# **1ste code van raspberry pi**

import RPi.GPIO as GPIO

from gpiozero import Buzzer, Button, LED

import paho.mqtt.client as mqtt

import time

from time import sleep

import subprocess # Voor "servo 180" te versturen

buzzer = Buzzer(23) # (= GPIO 23, ...)

knop1 = Button(17,pull\_up=0)

toggle = True

led = LED(27)

mqtt\_broker = "192.168.0.242"

mqtt\_port = 1883

mqtt\_user = "EwoutDoms"

mqtt\_password = "SMOD3344"

mqtt\_topic\_prefix = "brievenbus/"

mqtt\_topic = mqtt\_topic\_prefix + "button\_press"

def on\_connect(client, userdata, flags, rc):

print("Geconnecteerd ")

client.subscribe([(mqtt\_topic\_prefix + "huis/open", 0),

(mqtt\_topic\_prefix + "huis/dicht", 0),])

def on\_message(client, userdata, msg):

topic = msg.topic

if topic.endswith("huis/open"):

buzzer.on()

print('Brievenbus geopend!!!')

elif topic.endswith("huis/dicht"):

buzzer.off()

print('Brievenbus gesloten')

def knop1\_callback():

global toggle

if knop1.is\_active == True and toggle == True:

print("knop is ingedrukt (Slot open)")

subprocess.run(["mosquitto\_pub", "-h", "localhost", "-t", "brievenbus/slot", "-m", "Servo 180", "-u", "EwoutDoms", "-P", "SMOD3344"])

print("Servo 180")

toggle = False

led.on()

sleep(1)

if knop1.is\_active == True and toggle == False:

print("knop is ingedrukt (Slot gesloten)")

subprocess.run(["mosquitto\_pub", "-h", "localhost", "-t", "brievenbus/slot", "-m", "Servo 0", "-u", "EwoutDoms", "-P", "SMOD3344"])

print("Servo 0")

toggle = True

led.off()

sleep(1)

client = mqtt.Client(mqtt.CallbackAPIVersion.VERSION1)

client.username\_pw\_set(mqtt\_user, mqtt\_password)

client.on\_connect = on\_connect

client.on\_message = on\_message

client.connect(mqtt\_broker, mqtt\_port, 60)

knop1.when\_pressed = knop1\_callback

client.loop\_forever()

# **2de code van raspberry pi**

import re

from typing import NamedTuple

import paho.mqtt.client as mqtt

from influxdb import InfluxDBClient

INFLUXDB\_ADDRESS = '192.168.0.242'

INFLUXDB\_USER = 'Ewout1'

INFLUXDB\_PASSWORD = 'SMOD3344a'

INFLUXDB\_DATABASE = 'EINDPRO'

MQTT\_ADDRESS = '192.168.0.242'

MQTT\_USER = 'EwoutDoms'

MQTT\_PASSWORD = 'SMOD3344'

MQTT\_TOPIC = 'brievenbus/+/+'

MQTT\_REGEX = 'brievenbus/([^/]+)/([^/]+)'

MQTT\_CLIENT\_ID = 'MQTTInfluxDBBridge'

influxdb\_client = InfluxDBClient(INFLUXDB\_ADDRESS, 8086, INFLUXDB\_USER,

INFLUXDB\_PASSWORD, None)

class SensorData(NamedTuple):

location: str

measurement: str

value: float

def on\_connect(client, userdata, flags, rc):

""" The callback for when the client receives a CONNACK response from the

server."""

print('Connected with result code ' + str(rc))

client.subscribe(MQTT\_TOPIC)

def \_parse\_mqtt\_message(topic, payload):

match = re.match(MQTT\_REGEX, topic)

if match:

location = match.group(1)

measurement = match.group(2)

if measurement == 'status':

return None

return SensorData(location, measurement, float(payload))

else:

return None

def \_send\_sensor\_data\_to\_influxdb(sensor\_data):

json\_body = [

{

'measurement': sensor\_data.measurement,

'tags': {

'location': sensor\_data.location

},

'fields': {

'value': sensor\_data.value

}

}

]

influxdb\_client.write\_points(json\_body)

def on\_message(client, userdata, msg):

"""The callback for when a PUBLISH message is received from the server."""

print(msg.topic + ' ' + str(msg.payload))

sensor\_data = \_parse\_mqtt\_message(msg.topic, msg.payload.decode('utf-8'))

if sensor\_data is not None:

\_send\_sensor\_data\_to\_influxdb(sensor\_data)

def \_init\_influxdb\_database():

databases = influxdb\_client.get\_list\_database()

if len(list(filter(lambda x: x['name'] == INFLUXDB\_DATABASE, databases))) == 0:

influxdb\_client.create\_database(INFLUXDB\_DATABASE)

influxdb\_client.switch\_database(INFLUXDB\_DATABASE)

def main():

\_init\_influxdb\_database()

mqtt\_client = mqtt.Client(mqtt.CallbackAPIVersion.VERSION1, MQTT\_CLIENT\_ID)

mqtt\_client.username\_pw\_set(MQTT\_USER, MQTT\_PASSWORD)

mqtt\_client.on\_connect = on\_connect

mqtt\_client.on\_message = on\_message

mqtt\_client.connect(MQTT\_ADDRESS, 1883)

mqtt\_client.loop\_forever()

if \_\_name\_\_ == '\_\_main\_\_':

print('MQTT to InfluxDB bridge')

main()