libraries to interact with the user.

**Dependencies**

To run this script, you need to install the following Python libraries:

* pyttsx3 for text-to-speech conversion
* speech\_recognition for recognizing spoken commands
* datetime for retrieving the current time
* wikipedia for searching Wikipedia
* webbrowser for opening websites

You can install the required libraries using pip:

**Bash code:**

pip install pyttsx3 SpeechRecognition wikipedia

**Script Breakdown**

**Initialization**

**Python Code:**

import pyttsx3

import speech\_recognition as sr

import datetime

import wikipedia

import webbrowser

engine = pyttsx3.init('sapi5')

voices = engine.getProperty('voices')

engine.setProperty('voice', voices[0].id) # 0 for male voice, 1 for female voice

This section imports the necessary libraries and initializes the text-to-speech engine with a selected voice.

**Functions**

**speak(audio)**

**Python Code:**

def speak(audio):

engine.say(audio)

engine.runAndWait()

This function converts text to speech and plays the audio.

**wishMe()**

**Python Code:**

def wishMe():

hour = datetime.datetime.now().hour

if 0 <= hour < 12:

speak("Good Morning!")

elif 12 <= hour < 18:

speak("Good Afternoon!")

else:

speak("Good Evening!")

speak("I am your assistant. How can I help you?")

This function greets the user based on the current time of day and introduces the assistant.

**takeCommand()**

**Python Code:**

def takeCommand():

r = sr.Recognizer()

with sr.Microphone() as source:

print("Listening...")

r.pause\_threshold = 1 # pause threshold for listening

audio = r.listen(source)

try:

print("Recognizing...")

query = r.recognize\_google(audio, language='en-in')

print(f"User said: {query}\n")

except sr.UnknownValueError:

speak("Sorry, I did not understand that.")

return "None"

except sr.RequestError:

speak("Sorry, the service is down.")

return "None"

except Exception as e:

print(e)

speak("Sorry, I could not recognize your voice.")

return "None"

return query

This function listens to the user's voice, processes it using Google's speech recognition, and returns the recognized text. It includes error handling for different exceptions.

**singASong()**

**Python Code:**

def singASong():

lyrics = "Twinkle, twinkle, little star, How I wonder what you are. Up above the world so high, Like a diamond in the sky."

speak("Alright, here's a snippet of the song for you.")

speak(lyrics)

This function makes the assistant sing a snippet of the song "Twinkle, Twinkle, Little Star."

**Main Program**

**Python Code:**

if \_\_name\_\_ == "\_\_main\_\_":

wishMe()

while True:

query = takeCommand().lower()

if 'wikipedia' in query:

speak('Searching Wikipedia...')

query = query.replace("wikipedia", "")

try:

results = wikipedia.summary(query, sentences=2)

speak("According to Wikipedia")

print(results)

speak(results)

except wikipedia.exceptions.DisambiguationError as e:

speak("There are multiple results for this query. Please be more specific.")

except wikipedia.exceptions.PageError:

speak("Sorry, I couldn't find any results for your query.")

except Exception as e:

print(e)

speak("An error occurred while searching Wikipedia.")

elif 'open youtube' in query:

speak("Opening YouTube")

webbrowser.open("youtube.com")

elif 'open google' in query:

speak("Opening Google")

webbrowser.open("google.com")

elif 'sing a song' in query:

singASong()

elif 'the time' in query:

strTime = datetime.datetime.now().strftime("%H:%M:%S")

speak(f"Sir, the time is {strTime}")

elif 'bye' in query or 'exit' in query:

speak("Goodbye! Have a nice day.")

break

else:

speak("I'm not sure how to help with that.")

The main program starts by greeting the user. It then enters an infinite loop where it listens for commands and performs actions based on the recognized text. It handles various commands such as searching Wikipedia, opening websites, singing a song, telling the current time, and exiting the program.

**Usage**

1. Ensure your microphone is connected and working.
2. Run the script using a Python interpreter.
3. The assistant will greet you and wait for your commands.
4. Speak clearly into the microphone to issue commands.
5. To exit the assistant, say "bye" or "exit."

**Enhancements and Customization**

You can expand the assistant's capabilities by adding more commands and corresponding actions in the main program loop. For example, you can add commands to open specific websites, provide weather updates, set reminders, and more.