

IoT con MQTT (<https://mqtt.org/>)

Lo scopo di questa è utilizzare il protocollo MQTT per lo scambio di messaggi fra dispositivi IoT.

L'attività si sviluppa nelle seguenti fasi:

1. Studio delle funzionalità del protocollo MQTT
2. Utilizzo di un broker MQTT gratuito di test su Cloud e di un client MQTT standard, per esplorare le funzionalità
3. Scrittura di due client (publisher e subscriber) Python sotto Windows, che interagiscono col broker MQTT gratuito di test su Cloud
4. Installazione di un broker MQTT su Windows
5. Installazione di un broker MQTT su macchina virtuale Debian sotto Virtualbox
6. Installazione del broker MQTT su Raspberry
7. Trasferimento dei client su Raspberry ed integrazione col sensore/attuatore
8. Uso di una dashboard MQTT per il controllo del sensore/attuatore

1. Studio delle funzionalità del protocollo MQTT

- a. Introduzione a MQTT ([Video 4:04](#))
- b. Introduzione a MQTT ([testo](#))

2. Utilizzo di un broker MQTT gratuito di test su Cloud e di un client MQTT standard, per esplorare le funzionalità

- a. Installare un client standard per Windows, ad esempio [MQTTX](#) (non serve essere amministratori)
- b. Lavorare con un compagno:
 - Lanciare MQTTX su entrambe le macchine e connettersi al broker gratuito di test test.mosquitto.org
 - Da una delle due macchine effettuare la **subscription** ad un topic con nome a scelta
 - Dall'altra macchina effettuare un invio (Plaintext) sullo stesso topic: verificare che chi ha effettuato la subscription riceva il messaggio
 - Effettuare la subscription da una terza macchina e riprovare ad inviare un messaggio

3. Scrittura di due client (publisher e subscriber) Python sotto Windows, che interagiscono col broker MQTT gratuito di test su Cloud

- a. Installare la libreria Pythom **paho-mqtt**
- b. Fare riferimento a questo [tutorial](#)
- c. Provare il seguente programma di esempio che pubblica su di un topic e verificare, tramite MQTTX, la corretta pubblicazione

```

1  import paho.mqtt.publish as publish
2  import time
3  TOPIC="prova1"
4  BROKER="test.mosquitto.org"
5
6  # pubblica ad intervalli di 5 secondi un numero progressivo
7  i=0
8  while True:
9      i=i+1
10     msg=str(i)
11     print(i)
12
13     publish.single(TOPIC, msg, hostname=BROKER)
14
15     time.sleep(5)

```

- d. Provare il seguente programma che sottoscrive un topic e, alla ricezione del messaggio lo stampa; inviare da MQTTX qualche messaggio sul topic

```

1  import paho.mqtt.client as client
2  TOPIC="prova1"
3  #BROKER="test.mosquitto.org"
4  BROKER="192.168.43.47"
5
6  # funzione di callback dopo la corretta connessione
7  def on_connect(subscriber, userdata, flags, rc):
8      print("Connesso con return code "+str(rc))
9
10     # sottoscrizione dopo che la connessione è avvenuta
11     subscriber.subscribe(TOPIC)
12
13     # funzione di callback alla ricezione di messaggi
14     def on_message(subscriber, userdata, msg):
15         print(msg.topic+" "+str(msg.payload))
16
17     subscriber = client.Client()
18     subscriber.on_connect = on_connect
19     subscriber.on_message = on_message
20
21     subscriber.connect(BROKER, 1883, 60)
22
23     # ciclo di attesa messaggi
24     subscriber.loop_forever()

```

- e. Lanciare entrambi i programmi in modo che il publisher invii e il subscriber riceva

4. Installazione di un broker MQTT su Windows

- a. *Installare (password di amministratore) il broker MQTT **mosquitto** da [questo link](#)*
- b. *Copiare in una propria cartella il file di configurazione:*

```
C:\"Program Files"\mosquitto\mosquitto.conf
```

ed impostare le seguenti due opzioni:

```
allow_anonymous true  
listener 1883
```

- c. *Lanciare da shell mosquitto in modalità verbose*

```
C:\"Program Files"\mosquitto\mosquitto -v -c mosquitto.conf
```

- d. *Provare i due programmi publisher e subscriber della fase 3, osservando i messaggi di log prodotti da mosquitto nella finestra della shell*

5. Installazione di un broker MQTT su macchina virtuale Debian sotto Virtualbox (opzionale)

- a. *Creare una macchina virtuale Linux ed installare [questa immagine Debian](#)*
- b. *In fase di installazione, impostare il proxy (se lavorate a scuola) e specificare **"root"** come password per root e specificare come utente da creare **"mosquitto"** con password **"mosquitto"** e ancora **"mosquitto"** come nome macchina.*
- c. *Loggarsi come "root"*
*Per impostare o eliminare il proxy è necessario creare il file **apt.conf**, usando l'editor **nano**, nella cartella **/etc/apt/**; il file deve contenere la seguente riga (se è necessario disattivare il proxy, anteporre un #):*

```
Acquire::http::Proxy "http://proxy:3128";
```

- d. *Installare i tools di gestione rete*

```
apt install net-tools
```

- e. *Installare il broker MQTT **mosquitto***

```
apt install mosquitto
```

- f. *Mosquitto viene installato per default come servizio: risulta però più comodo, per scopi di debugging, lanciarlo da shell. Occorre quindi fermare il servizio e disabilitarlo. Controllare con **netstat** che effettivamente il servizio sia fermato*

```
netstat -lp|grep 1883  
systemctl stop mosquitto  
netstat -lp|grep 1883
```

```
systemctl disable mosquitto
```

- g. Creare il file di configurazione **mosquitto.conf** ad esempio nella home di root; il file contiene le seguenti righe per permettere l'accesso senza autenticazione e l'ascolto anche sulla rete esterna e non solo sulla macchina locale

```
allow_anonymous true  
listener 1883
```

- h. Lanciare mosquitto in modalità verbose

```
mosquitto -c mosquitto.conf -v
```

- i. Provare i due programmi publisher e subscriber della fase 3, osservando i messaggi di log prodotti da mosquitto nella finestra della shell

6. Installazione del broker MQTT su Raspberry (opzionale)

Procedere all'installazione in modo analogo a quanto fatto per Debian, partendo dal punto 5.e. Accedere però come utente normale ed utilizzare **sudo** per le operazioni di installazione e configurazione

7. Trasferimento dei client su Raspberry ed integrazione col sensore/attuatore

Trasferire il subscriber e il publisher su Raspberry ed integrarli in modo tale che:

- Il publisher riceva il pacchetto dal sensore e li pubblichi sulla topic **tps/sensoreLuce**
- Il subscriber riceva messaggi di due topic: **tps/motoreVelocita** e **tps/motoreDirezione** e prepari e invii il pacchetto per l'attuatore
- Inviare e ricevere i messaggi tramite MQTTX: usare un formato JSON

8. Uso di una dashboard MQTT per il controllo del sensore/attuatore

Scopo di questa fase è utilizzare un client MQTT grafico (dashboard), ad esempio **IoTMQTTPanel** per Android.

Configurarla in modo che riceva i dati del publisher e li mostri, ad esempio, con uno **Vertical Meter** e che invii dati al subscriber usando, ad esempio, **Radio Buttons** per la direzione e uno **Slider** per la velocità.