Talking-Pi

Personalization Manual

Index

1. Introduction

Dear customer,

Thank you for choosing our product.

The own programming and modification of the system is an essential and important part of this project.

In the following steps, we will explain step by step how to set up the Google Assistant API, make your Talking-Pi ready for use, and how to program and modify your Talking-Pi according to your own needs.

If, however, problems should occur while working with your Talking-Pi, please feel free to contact us.

Current instructions, projects and a community forum can also be found here:

Talking-Pi Website

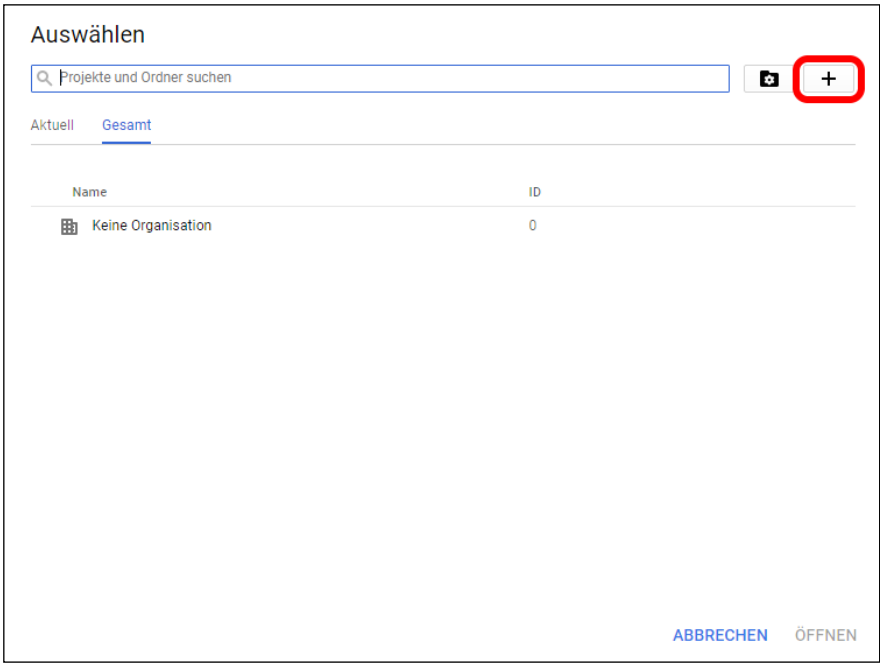
2. Setting up the Cloud Speech API

With our pre-installed Talking-Pi operating system, the Talking-Pi waits patiently to be activated.
You can download the image here.
Install the image on a micro SD card (a capacity of at least 8GB is recommended) and then insert it into your Raspberry Pi.

With a modified Cloud Speech API, 80 languages are available.
You can also enter your own wake words and commands in the system.

Attention! The Google Cloud Speech API is free of charge if you use it for less than 60 minutes per month. Otherwise, you will be charged $0.06 per 15 seconds. However, you will receive a notification if you exceed the 60-minute limit.

First, open up the Google Cloud Console.
Log in with your Google Account or create a new one if you don't already have one.
Open the project overview and create a new project.



Next, select the option "Billing" from the left menu.

Add a new payment method and follow the instructions.

After adding a new billing option, return to the billing menu and make sure your newly created project is linked to this payment option.

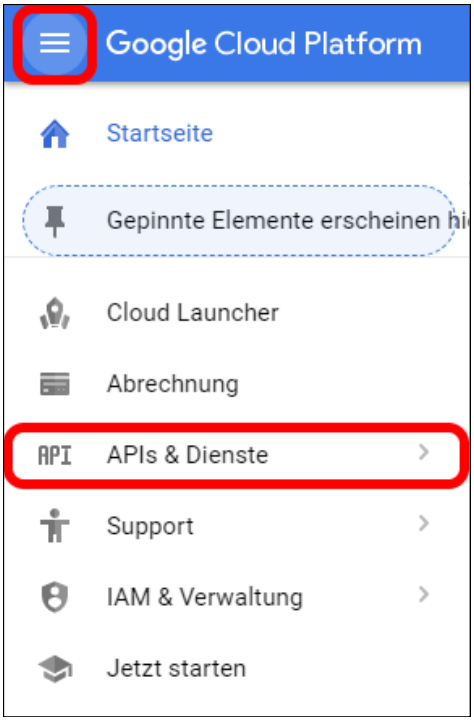
To link your project to a payment option, you can simply select the three items.

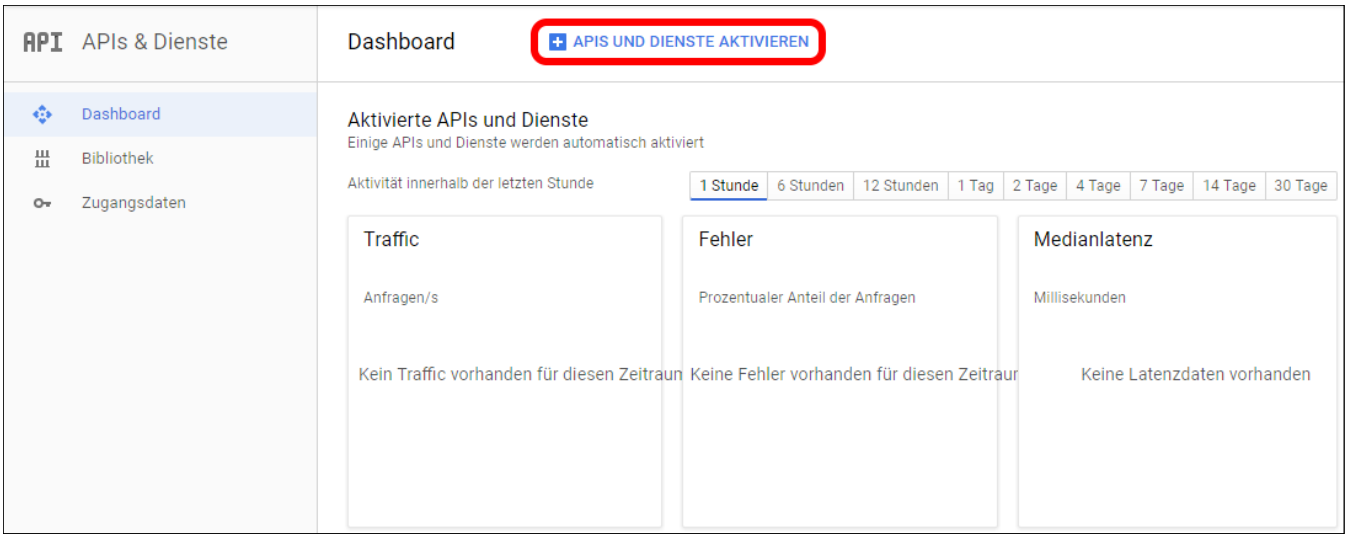
Next, the required services are activated.

Use the icon marked in the picture to add a new project to your list.

Then simply enter any name for your project and complete the process.

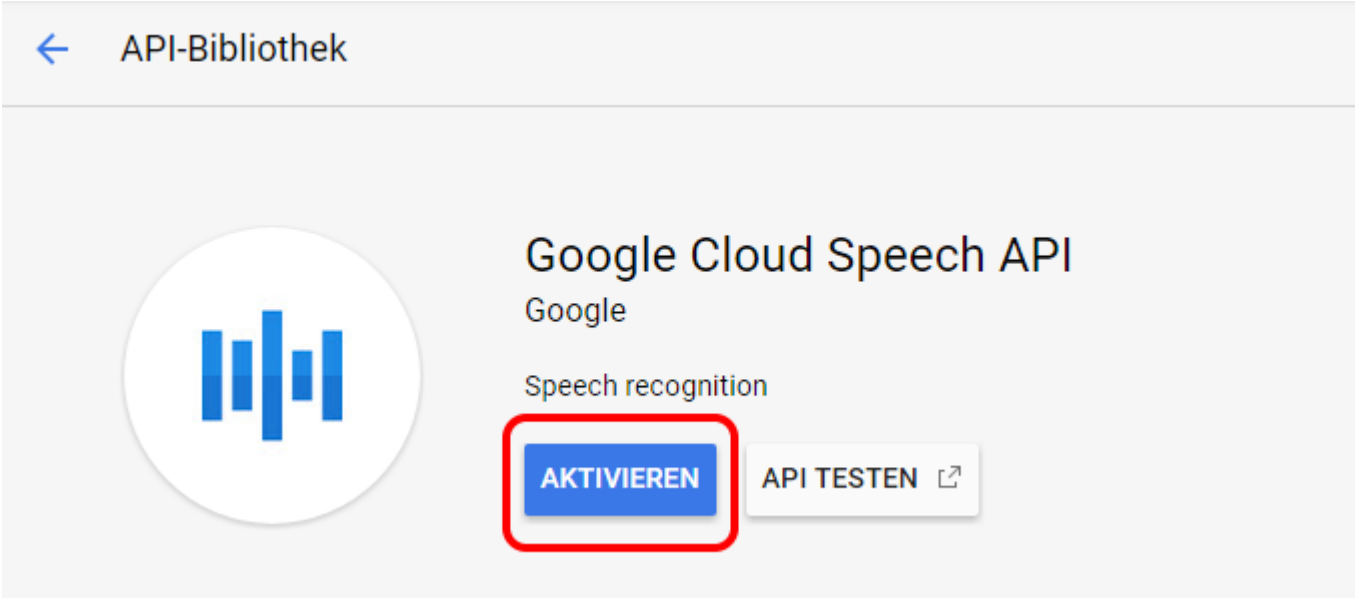
Then open the menu and select the option "**APIs & Services**" and activate the menu.


Google Cloud Platform
Startseite
Gepinnte Elemente erscheinen hi
Cloud Launcher
Abrechnung
APIs & Dienste
Support
IAM & Verwaltung
Jetzt starten

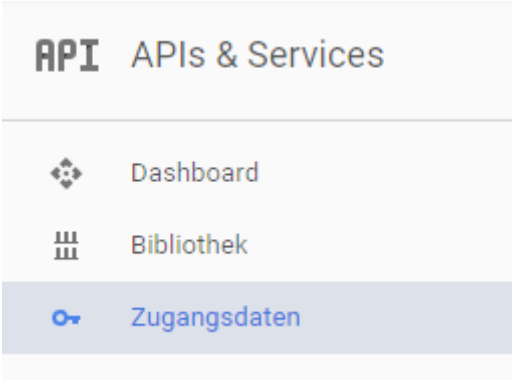

APIs & Dienste
Dashboard
Bibliothek
Zugangsdaten
Dashboard
APIS UND DIENSTE AKTIVIEREN
Aktivierte APIs und Dienste
Einige APIs und Dienste werden automatisch aktiviert
Aktivität innerhalb der letzten Stunde
1 Stunde
6 Stunden
12 Stunden
1 Tag
2 Tage
4 Tage
7 Tage
14 Tage
30 Tage
Traffic
Anfragen/s
Kein Traffic vorhanden für diesen Zeitraun
Fehler
Prozentualer Anteil der Anfragen
Keine Fehler vorhanden für diesen Zeitraur
Medianlatenz
Millisekunden
Keine Latenzdaten vorhanden

In the API library that now opens, search for the term "**Google Cloud Speech API**".

Select the found API and activate it.

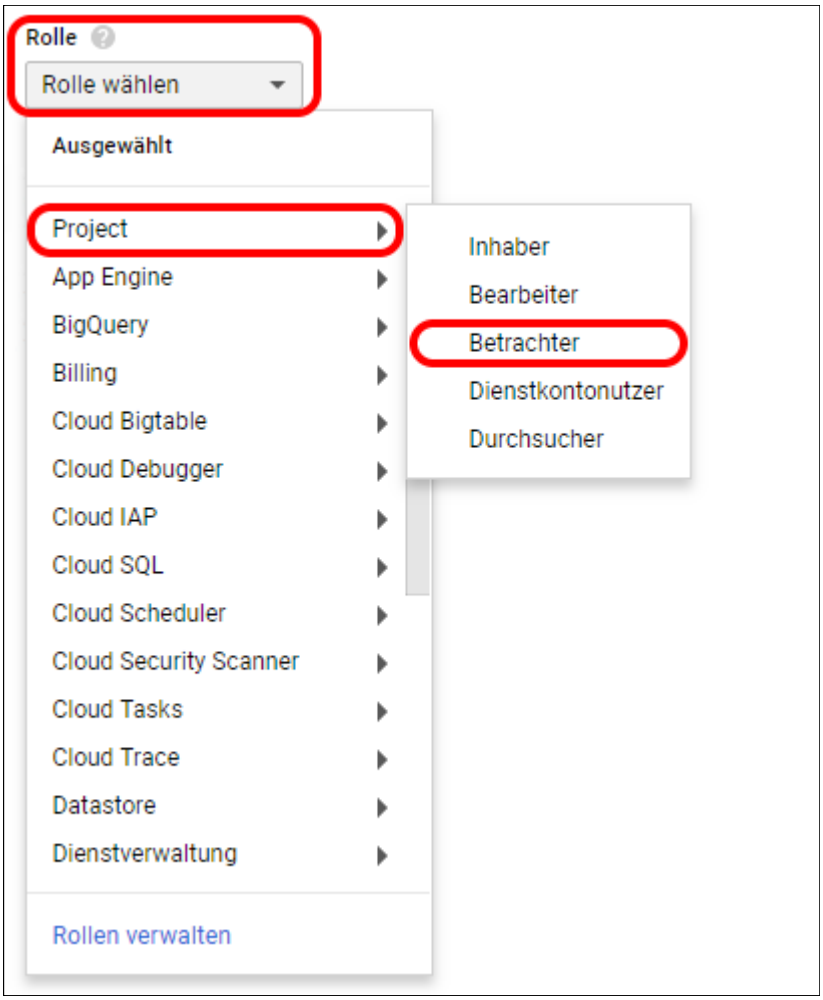


Now select the item "**Access data**" in the "**API & Services**" overview.

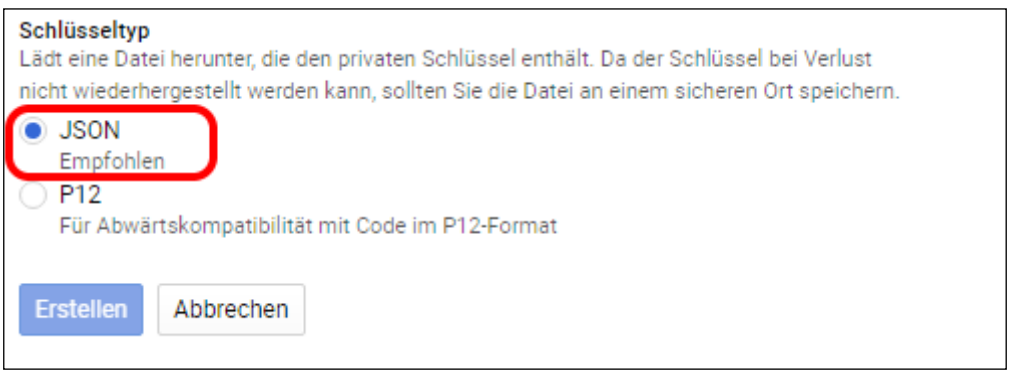


Select "**Create login data**" and create a "**Service account key**".

Here you define the project and viewer as roles:



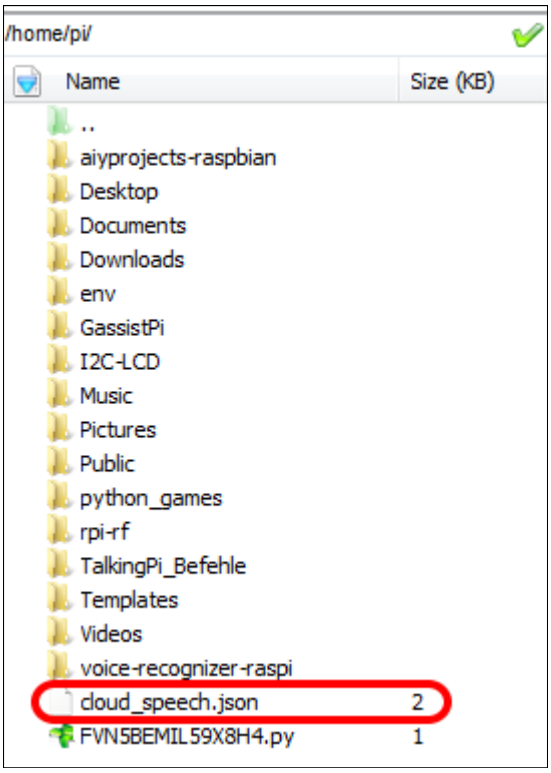
Select "**JSON**" as the key type.



Now create your access data by clicking on "**Create**".

The download of your data starts automatically.

Rename the downloaded file to **cloud_speech. json** and copy it to the user directory of your TalkingPis.



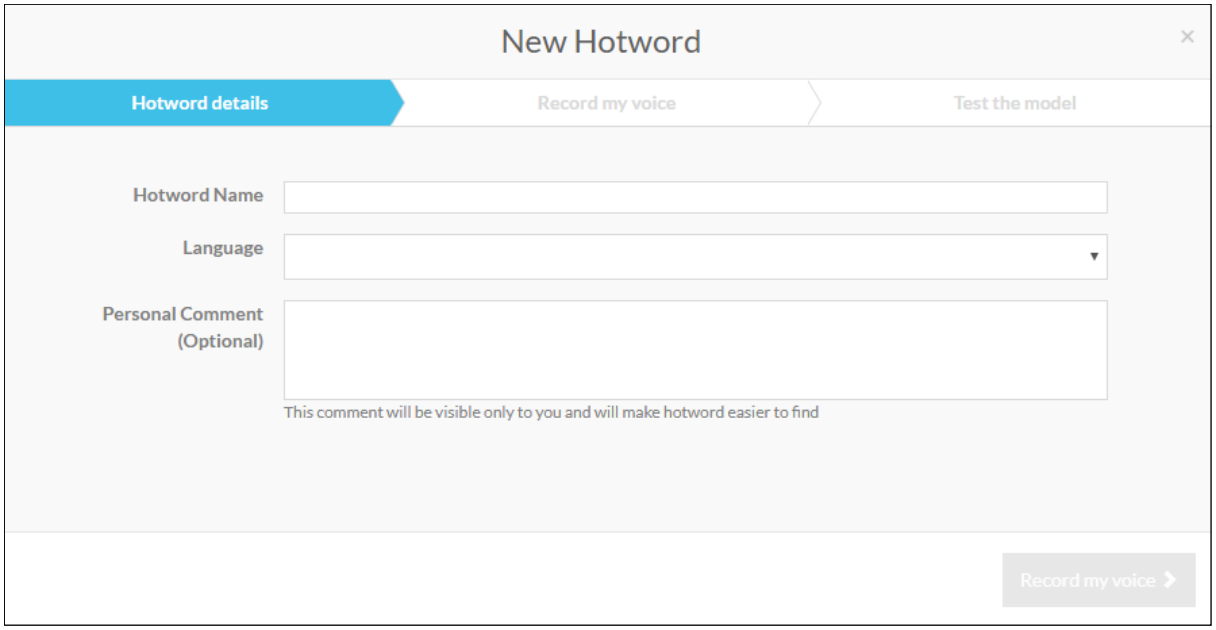
After a restart, the Cloud Speech API is ready for use on your TalkingPi.

3. Custom Wake-Words

The activation can be done either by pressing the button or by a so-called wake-word, which has been integrated into the system before.
In the system we have prepared, the talking pi reacts to the two Wake-Words "Talking-Pi" and "Alexa".

If these two wake-words are not enough for you, or if you want to personalize your Talking-Pi, you can of course also teach your talking-pi your own wake-words.
For user-defined wake-word detection we use the Gassist Pi Library in combination with the Snowboy Hotword Detection.
These two libraries are of course already installed on our prepared operating system.

For the user-defined wake-words you can either record your own hotword or use an existing wake-word in the Snowboy Dashboard.

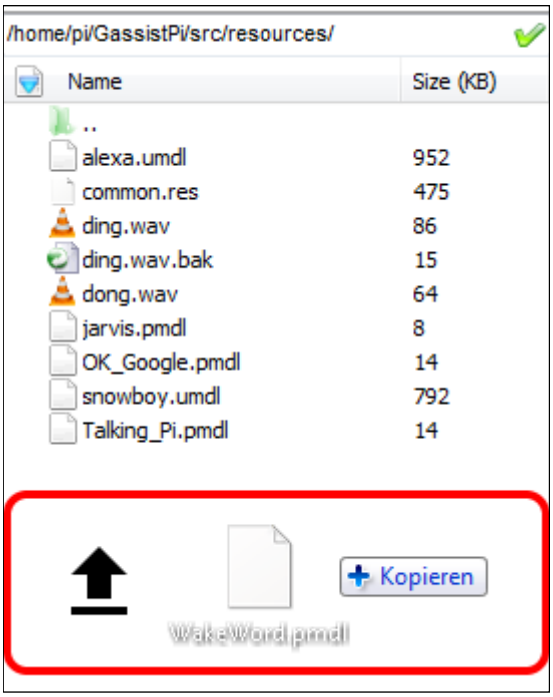


You record the wake word three times to adapt it to your voice.

Afterwards, you can download the .pmdl file that matches your wake word.

Then copy this file to the following folder of your Talking-Pi:

/home/pi/GassistPi/src/resources/



In order to activate the wake-word in your talking-pi it is necessary to modify the configuration file of the Gassist Pi library.

To do this, type the following command into a terminal on your Talking-Pi:

```
sudo nano /home/pi/GassistPi/src/talkingPi_snowboy.py
```

As in the following example, modify the **models** array to add or remove your own wake word configuration files.

```
models = ['/home/pi/GassistPi/src/resources/alexa.umdl', '/home/pi/GassistPi/
src/resources/Talking_Pi.pmdl', 'home/pi/GassistPi/src/resources/
[mywakeword].pmdl']
```

Please note that the phrase "[mywakeword]" is a placeholder to be replaced by the filename of your own wake word.

You can then save your changes with the key combination **CTRL+O** and exit the editor with the combination **CTRL+X.**
After restarting the system, your personal wake word should be integrated into the system and the Talking-Pi should react to its new command.

4. Custom Commands

To add your own voice commands, which the Talking-Pi recognizes as a separate command, it is necessary to first define the command as an expected input.

This increases the detection rate.

Open the file "**googlecloudtalkingpi.py**" with the following command:

```
sudo nano /home/pi/voice-recognizer-raspi/src/googlecloudtalkingpi.py
```

Add the following **recognizer. expect_phrase** command to the existing phrases:

```
recognizer.expect_phrase('[Ihr eigenes Sprachkommando]')
```

Replace the placeholder with your own voice command.

```
def main():
    # Definition aller zu erwartenden Befehle
    # Diese Erwartungen erhoehen die Erkennungsgenauigkeit des Talking-Pis
    # Aehnlich klingende Befehle, leichte Abweichungen oder auch Akzente in der Sprache
    # koennen durch die Definierung der Erwartungen besser erkannt werden
    recognizer = aiy.cloudspeech.get_recognizer()
    # Sie koennen hier weitere Phrasen hinzufuegen!
    recognizer.expect_phrase('mach das Licht an')
    recognizer.expect_phrase('mach das Licht aus')
    recognizer.expect_phrase('mach das Licht oben an')
    recognizer.expect_phrase('mach das Licht oben aus')
```

Now that we have defined the expected speech input, it is now possible to define the command from actual recognition.

To do this, add the following content to the existing definitions as the file is opened.

```
elif '[Ihr eigenes Sprachkommando]' in text:
      subprocess.call(["python","/home/pi/I2C-LCD/EigenerBefehl.py"])
      subprocess.call(["bash","/home/pi/TalkingPi_Befehle/Receiver/
EigenerBefehl"])
     subprocess.Popen(["aplay", "/home/pi/GassistPi/src/resources/dong.wav"],
stdin=subprocess.PIPE, stdout=subprocess.PIPE, stderr=subprocess.PIPE)
      time.sleep(2)
```

Please be sure to replace the placeholder with your own command.

You may also need to correct the indentations after copying.

To do this, simply refer to the existing command definitions.

```
print('You said "', text, '"')
# Pruefe auf folgenden Befehl
if 'mach das Licht an' in text:
    # Wenn der Befehl passt, dann fuehre die folgenden Befehls-Dateien aus
    if Display:
        # Der folgende Befehl fuer das Display wird nur ausgefuert wenn der entsprechende Wert oben gesetzt wurde. Dieser ist standardmaessig aktiv.
        subprocess.call(["python","/home/pi/I2C-LCD/Befehl1.py"])
    subprocess.call(["bash","/home/pi/TalkingPi_Befehle/Licht/LichtAn.sh"])
    # Spiele ausserdem folgende Tondatei ab
    subprocess.Popen(["aplay", "/home/pi/GassistPi/src/resources/dong.wav"], stdin=subprocess.PIPE, stdout=subprocess.PIPE, stderr=subprocess.PIPE)
    # Warte danach 2 Sekunden
    time.sleep(2)
elif 'mach das Licht aus' in text:
    if Display:
        subprocess.call(["python","/home/pi/I2C-LCD/Befehl2.py"])
    subprocess.call(["bash","/home/pi/TalkingPi_Befehle/Licht/LichtAus.sh"])
    subprocess.Popen(["aplay", "/home/pi/GassistPi/src/resources/dong.wav"], stdin=subprocess.PIPE, stdout=subprocess.PIPE, stderr=subprocess.PIPE)
    time.sleep(2)
```

Within the newly defined command calls, three different files are called.

The first two files ("/home/pi/I2C-LCD/EigenerBefehl. py" and "/home/pi/TalkingPi_Befehle/Receiver/ EigenerBefehl") define the actions that are called when the corresponding voice command is executed.

In this example, the first file represents the output on the display and the second file represents the actual command sequence.

The third file defined the sound that will be output as soon as the voice command is detected.

You can of course enter as many files as you like here.

The structure of the three files (display, command, sound) created by us serves only as an overview.

Please note that the files to be executed must still be created by you in the corresponding folders.

You can of course define as many files as you need.

In our standard configuration we have defined the following files for execution.

These can be used for your personal configuration.

Display-Control:

File location: /home/pi/I2C-LCD/[YourFile].py

Example:

```
import lcddriver
from time import *

lcd = lcddriver.lcd()
lcd.lcd_clear()

lcd.lcd_display_string("+++   Ausgabe  +++", 1)
lcd.lcd_display_string("+Ich bin eine Ausgabe+", 2)
```

Sound-Output:

File location: /home/pi/GassistPi/src/resources/[IhreTonDatei].wav

Other executable files:

File location: /home/pi/TalkingPi_Befehle/

File format: .sh

Note: The individual command files are subdivided into the following folders: Projector, Light, Computer and Receiver.

You can of course add more categories.

However, please make sure that you specify them correctly when referencing files.

After restarting your Talking-Pi, your new voice command should be recognized and processed successfully.

5. Support

We also support you after your purchase.

If there are any questions left or if you encounter any problems, feel free to contact us by mail, phone or by our ticket-supportsystem on our website.

E-Mail: service@joy-it.net

Ticket-System: http://support.joy-it.net

Phone: +49 (0)2845 98469 – 66 (11- 18 Uhr)

Please visit our website for more informations:

www.joy-it.net