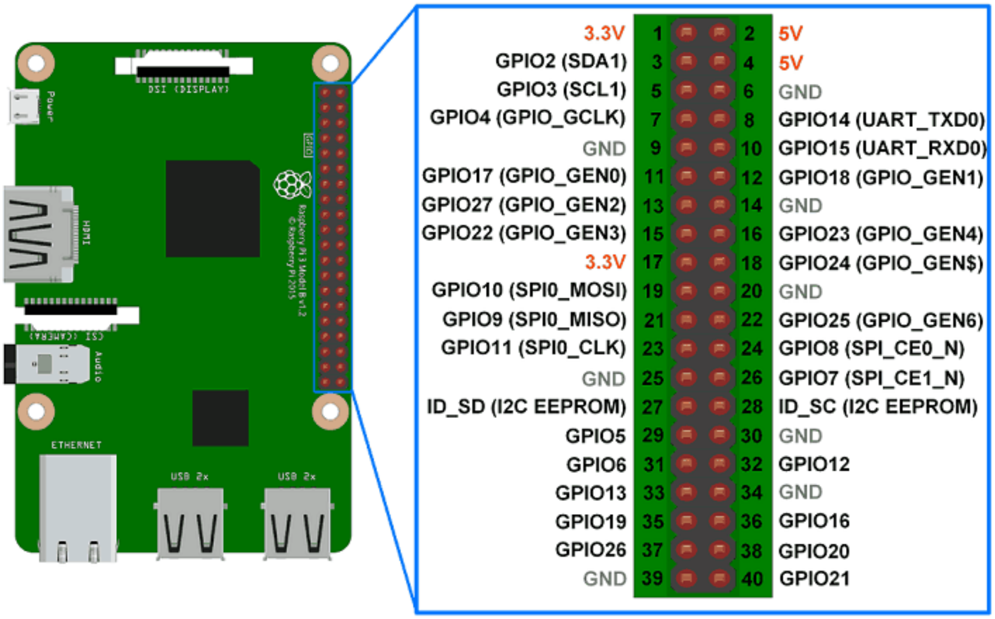
Pinout:



3.3V	1	2	5V
GPIO2 (SDA1)	3	4	5V
GPIO3 (SCL1)	5	6	GND
GPIO4 (GPIO_GCLK)	7	8	GPIO14 (UART_TXD0)
GND	9	10	GPIO15 (UART_RXD0)
GPIO17 (GPIO_GEN0)	11	12	GPIO18 (GPIO_GEN1)
GPIO27 (GPIO_GEN2)	13	14	GND
GPIO22 (GPIO_GEN3)	15	16	GPIO23 (GPIO_GEN4)
3.3V	17	18	GPIO24 (GPIO_GEN$)
GPIO10 (SPI0_MOSI)	19	20	GND
GPIO9 (SPI0_MISO)	21	22	GPIO25 (GPIO_GEN6)
GPIO11 (SPI0_CLK)	23	24	GPIO8 (SPI_CE0_N)
GND	25	26	GPIO7 (SPI_CE1_N)
ID_SD (I2C EEPROM)	27	28	ID_SC (I2C EEPROM)
GPIO5	29	30	GND
GPIO6	31	32	GPIO12
GPIO13	33	34	GND
GPIO19	35	36	GPIO16
GPIO26	37	38	GPIO20
GND	39	40	GPIO21

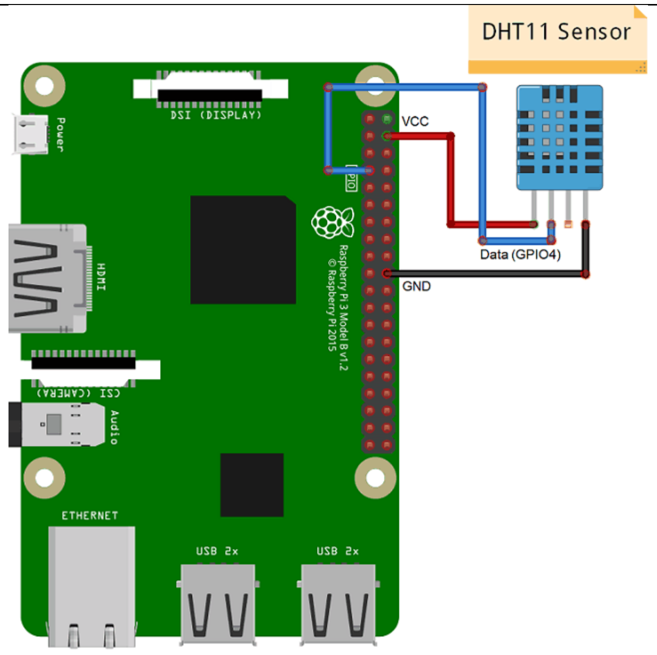
Exp. 2 – Interfacing sensors/actuators

DHT11 Sensor (Temperature + Humidity Sensor)

```
git clone https://github.com/adafruit/Adafruit_Python_DHT.git
```

Then go to folder Adafruit_Python_DHT in downloads, locate setup.py (ls command)

```
sudo python setup.py install
```



Open simpletest from from downloaded library in examples. sensor = Adafruit_DHT.DHT11 or 22

Exp. 2 –RFID Interfacing

- Radio Frequency Identification (RFID)
- RFID RC522
- Active/passive/semi-passive
- Tag/Reader
- LF/HF/UHF/Microwave/MM Wave
- Principle of transformer (resonant frequency and load modulation)
- Electromagnetic field
- Memory chip/power battery/clock-oscillator
- Coupling (Back scatter/Inductive/capacitive)

Connections:

• SDA connects to Pin 24. • SCK connects to Pin 23. • MOSI connects to Pin 19. • MISO connects to Pin 21. • GND connects to Pin 6. • RST connects to Pin 22. • 3.3v connects to Pin 1.

```
sudo raspi-config  ->  SPI Enable
```

If not enable by above method manual method:

```
sudo nano /boot/config.txt      ->      “dtparam=spi=on“
```

Programming:

```
sudo pip3 install spidev
sudo pip3 install mfrc522
```

Write:

```
#!/usr/bin/env python
import RPi.GPIO as GPIO
from mfrc522 import SimpleMFRC522
reader = SimpleMFRC522()
try:
    text = input('New data:')
    print("Now place your tag to write")
    reader.write(text)
    print("Written")
finally:
    GPIO.cleanup()
```

Read:

```
reader = SimpleMFRC522()
#!/usr/bin/env python
import RPi.GPIO as GPIO
from mfrc522 import SimpleMFRC522
reader = SimpleMFRC522()
try:
     id, text = reader.read()
     print(id)
     print(text)
finally:
        GPIO.cleanup()
```

Exp. 4 – BLE (Bluetooth Low Energy)

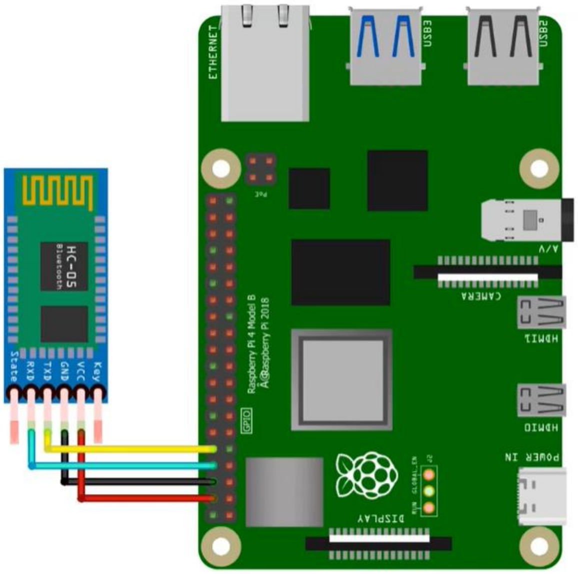
- applications where data needs to be transferred in small amounts at relatively low speed while consuming low amounts of power (e.g., heart rate monitor, step counter, wireless keyboard)

After connecting on-chip Bluetooth:

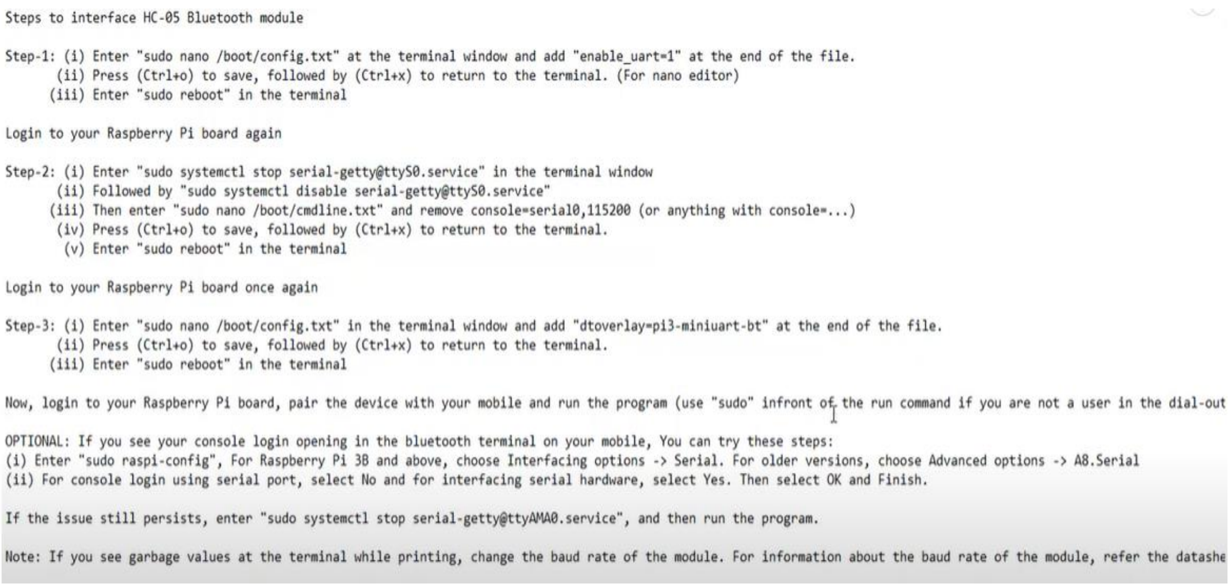
```
sudo apt-get install bluemanbluez Bluetooth
```

Command line + on-chip Bluetooth

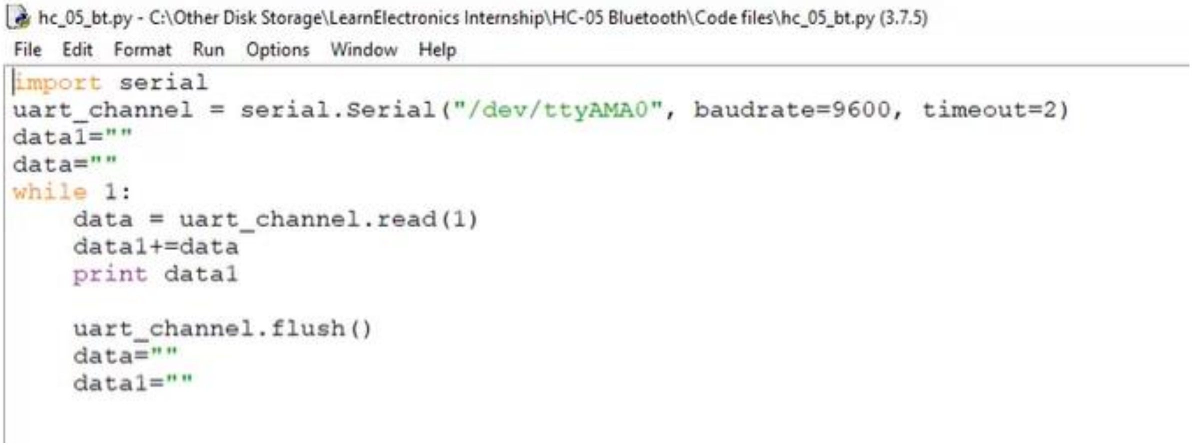
```
Bluetoothctl
power on
devices
scan on
agent on
pair <mac address>
trust <mac address>
Connect <mac address>
```



Android app: Bluetooth Terminal HC-05



```
Steps to interface HC-05 Bluetooth module

Step-1: (i) Enter "sudo nano /boot/config.txt" at the terminal window and add "enable_uart=1" at the end of the file.
       (ii) Press (Ctrl+o) to save, followed by (Ctrl+x) to return to the terminal. (For nano editor)
      (iii) Enter "sudo reboot" in the terminal

Login to your Raspberry Pi board again

Step-2: (i) Enter "sudo systemctl stop serial-getty@ttyS0.service" in the terminal window
       (ii) Followed by "sudo systemctl disable serial-getty@ttyS0.service"
      (iii) Then enter "sudo nano /boot/cmdline.txt" and remove console=serial0,115200 (or anything with console=...)
       (iv) Press (Ctrl+o) to save, followed by (Ctrl+x) to return to the terminal.
        (v) Enter "sudo reboot" in the terminal

Login to your Raspberry Pi board once again

Step-3: (i) Enter "sudo nano /boot/config.txt" in the terminal window and add "dtoverlay=pi3-miniuart-bt" at the end of the file.
       (ii) Press (Ctrl+o) to save, followed by (Ctrl+x) to return to the terminal.
      (iii) Enter "sudo reboot" in the terminal

Now, login to your Raspberry Pi board, pair the device with your mobile and run the program (use "sudo" infront of the run command if you are not a user in the dial-out

OPTIONAL: If you see your console login opening in the bluetooth terminal on your mobile, You can try these steps:
(i) Enter "sudo raspi-config", For Raspberry Pi 3B and above, choose Interfacing options -> Serial. For older versions, choose Advanced options -> A8.Serial
(ii) For console login using serial port, select No and for interfacing serial hardware, select Yes. Then select OK and Finish.

If the issue still persists, enter "sudo systemctl stop serial-getty@ttyAMA0.service", and then run the program.

Note: If you see garbage values at the terminal while printing, change the baud rate of the module. For information about the baud rate of the module, refer the datashe
```



hc_05_bt.py - C:\Other Disk Storage\LearnElectronics Internship\HC-05 Bluetooth\Code files\hc_05_bt.py (3.7.5)
File Edit Format Run Options Window Help

```python
import serial
uart_channel = serial.Serial("/dev/ttyAMA0", baudrate=9600, timeout=2)
data1=""
data=""
while 1:
        data = uart_channel.read(1)
        data1+=data
        print data1

        uart_channel.flush()
        data=""
        data1=""
```

MQTT (Message Queuing Telemetry Transport)

Installing Mosquitto

```
$sudo apt-get update
$sudo apt-get install mosquito
$sudo apt-get install mosquitto-clients
```

Subscribe to test1

```
mosquitto_sub -t "test"
Do this not above:
mosquitto_sub –h localhost –v –t test_channel
```

Publish test1

```
$mosquitto_pub -m "message from mosquitto_pub client" -t "test"
Do not use above use this:
mosquitto_pub –h localhost –t test_channel –m “MIT-WPU”
```

https://gist.github.com/bradmontgomery/8f1de0e56fa86c29a7daadab1c370c56

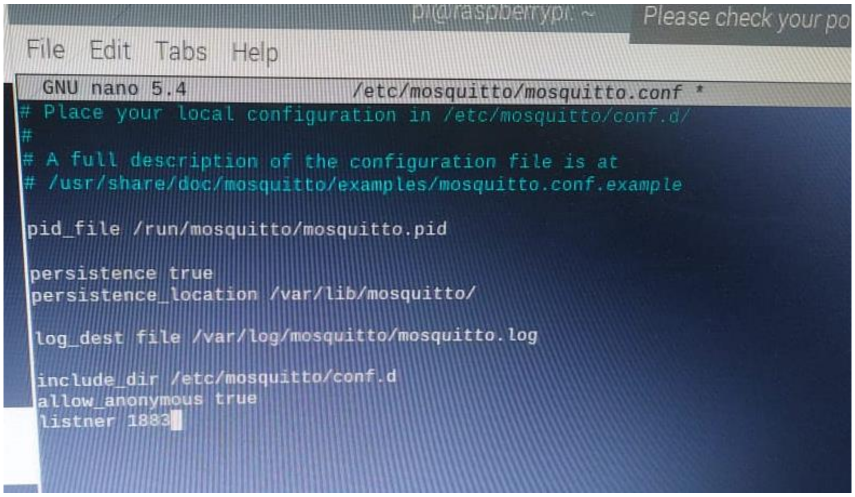
Using IP Address:

```
sudo nano /etc/mosquitto/ mosquito.conf
    ➔ Make changes
Sudo systemctl restart mosquito
    ➔ To check ip address:
hostname –I
netstat –t ln
mosquito_sub –h ipaddress –v –t Test_channel
    ➔ For publisher
mosquitto_pub –h ipaddress  –t test_channel –m “MIT-WPU”
```

Uninstall MQTT: (NO NEED)

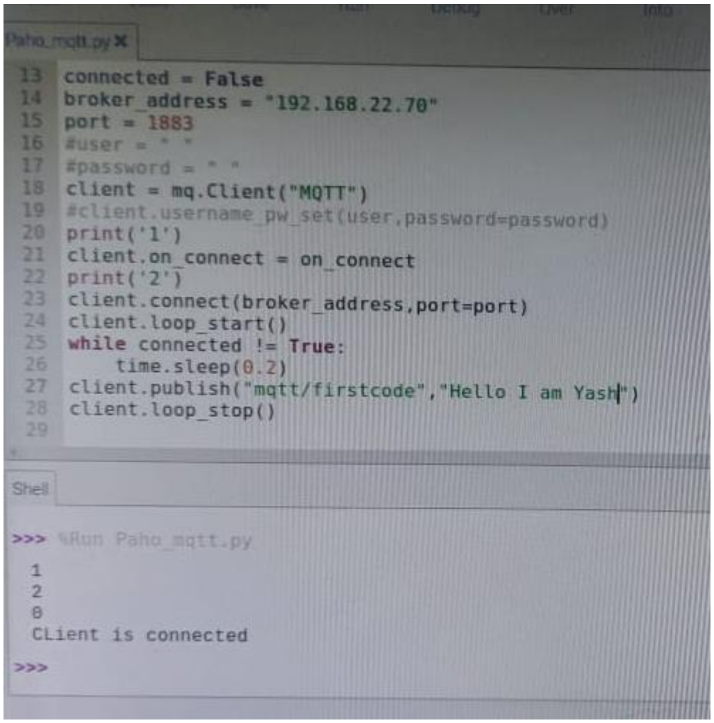
```
sudo apt-get --purge remove mosquito
```

MQTT .CONFIG



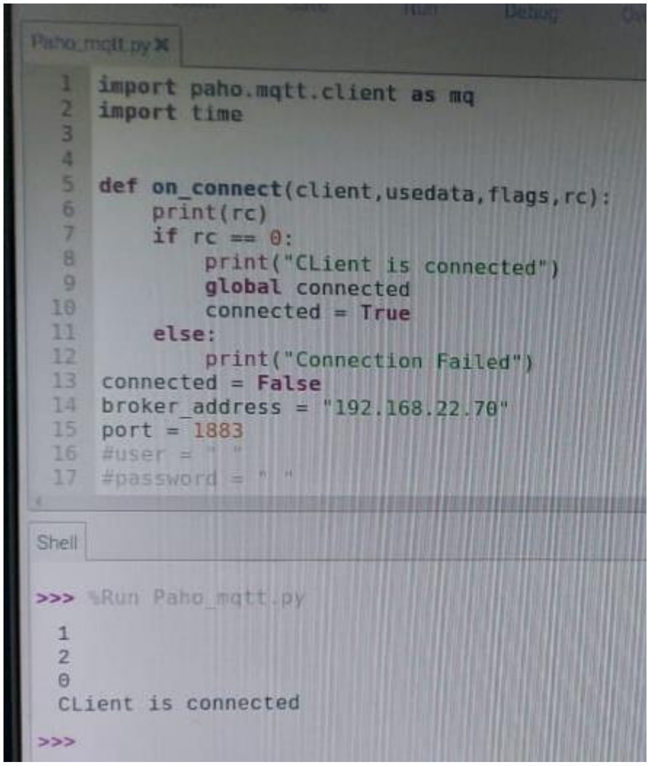
```
  GNU nano 5.4                    /etc/mosquitto/mosquitto.conf *
# Place your local configuration in /etc/mosquitto/conf.d/
#
# A full description of the configuration file is at
# /usr/share/doc/mosquitto/examples/mosquitto.conf.example

pid_file /run/mosquitto/mosquitto.pid

persistence true
persistence_location /var/lib/mosquitto/

log_dest file /var/log/mosquitto/mosquitto.log

include_dir /etc/mosquitto/conf.d
allow_anonymous true
listner 1883
```

Python script using paho MQTT:



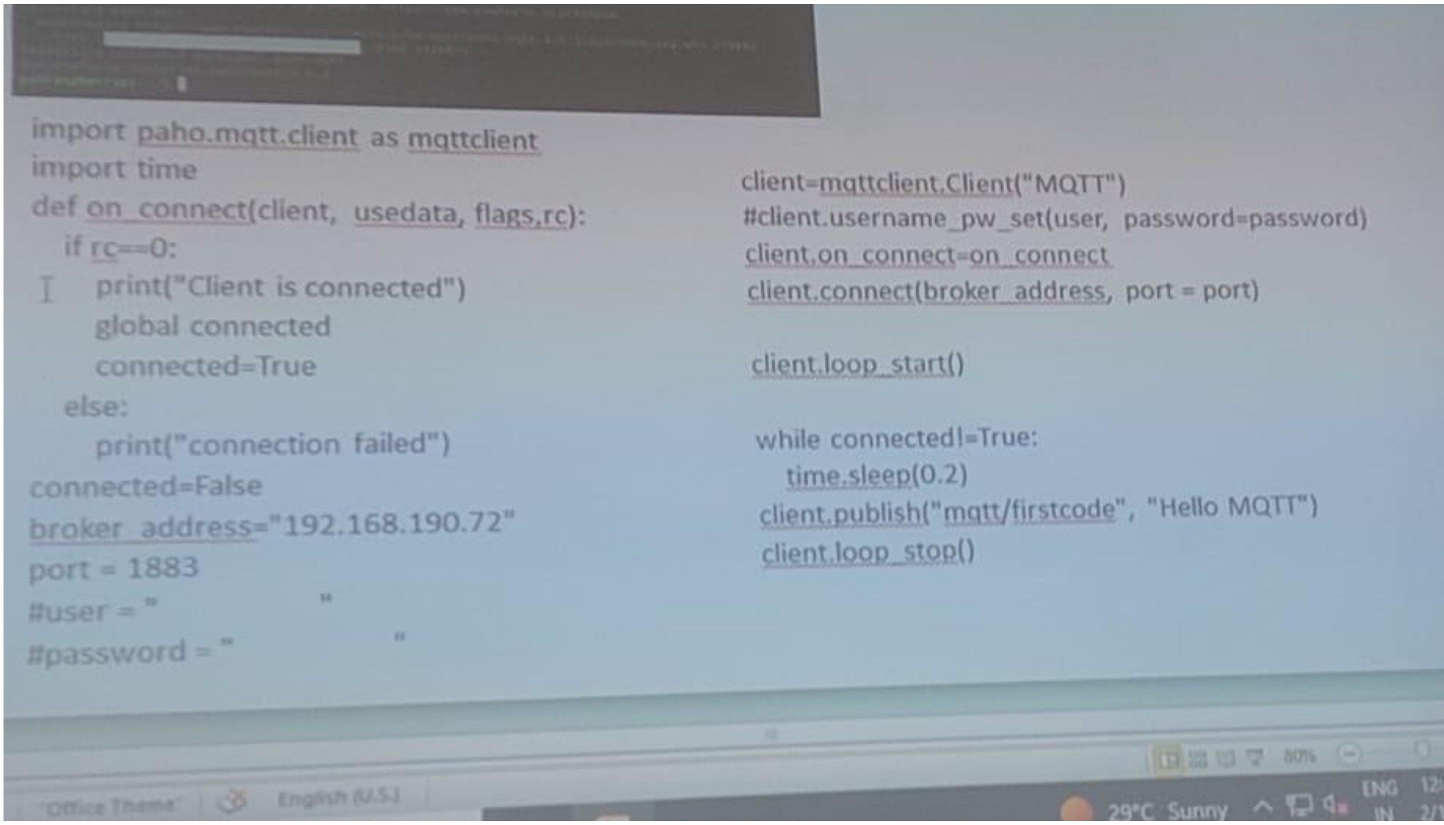
```python
connected = False
broker_address = "192.168.22.70"
port = 1883
#user = " "
#password = " "
client = mq.Client("MQTT")
#client.username_pw_set(user,password=password)
print('1')
client.on_connect = on_connect
print('2')
client.connect(broker_address,port=port)
client.loop_start()
while connected != True:
    time.sleep(0.2)
client.publish("mqtt/firstcode","Hello I am Yash")
client.loop_stop()
```

```
>>> %Run Paho_mqtt.py
 1
 2
 0
 CLient is connected
>>>
```



```python
import paho.mqtt.client as mq
import time


def on_connect(client,usedata,flags,rc):
    print(rc)
    if rc == 0:
        print("CLient is connected")
        global connected
        connected = True
    else:
        print("Connection Failed")
connected = False
broker_address = "192.168.22.70"
port = 1883
#user = " "
#password = " "
```

```
>>> %Run Paho_mqtt.py
 1
 2
 0
 CLient is connected
>>>
```



```
import paho.mqtt.client as mqttclient
import time
def on_connect(client, usedata, flags,rc):
  if rc==0:
    print("Client is connected")
    global connected
    connected=True
  else:
    print("connection failed")
connected=False
broker_address="192.168.190.72"
port = 1883
#user = "          "
#password = "          "
```

```
client=mqttclient.Client("MQTT")
#client.username_pw_set(user, password=password)
client.on_connect=on_connect
client.connect(broker_address, port = port)

client.loop_start()

while connected!=True:
  time.sleep(0.2)
client.publish("mqtt/firstcode", "Hello MQTT")
client.loop_stop()
```