Led (Subscriber)

```
import paho.mqtt.client as PahoMQTT
import time
import json
class Led:
    def __init__(self, clientID,topic,broker):
        self.clientID = clientID
        # create an instance of paho.mqtt.client
        self._paho_mqtt = PahoMQTT.Client(clientID, False)
        # register the callback
        self._paho_mqtt.on_connect = self.myOnConnect
        self._paho_mqtt.on_message = self.myOnMessageReceived
        self.topic = topic
        self.messageBroker =broker
        self.status=""

    def start (self):
        #manage connection to broker
        self._paho_mqtt.connect(self.messageBroker, 1883)
        self._paho_mqtt.loop_start()
        # subscribe for a topic
        self._paho_mqtt.subscribe(self.topic, 2)

    def stop (self):
        self._paho_mqtt.unsubscribe(self.topic)
        self._paho_mqtt.loop_stop()
        self._paho_mqtt.disconnect()

    def myOnConnect (self, paho_mqtt, userdata, flags, rc):
        print ("Connected to %s with result code: %d" % (self.messageBroker, rc))

    def myOnMessageReceived (self, paho_mqtt , userdata, msg):
        d=json.loads(msg.payload)
        self.status=d['value']
        client=d['client']
        timestamp=d['timestamp']
        print(f'The led has been set to {self.status} at time {timestamp} by the client {cli


if __name__ == "__main__":
    broker=json.load(open("settings1.json"))['broker']
    test = Led("MyLed","ledCommand",broker)
    test.start()
    while True:
```

```
        time.sleep(1)
    test.stop()
```

Client (Publisher)

```
import paho.mqtt.client as PahoMQTT
import time
import json
class LedCommander:
    def __init__(self, clientID, topic,broker):
        self.clientID = clientID
        self.topic=topic
        # create an instance of paho.mqtt.client
        self._paho_mqtt = PahoMQTT.Client(self.clientID, False)
        # register the callback
        self._paho_mqtt.on_connect = self.myOnConnect
        self.message={'client': clientID,'n':'switch','value':'', 'timestamp':''}
        #self.messageBroker = 'iot.eclipse.org'
        self.messageBroker =broker

    def start (self):
        #manage connection to broker
        self._paho_mqtt.connect(self.messageBroker, 1883)
        self._paho_mqtt.loop_start()

    def stop (self):
        self._paho_mqtt.loop_stop()
        self._paho_mqtt.disconnect()

    def myPublish(self,value):
        self.message['timestamp']=str(time.time())
        self.message['value']=value
        self._paho_mqtt.publish(self.topic, json.dumps(self.message), 2)

    def myOnConnect (self, paho_mqtt, userdata, flags, rc):
        print ("Connected to %s with result code: %d" % (self.messageBroker, rc))



if __name__ == "__main__":
    broker=json.load(open("settings1.json"))['broker']
    led_client = LedCommander("LedCommander",'ledCommand',broker)
    led_client.start()
    print('Welcome to the client to switch on/off the lamp\n')
    done=False
```

```
command_list='Type:\n"on" to set the light on\n"off" to set it off\n"q" to quit\n'
while not done:
    print(command_list)
    user_input=input()
    if user_input=="on":
        led_client.myPublish(user_input)
    elif user_input=="off":
        led_client.myPublish(user_input)
    elif user_input=='q':
        done=True
    else:
        print('Unknown command')
led_client.stop()
```