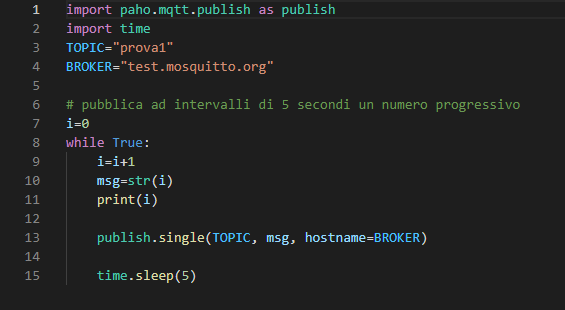
**IoT con MQTT (**[**https://mqtt.org/**](https://mqtt.org/)**)**

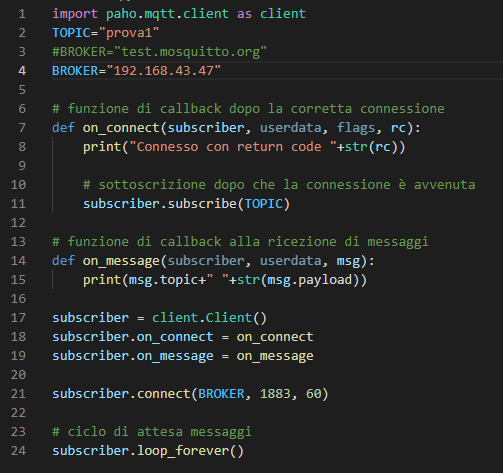
*Lo scopo di questa è utilizzare il protocollo MQTT per lo scambio di messaggi fra dispositivi IoT.*

*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*
9. **Studio delle funzionalità del protocollo MQTT**
   1. *Introduzione a MQTT (*[*Video 4:04*](https://www.youtube.com/watch?v=EIxdz-2rhLs)*)*
   2. *Introduzione a MQTT (*[*testo*](https://advantech-bb.com/basics-mqtt/)*)*
10. **Utilizzo di un broker MQTT gratuito di test su Cloud e di un client MQTT standard, per esplorare le funzionalità**
    1. *Installare un client standard per Windows, ad esempio* [*MQTTX*](https://mqttx.app/) *(non serve essere amministratori)*
    2. *Lavorare con un compagno:*
       * *Lanciare MQTTX su entrambe le macchine e connettersi al broker gratuito di test* [*test.mosquitto.org*](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*
11. **Scrittura di due client (publisher e subscriber) Python sotto Windows, che interagiscono col broker MQTT gratuito di test su Cloud**
    1. *Installare la libreria Pythom* ***paho-mqtt***
    2. *Fare riferimento a questo* [*tutorial*](https://antima.it/tutorial-utilizzare-mqtt-con-python-la-classe-client-parte-1/)
    3. *Provare il seguente programma di esempio che pubblica su di un topic e verificare, tramite MQTTX, la corretta pubblicazione*



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



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

1. **Installazione di un broker MQTT su Windows**
   1. *Installare (password di amministratore) il broker MQTT* ***mosquitto*** *da* [*questo link*](https://mosquitto.org/download/)
   2. *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

* 1. *Lanciare da shell mosquitto in modalità verbose*

*C:\****”****Program Files****”****\mosquitto\*mosquitto –v –c mosquitto.conf

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

1. **Installazione di un broker MQTT su macchina virtuale Debian sotto Virtualbox (opzionale)**
   1. *Creare una macchina virtuale Linux ed installare* [*questa immagine Debian*](https://cdimage.debian.org/debian-cd/current/amd64/iso-cd/debian-11.2.0-amd64-netinst.iso)
   2. *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.*
   3. *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";

* 1. *Installare i tools di gestione rete*

apt install net-tools

* 1. *Installare il broker MQTT* ***mosquitto***

apt install mosquitto

* 1. *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

* 1. *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

* 1. *Lanciare mosquitto in modalità verbose*

mosquitto –c mosquitto.conf -v

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

1. **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*

1. **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*

1. **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****.***

*Configurarlo 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.*