Control per veu de una Raspberry Pi

19 de juny del 2019 Albert Orteu Estivill

Índex

Resum

El control per veu es cada dia més estès com per exemple amb Amazon Alexa. Això es relaciona també amb la demòtica que serà popularitzada l'any que ve per diverses multinacionals com IKEA.…También es pot veure aquest projecte com a una substitució de un comandament a distància IR.

El projecte que presentem en aquesta memòria consisteix amb en el control de una RPI desenvolupament de codi en python3 i l'ús d'Alexa en el núvol.

1. Prefaci

El projecte volia ser gradual: Primer amb llibreria local i sense connexió Internet i després usant els serveis de Amazon.

1. El reconeixement de veu presentant per pantalla “print” lo reconegut.
2. Completar lo anterior però gravant a mes un fitxer .wav
3. Completar lo anterior però reproduint per altaveus o auriculars
4. Completar lo anterior afegint la lògica per encendre o apagar un led (GPIO)
5. Modificar lo anterior on el reconeixement de veu es amb un skill de Alexa
6. Completar lo anterior amb un echo input o echo dot Alexa
7. Completar lo anterior encenent o apagant el televisor (IRC)
8. Completar lo anterior usant un lambda de Alexa.

S ha completat fins el punt 6.

2. Hardware utilitzat

- Raspberry Pi with el S.O. Raspbian (en SDcard, nomes command line, ssh activat)
- USB micròfon
- Auriculars (conectats al on-board 3.5mm audio jack que nomes es de audio out)
- Per GPIO: Breadboard, T-shape, conector cable, jumper cables, led, resistor 200
- Adicionalment per IRC: sensor Welleman wma316 trasmisor infrarrojos

3. Instal·lació hardware

Connectar el USB micròfon i auriculars a un USB. Conectar T-shape....

Connexió de Led:

GRND - LED- RESISTENCIA- GPIO025

El emissor IR no s’ha instal·lat ni provat.

4. Preparació del micròfon i altaveu

Apart del GPIO, cal instal·lar alguns paquets per el micròfon i auriculars

Averiguar el número de dispositiu del USB de gravació i reproducció

- Executar la comana “arecord –l” llista els dispositius de entrada de audio: HDMI, USB, etc.... Comparar l’execució de la comana abans i després de connectar el micròfon USB i veure com apareix una línea addicional en la resposta de la comana que reflexa la connexió del micròfon (el jack de 35 mm en el RPI és nomes de sortida)
- Executar la comana “aplay –l” llista els dispositius de sortida de audio:Jack 35 mm, HDMI, USB Comparar l’execució abans i després de connectar l’altaveu (35 mm) i veure com apareix una línea nova amb els auriculars connectats .

Comanes addicionals útils per llistar els ports de USB

- lsusb
- lsusb -t (ha d’aparèixer el micròfon un cop connectat)

Per provar el micròfon i els auriculars fem una gravació i la reproduïm (dispositiu es defineix a plughw, plughw:1,0 plughw:0,1 etc....)

- arecord --device=plughw:1,0 --format S16_LE --rate 44100 -c1 test.vaw
- aplay --device=hw:1,0 test.wav

> Nota: AVANCAT. A /usr/share/alsa/alsa.conf hi ha el dispositiu per defecte. Es pot canviar a /etc/asound.conf. I si no funciona, també es pot canviar el port de USB fins que funcioni :-)

Si la proba anterior no es gravés prou alt, setejar el nivell de captura del micròfon al màxim

- Executar la comana “alsamixer” i prémer F6 i tabulador i amb el cursos incrementar el nivell de la columna “captura”

5. Llibreries necessàries per el reconeixement de veu

- sudo apt-get install python3-pyaudio
- sudo pip3 install SpeeachRecognition

Hi ha algunes incompatibilitats de les llibreries python2 i python3 que cal tenir en compte que poden complicar la instal·lació.

6. Codi final per encendre i apagar el LED amb llibreria local

Es presenta a continuació el codi final.

Lo que fa el següent codi es enregistrar en una variable “audio” la veu i es fa un reconeixement de la variable “audio” i es determina si conte les paraules adients per encendre o apagar un led. Si les conte, llavors s’envien les comanes GPIO corresponens amb l’ordre reconeguda. (Abans d’això s’escriu també “audio” en un fitxer; això es estrictament innecessari però resulta curiós veure que lo que s’escriu en el fitxer amb el mètode write després d’aplicar Recognizer a “audio” és diferent de lo que s’obté després d’aplicar amb recognize_google a “audio”. No es va aconseguir fer funcionar el mètode listen de la llibreria speech_recognition.)

```
root@raspberrypi:/home/pi# cat speechToText_8.py
import  RPi.GPIO as GPIO
import re
import speech_recognition as sr
def contains_word(s, w):
    return (' ' + w + ' ') in (' ' + s + ' ')

r = sr.Recognizer()
m = sr.Microphone(device_index = 1, sample_rate = 48000)
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)
GPIO.setup(25,GPIO.OUT)

with m as source:
    print("SAY COMMAND IN 5s")
    audio = r.record(source, duration = 5)
    print("STOP TALKING")
with open("microphone-results.wav", "wb") as f:
    f.write(audio.get_wav_data())

try:
    command = r.recognize_google(audio,language="es-ES")
    print("La comana reconeguda es: " + command)   # recognize speech using Google Speech Recognition
    if contains_word(command,'enciende') and (contains_word(command,'LED') or contains_word(command,'led')):
        print ("Ha trovar els strings adecuats en la comana 1 encendre led")            # command is  led red on
        GPIO.output(25,GPIO.HIGH)
        print ("El led vermell s ha d encendre")
    if contains_word(command,'apaga') and (contains_word(command,'LED') or contains_word(command,'led')):
        print ("Ha trovar els strings adecuats en la comana 2 apagar led")
        GPIO.output(25,GPIO.LOW)
        print ("El led vermell s ha d apagar")
except LookupError:                             # speech is unintelligible
    print("Could not understand audio")
```

7. Execució encendre i apagar el LED amb llibreria local

ENCENDRE LED

A continuació la traça deixada per una execució

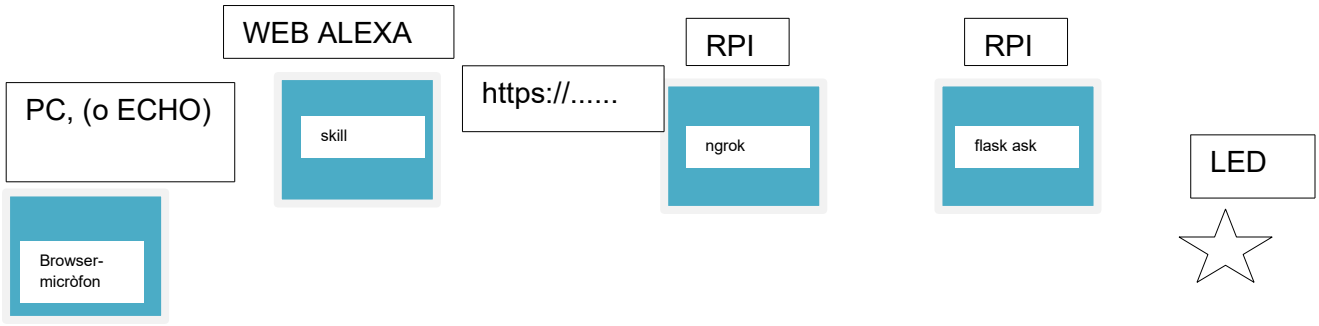
Apareixen alguns missatges d error que no son rellevants ja que tenen que veure amb altres ports USB o HDMI

```
python3 speechToText_8.py
…..
ack server is not running or cannot be started
JackShmReadWritePtr::~JackShmReadWritePtr - Init not done for -1, skipping unlock
JackShmReadWritePtr::~JackShmReadWritePtr - Init not done for -1, skipping unlock
SAY COMMAND IN 5s
STOP TALKING
La comana reconeguda es: enciende el led on
Ha trovar els strings adecuats en la comana 1 encendre led
El led vermell s ha d encendre
```

APAGAR

```
python3 speechToText_8.py
…..
JackShmReadWritePtr::~JackShmReadWritePtr - Init not done for -1, skipping unlock
SAY COMMAND IN 5s
STOP TALKING
La comana reconeguda es: apaga LED vermell
Ha trovar els strings adecuats en la comana 2 apagar led
El led vermell s ha d apagar
```

8. Arquitectura de la solució amb Alexa.



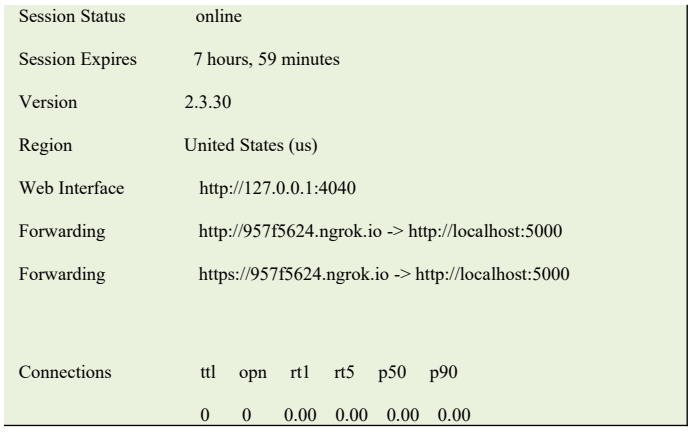
FROND END **BACK END**

9. BACK END. Llibreries que s'han de instal·lar en la RPI.

Apart de GPIO, el següent per crear un túnel i un web server.

Ngrok: paquet client que al executar-lo ens dona una URL https://.... per compartir que ens permetra que ens conectem per internet public desde qualsevol lloc a un port de la RPI. En aquest port estara installat i corrent la nostra web server

- Installar ngrok en el directori home (anar a https://ngrok.com/download triar arm, baixarlo a windows i amb winscp pasalo a /home/pi i fer un unzip) o be mes facil encara wget https://bin.equinox.io/c/4VmDzA7iaHb/ngrok-stable-linux-arm.zip
- Executar la comana ./ngrok http 5000



- Com a prova copiar la linea https que ens doni en aquell moment (valid per unes hores) https://957f5624.ngrok.io en un navegador de un pc fora de la area local. Apareixera:

Flask-Ask: es el nostre web server. Extensió de flask (web server) específic per accions de Alexa: interaccions, repeticions, accions son fàcils de programar.

- pip3 install Flask
- pip3 install flask-ask

10. 10.Codi a executar al web server. Execució

En vermell, en el codi, es resalta la part que ajuda a comprendre com funciona tot. **ledintent** es el nom de la accio (intent) que esta asociat al reconeixement de veu Raspberri Pi turn on/off

```
root@raspberrypi:/home/pi# cat led_soloStatus.py
import logging
import os

from flask import Flask
from flask_ask import Ask, request, session, question, statement
import RPi.GPIO as GPIO

app = Flask(__name__)
ask = Ask(app, "/")
logging.getLogger('flask_ask').setLevel(logging.DEBUG)

STATUSON = ['on','high']
STATUSOFF = ['off','low']

@ask.launch
def launch():
    speech_text = 'Welcome to Raspberry Pi Automation.'
    return question(speech_text).reprompt(speech_text).simple_card(speech_text)

@ask.intent('ledIntent', mapping = {'status':'status'})
def Gpio_Intent(status,room):
    GPIO.setwarnings(False)
    GPIO.setmode(GPIO.BCM)
    GPIO.setup(25,GPIO.OUT)
    if status in STATUSON:
        GPIO.output(25,GPIO.HIGH)
        return statement('turning {} lights'.format(status))
    elif status in STATUSOFF:
        GPIO.output(25,GPIO.LOW)
        return statement('turning {} lights'.format(status))
    else:
        return statement('Sorry not possible.')

@ask.intent('AMAZON.HelpIntent')
def help():
    speech_text = 'You can say hello to me!'
    return question(speech_text).reprompt(speech_text).simple_card('HelloWorld', speech_text)


@ask.session_ended
def session_ended():
    return "{}", 200


if __name__ == '__main__':
    if 'ASK_VERIFY_REQUESTS' in os.environ:
        verify = str(os.environ.get('ASK_VERIFY_REQUESTS', '')).lower()
        if verify == 'false':
            app.config['ASK_VERIFY_REQUESTS'] = False
    app.run(debug=True)
```

Execució:

- python3 led_soloStatus.py

11. FRONT END. Definició de la interfície per Alexa

Tot aixo es converteix en un FORMULARI SKILL en la web amazon de alexa. No hi ha codi.

Ens calen 2 coses:

1- Unes frases que faran referència a unes accions
2- Una adressa htttp on s adressaran aquestes accions

Les frases:

Alexa open Raspberry PI and turn led ON
Alexa open Raspberry PI and turn led OFF
Estructura:

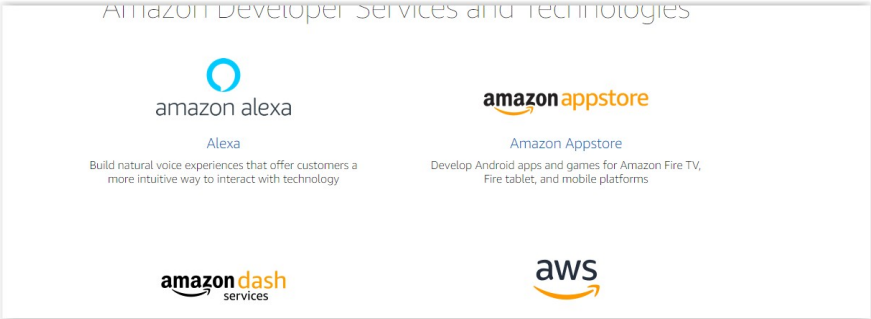
Paraules reservades Alexa i open (open, do, make etc....)
Invocacio Raspberry PI
Accio: turn led (parametre es diu “intent”, en angles es un fals amic)
Variable de la accio (“slot”): ON y OFF
“Sample uterance” son mostres de expresions similars que un pot dir que sempre desencadenaran la mateixa accio: turn led on / turn light on / please turn on / etc.... (despres Alexa pot deduir que fer)

La addresa:

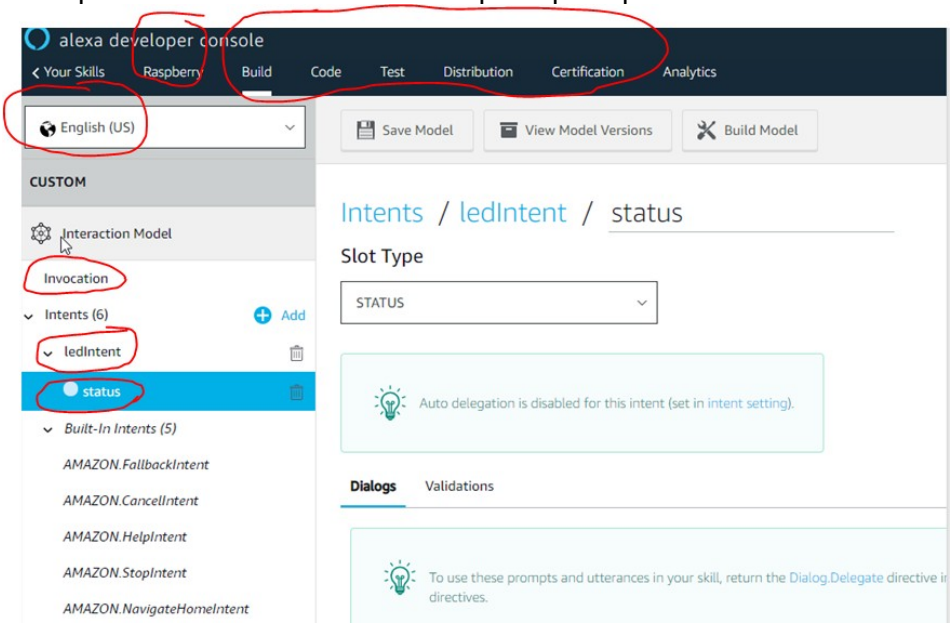
La que hem obtingut amb ngrok.

12. FRONT END. Skill a Amazon Cloud pas a pas

1- Crear un compte a Amazo Developer. https://developer.amazon.com/?



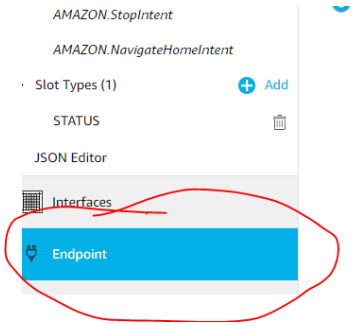
2. Entrar a Amazon Alexa
3. Click en Crear Alexa Skill
4. Omplir el formulari. En vermell les parts principals del resultat final

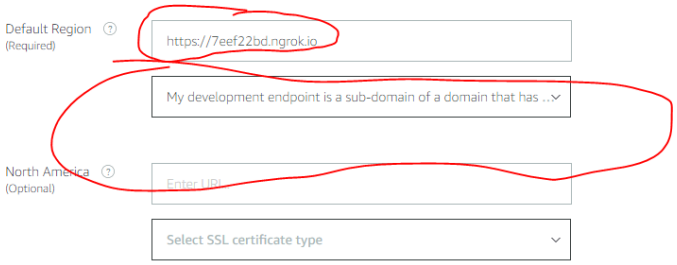


1. Enter a skill name. In this example, it's "raspberry pi".
2. Make sure "Custom" is selected then click on "Create skill".
3. Make sure "Start from scratch" is selected then click on "Choose".
4. Click on "+" icon next to "Slot types" in the left pane.
5. Enter "STATUS" and click on "Create custom slot type" button.
6. Add "on" and "off" as the slot values.
7. Click on "+" icon next to "Slot types" in the left pane again.
8. Enter "ledIntent" as an intent name, then click on "Create custom intent".
9. Enter below as a Sample Utterances, then click on "+" button turn the light {status}.

10. Select “STATUS” types status slots respectively.
11. Click on “Build Model” button on the top. If everything goes right, you’ll get a “Build Successful” notification.

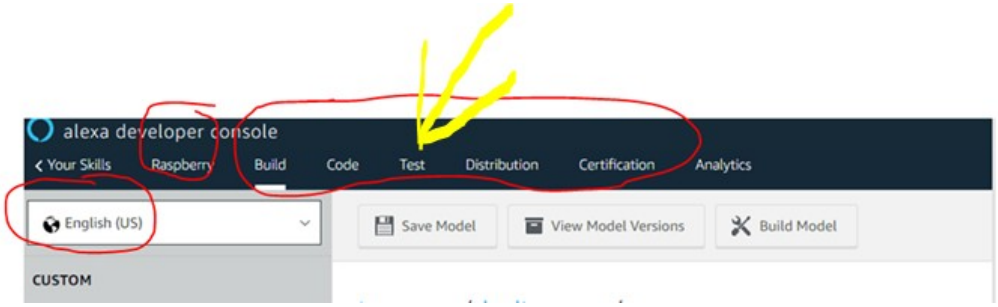
5. Colocar la addresa





1. Select Endpoint in the left pane and and check on “HTTPS”.
2. Enter the URL generated by serveo in Step 3.2 under Default Region.
3. Select “My development endpoint is sub-domain of domain that has a wildcard certificate from a certificate authority” for SSL certificate type.
4. Click on “Save Endpoints”

13. PROVANT amb text o veu en la consola



ENCENDRE

1. Select “Test” tab in the top menu bar.
2. Select “Development” in the pull down menu to enable Skill testing.
3. Say or input below to Alexa Simulator.
4 Alexa ask raspberry pi to turn the led on

APAGAR

1. Select “Test” tab in the top menu bar.
2. Select “Development” in the pull down menu to enable Skill testing.
3. Say or input below to Alexa Simulator.
4 Alexa ask raspberry pi to turn the led on