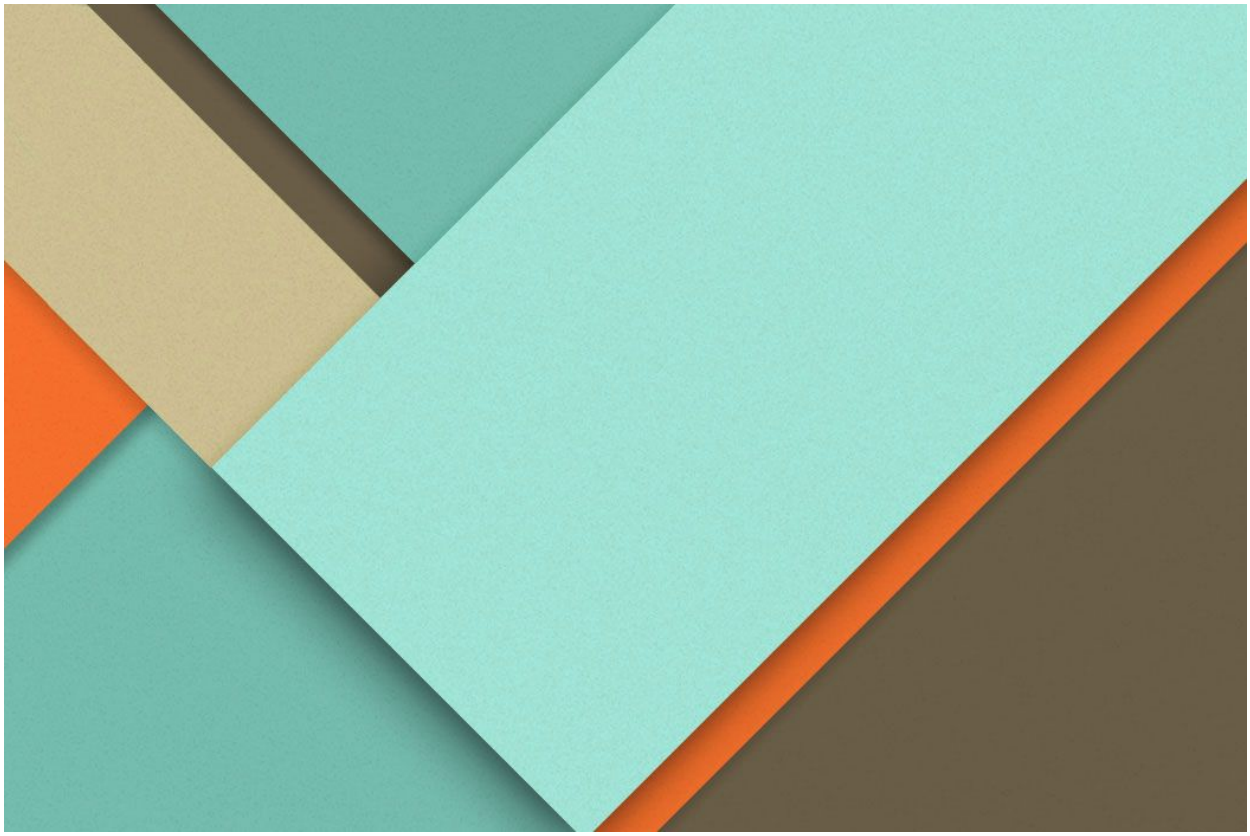


Conexão MQTT

01/12/2019

Mateus de Moura Ramos Bittencourt
Universidade de Brasília
Matrícula 160079284

MQTT

O MQTT (Message Queue Telemetry Transport) é um protocolo para comunicação assíncrona, isso é, emissor e transmissor não estão ligados de alguma forma. Ele implementa um sistema de publicação e assinatura, que pode ser entendida como um publicante escreve mensagens para aqueles que estão assinados lerem.

Algumas informações importantes sobre esse protocolo são que: Existe um mediador entre os clientes, o broker, que trata as comunicações e encaminha a todos os assinantes as mensagens publicadas sobre aqueles tópicos específicos. Um cliente conectado pode desempenhar ambos os papéis, tanto de publicante quanto de assinante. Como não há conexão entre os dois, não há a necessidade de se conhecerem antes de estabelecerem uma comunicação.

Esse protocolo é muito utilizado para comunicações IoT, tornando-se muito importante para a maioria das aplicações atuais.

MQTT versus HTTP

Recentemente o protocolo MQTT cresceu bastante a ponto de ser repor o HTTP em várias aplicações. Isto deve-se a vários fatores, como o fato de o MQTT ser do tipo “publisher/subscriber”, mais leve e de melhor aplicação em dispositivos mobile. Sua velocidade supera em larga escala a do protocolo HTTP, pois ele é leve devido à sua transferência de dados no formato de bytes.

Vale ressaltar ainda que o uso do MQTT é de mais fácil aprendizagem, sendo ele mais simples que o HTTP. O protocolo MQTT é fácil de usar. É essencial quando o tempo de resposta e largura de banda estão em consideração. Também é ótimo em caso de conectividade intermitente.

HTTP é um protocolo bom e muito utilizado, mas o MQTT é mais adequado quando se refere ao desenvolvimento da IoT.

Objetivo

O objetivo desse projeto é implementar um protocolo MQTT, onde a leitura de um sensor é publicada e existem clientes assinantes de tópicos diferentes lendo essas leituras. Para tal, um arquivo de texto contém informações de leitura da umidade e da temperatura de um local.

Implementação

Neste trabalho foi utilizado a biblioteca mosquitto, que facilita muito a implementação da comunicação entre o broker e os clientes. Essa biblioteca é utilizada em python, e para seu download e utilização basta seguir as instruções:

Para baixar a biblioteca em um sistema Linux, digite no terminal:

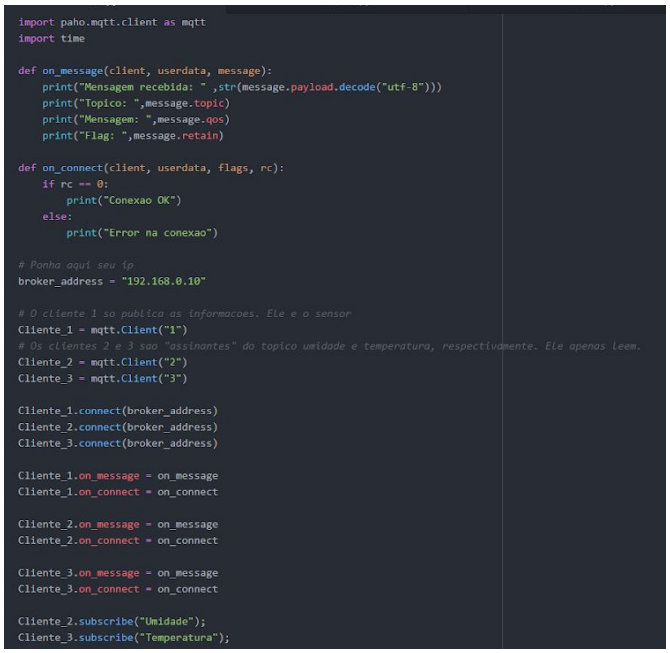
```
pip install paho-mqtt
```

E para sua utilização no código, basta importar a biblioteca conforme mostrado a seguir:

```
import paho.mqtt.client
```

A partir desse passo, sua biblioteca já está pronta para uso.

A seguir, vemos uma imagem referente ao código e uma breve explicação de algumas funções utilizadas.



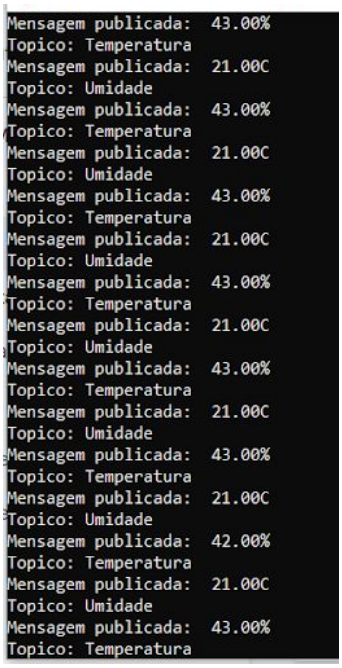
```python
import paho.mqtt.client as mqtt
import time

def on_message(client, userdata, message):
    print("Mensagem recebida: " ,str(message.payload.decode("utf-8")))
    print("Topico: ",message.topic)
    print("Mensagem: ",message.qos)
    print("Flag: ",message.retain)

def on_connect(client, userdata, flags, rc):
    if rc == 0:
        print("Conexao OK")
    else:
        print("Error na conexao")

# Ponha aqui seu ip
broker_address = "192.168.0.10"

# O cliente 1 so publica as informacoes. Ele e o sensor
Cliente_1 = mqtt.Client("1")
# Os clientes 2 e 3 sao "assinantes" do topico umidade e temperatura, respectivamente. Ele apenas leem.
Cliente_2 = mqtt.Client("2")
Cliente_3 = mqtt.Client("3")

Cliente_1.connect(broker_address)
Cliente_2.connect(broker_address)
Cliente_3.connect(broker_address)

Cliente_1.on_message = on_message
Cliente_1.on_connect = on_connect

Cliente_2.on_message = on_message
Cliente_2.on_connect = on_connect

Cliente_3.on_message = on_message
Cliente_3.on_connect = on_connect

Cliente_2.subscribe("Umidade");
Cliente_3.subscribe("Temperatura");
```

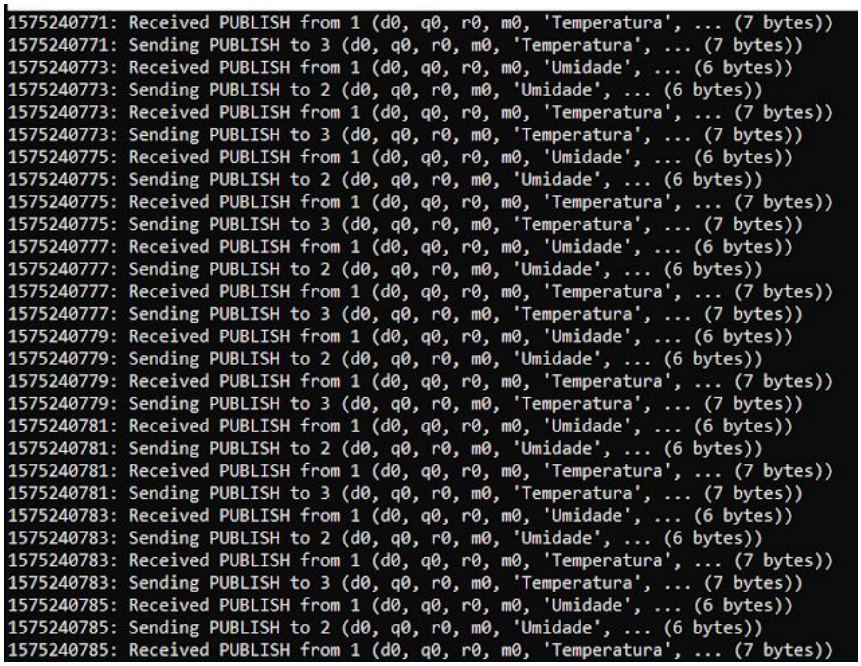
As duas funções inicializadas no início do código dizem respeito à mensagem recebida pelo assinante, e à conexão de um novo cliente, respectivamente.

A inicialização de um cliente é através de cliente = mqtt.client("Nome Qualquer"). Para se inscrever num tópico basta usar a função subscribe seguida do tópico. E para publicação basta usar a função publish, seguida do tópico e da mensagem a ser publicada.

Desse modo, o código acima inicializa 3 clientes (1, 2 e 3) e conectam os 3 no mesmo broker. Dois deles (2 e 3) se inscrevem em tópicos diferentes, e o último (1) publica as informações lidas do sensor, sendo metade delas com tópico temperatura e metade umidade. A leitura desse sensor é feita através da leitura do arquivo texto contendo todas as leituras.

A seguir estão os resultados encontrados:





Para compilar e executar o código basta escrever

python 160079284.py

no terminal linux. Vale lembrar que o arquivo texto utilizado estava nomeado como teste.txt, portanto para usar outro arquivo basta renomear esse campo no lugar especificado no código.

Conclusão

O protocolo pôde ser visto na prática de maneira simplória, através de 3 clientes. Sua implementação prática é mais complexa, pois possui deciframento de mensagens e outras especialidades, que o permitem ser amplamente utilizado para comunicação remota.

Nota-se também que no código houveram vários lugares onde o delay foi implementado apenas para melhor visualização, e outros lugares pois é necessário um certo tempo de leitura para obter as mensagens escritas no callback.