Python Chat Bot

Getting started:

Install Thonny from https://thonny.org/ if you don't already have it on your PC

In Thonny's Tool section (see along the top), search for and install the following packages:

- PyAudio – this allows your python script to use your microphone
- SpeechRecognition – this enables python to recognise speech (only works with internet)
- pyttsx3 – this translates text to speech

Chat Bot Version 1.0

Let's get a basic chat bot working:

```
import speech_recognition as sr
import pyttsx3

# Initialize the recognizer
r = sr.Recognizer()

# Function to convert text to speech
def SpeakText(command):
    engine = pyttsx3.init()
    engine.say(command)
    engine.runAndWait()

# Use the microphone as source for input
with sr.Microphone() as source:

    # You should hear your Personal Assistant say the following:
    SpeakText("Hello there! What is your name?")

    # Wait for a second to let the recognizer adjust the energy threshold based on the surrounding noise level
    r.adjust_for_ambient_noise(source, duration=1)

    # Listen for the user's input
    audio = r.listen(source)

    # Using google to recognize audio
    name = r.recognize_google(audio)

    print("Hi " + name +  "!")

    SpeakText("It is nice to meet you " + name)
```

This has all the basic ingredients.

We load the packages and initialise the recognizer:

```
import speech_recognition as sr
import pyttsx3

# Initialize the recognizer
r = sr.Recognizer()
```

We create a function which allows us to use pyttsx3 to convert text to speech i.e. your PC will speak out any text we provide to it.

```
# Function to convert text to speech
def SpeakText(command):
    engine = pyttsx3.init()
    engine.say(command)
    engine.runAndWait()
```

And now for the main part of our script…

We firstly tell the recognizer to use our microphone:

```
# Use the microphone as source for input
with sr.Microphone() as source:
```

We then have our chat bot speak to us:

```
# Use the microphone as source for input
with sr.Microphone() as source:

    # You should hear your Personal Assistant say the following:
    SpeakText("Hello there! What is your name?")
```

Now using our microphone as a source, we firstly listen to determine how much surrounding noise there is in the room:

```
# Use the microphone as source for input
with sr.Microphone() as source:

    # You should hear your Personal Assistant say the following:
    SpeakText("Hello there! What is your name?")

    # Wait for a second to let the recognizer adjust the energy threshold based on the surrounding noise level
    r.adjust_for_ambient_noise(source, duration=1)
```

Now we wait for the user to say something:

```
    # Wait for a second to let the recognizer adju
    r.adjust_for_ambient_noise(source, duration=1)

    # Listen for the user's input
    audio = r.listen(source)
```

And then we send the audio to Google(!) who will interpret what the user said:

```
# Listen for the user's input
audio = r.listen(source)

# Using google to recognize audio
name = r.recognize_google(audio)
```

Finally, we print out what the user said and we pass it to our SpeakText() function for our PC to speak:

```
# Using google to recognize audio
name = r.recognize_google(audio)

print("Hi " + name +  "!")

SpeakText("It is nice to meet you " + name)
```

Save your script with version number 1.0.

Notes:

This basic chat bot script will work but only when the Google interpreter can recognize the audio we are sending it. Otherwise, it will produce an error. In the next version, we will add some error handling.

Chat Bot Version 1.1

Let's add some error handling to our script. We do this by using the python **try** statement (see https://www.w3schools.com/python/python_try_except.asp for more infomration).

Basically, the **try** statement allows us to "try" something and if it works, then great. However, if it does not work and produces an error, we can recognize this and do something about it.

Load your chat bot version 1.0 script and save it as chat bot version 1.1

Add the following variables, as follows:

```
import speech_recognition as sr
import pyttsx3

# Initialize the recognizer
r = sr.Recognizer()
validAudio = False
i = 0
attempts = 3
```

Do not change our SpeakText() function definition and instead go directly to the main section of our script.

Add the following changes, which just include some additional speech for the chat bot:

```
# Use the microphone as source for input
with sr.Microphone() as source:

    # You should hear your Personal Assistant say the following:
    SpeakText("Hello there! I am Python Chat Bot. You can call me PeeCeeBee.")

    # Wait for a second to let the recognizer adjust the energy threshold
    # based on the surrounding noise level
    print("PeeCeeBee is now listening.....")
    r.adjust_for_ambient_noise(source, duration=1)

    # You should now hear your Personal Assistance ask for your name
    SpeakText("What is your name?")
```

We are now going to add a **while** loop which will allow us to control the amount of attempts we want the user to try if the script cannot recognize their speech and creates an error, as mentioned earlier. Once all attempts have been used up, our script will stop.

Here is the start of the while loop:

```
while i < attempts:

    # Counting up the attempts
    i += 1

    # Listen for the user's input
    audio = r.listen(source)

    # Check if the audio is recognisable
    try:
        # Using google to recognize audio
        name = r.recognize_google(audio)
        validAudio = True
    except sr.UnknownValueError:
        print("Could not understand audio")
        validAudio = False
    except sr.RequestError as e:
        print("Recog Error; {0}".format(e))
        break
```

Here we count the attempts

Here we listen for the user's input

Here we introduce error handling!

Can you understand what the **try** statement is doing? It takes what the user said, sends it to Google, and if Google does not create an error, it allows the script to continue. However, if Google cannot understand what the user said, it will create an "UnknownValueError", which we need to allow for. Also, a "RequestError" may arise if you do not have an internet connection. We also want to allow for this.

Now let's use the error handling from the **try** statement, as follows:

```
    validAudio = False
except sr.RequestError as e:
    print("Recog Error; {0}".format(e))
    break

# If audio is recognised...
if validAudio == True:
    print("Hi " + name +  "!")
    SpeakText("It is nice to meet you " + name)
    break
# If audio is not recognised...
elif validAudio == False and i < attempts:
    SpeakText("Sorry. I did not quite understand that. Please try again.")
```

If the **try** statement does not see any error (validAudio == True) then it will allow the script to print and speak what the user said, and before stopping the script with the **break** command.

But, if the try statement does see an error (validAudio == False) it will let the user know and using the while loop allow the user to speak again.

The while loop will continue allowing the user to speak again after an error until all attempts have been used:

```
    if validAudio == True:
        print("Hi " + name +  "!")
        SpeakText("It is nice to meet you " + name)
        break
    # If audio is not recognised...
    elif validAudio == False and i < attempts:
        SpeakText("Sorry. I did not quite understand that. Plea

# No more attempts....
else:
    SpeakText("Sorry. There seems to be a problem. Good bye.")
```

Note: watch out for the correct indentation for the last lines above – the **else** should be on the same indentation level as the initial **while** statement.

Here is the complete main section:

```
# Use the microphone as source for input
with sr.Microphone() as source:

    # You should hear your Personal Assistant say the following:
    SpeakText("Hello there! I am Python Chat Bot. You can call me PeeCeeBee.")

    # Wait for a second to let the recognizer adjust the energy threshold
    # based on the surrounding noise level
    print("PeeCeeBee is now listening.....")
    r.adjust_for_ambient_noise(source, duration=1)

    # You should now hear your Personal Assistance ask for your name
    SpeakText("What is your name?")

    while i < attempts:

        # Counting up the attempts
        i += 1

        # Listen for the user's input
        audio = r.listen(source)

        # Check if the audio is recognisable
        try:
            # Using google to recognize audio
            name = r.recognize_google(audio)
            validAudio = True
        except sr.UnknownValueError:
            print("Could not understand audio")
            validAudio = False
        except sr.RequestError as e:
            print("Recog Error; {0}".format(e))
            break

        # If audio is recognised...
        if validAudio == True:
            print("Hi " + name +  "!")
            SpeakText("It is nice to meet you " + name)
            break
        # If audio is not recognised...
        elif validAudio == False and i < attempts:
            SpeakText("Sorry. I did not quite understand that. Please try again.")

    # No more attempts....
    else:
        SpeakText("Sorry. There seems to be a problem. Good bye.")
```

Save your script with version number 1.1.

Notes:

While we have introduced error handling into our chat bot script, there is still an issue if the bot cannot recognize any user input at all, for example if the user speaks too quietly. In our next version, we will introduce a way to handle this problem.