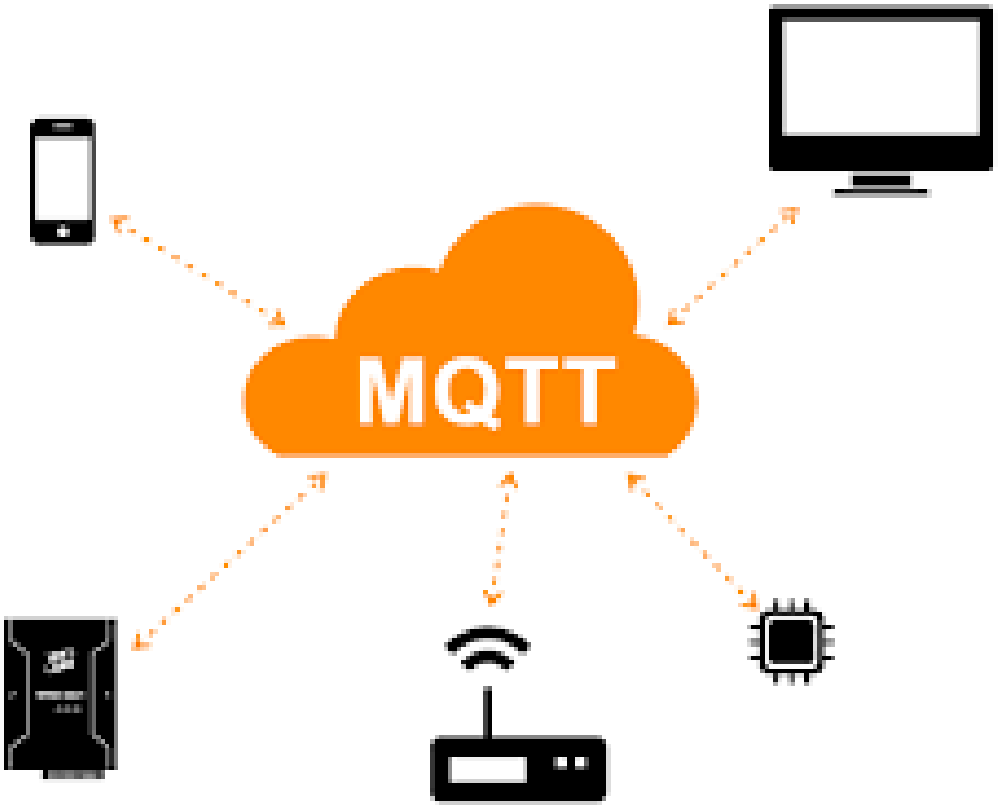
Conceitos e Tecnologias para Dispositivos Conectados (C115)

Prof. Samuel Baraldi Mafra

Inatel
Instituto Nacional de Telecomunicações

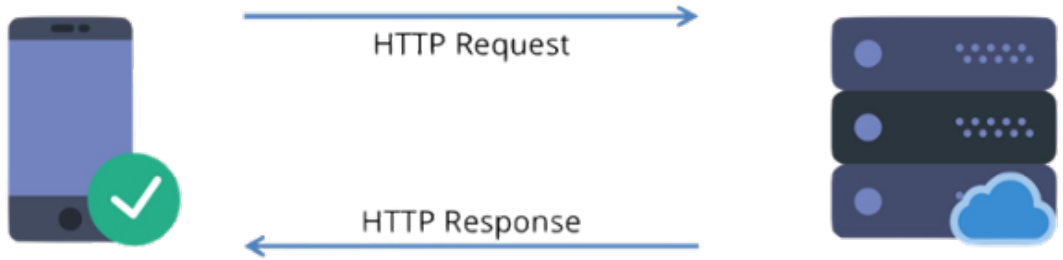
Protocolo MQTT:



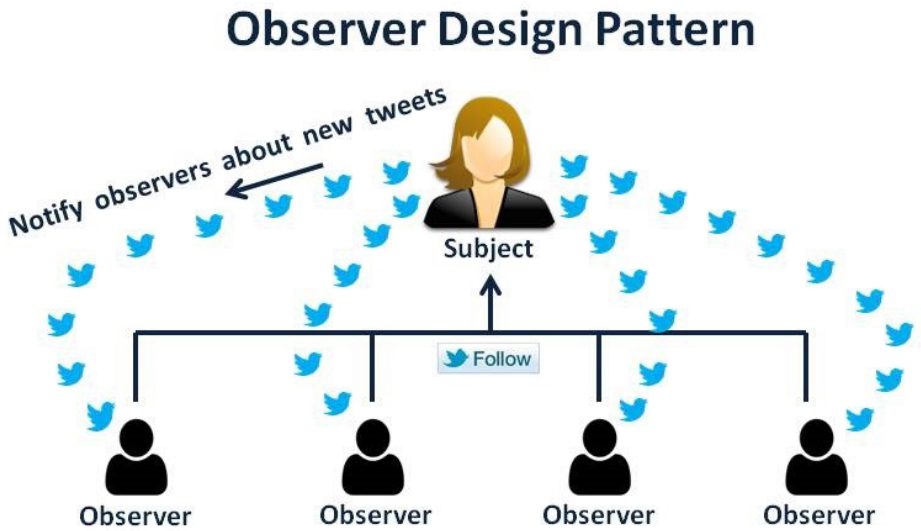
- MQTT(Message Queuing Telemetry Transport) é um protocolo de comunicação máquina para máquina (M2M - Machine to Machine) com foco em Internet of Things (IoT) que funciona em cima do protocolo TCP/IP.
- Um sistema MQTT se baseia na comunicação entre cliente e servidor, em que o primeiro pode realizar tanto postagens quanto captação de informação (cliente) e o segundo administra os dados a serem recebidos e enviados (broker).

- Nos anos 90 a IBM criou o protocolo MQTT.
- Sua origem se deu à necessidade de um protocolo simples e leve que conseguisse comunicar várias máquinas entre si;
- Uma comunicação que ocorreria utilizando microcontroladores para a obtenção de dados que tivesse uma taxa de transmissão leve para a comunicação entre as máquinas e os sensores.

- No padrão Request-Response há dois atores: Cliente e Servidor. Nesse protocolo o servidor fica o tempo todo ouvindo requisições que podem chegar dos seus cliente, porém ele estabelece uma conexão apenas temporária com o servidor, a medida que for realizando as requisições e obtendo as respostas, após isso a conexão é quebrada e o cliente não será notificado de nenhuma modificação.



- No padrão Observer possuímos dois atores principais: os observadores (observer) e o sujeito (subject).
- Os observadores irão realizar uma requisição para se inscrever no subject e dessa forma ser notificado quando houver alguma mudança de estado, o sujeito irá possuir uma lista dos seus observadores para que ele saiba para quem enviar as notificações quando houver a mudança de estado.
- Caso não seja mais interessante ao sujeito enviar informações para um observador específico ou o observador não se interessa mais pelo estado do sujeito, a inscrição pode ser desfeita.



- O Padrão Publish-Subscribe é muito parecido com o padrão Observe, porém nesse caso adicionamos o papel do Broker, que é responsável por filtrar as mensagens e saber exatamente para quem enviar.
- Dessa forma, o publisher e subscriber não precisam se conhecer diretamente e apenas precisam conhecer o Broker, que é quem fará a notificação da mudança de estados e enviará essa informação para aqueles que tiverem inscritos no tópico referenciado.
- Por fim, o publish precisa se preocupar apenas com enviar as informações e estabelecer a conexão exclusivamente com o Broker.

- É comum em alguns casos os sistemas terem atuações tanto de Publisher quanto de Subscriber.
- Considerando um caso de sistema que possui um sensor de temperatura e está ligado a um ar-condicionado e um sistema que acompanha a temperatura e envia o sinal para ligar o ar, os dois terão as duas situações;
- O sistema 1 precisa publicar as mudanças de temperatura e receber o sinal para ligar o ar;
- Enquanto o sistema 2 precisa receber as informações da temperatura e enviar o sinal para ligar o ar.

Tópicos

- No MQTT, a palavra tópico se refere a uma sequência UTF-8 que o broker usa para filtrar mensagens para cada cliente conectado. O tópico consiste em um ou mais níveis de tópico. Cada nível de tópico é separado por uma barra (separador de nível de tópico).
- Exemplo: SRS/Inatel/temperatura
  SRS/Inatel/umidade
  SRS/Inatel/temperatura/predio1/sala20

Curingas

- Quando um cliente se inscreve em um tópico, ele pode se inscrever no tópico exato de uma mensagem publicada ou pode usar curingas para se inscrever em vários tópicos simultaneamente. Um curinga só pode ser usado para assinar tópicos, não para publicar uma mensagem. Existem dois tipos diferentes de curingas: nível único e vários níveis .
  - Nível único: + Como o nome sugere, um curinga de nível único substitui um nível de tópico. O símbolo de mais representa um curinga de nível único em um tópico
  - Exemplo: SRS/+/temperatura
  - Multi nível: # O curinga multinível cobre muitos níveis de tópico. O símbolo de hash representa o curinga de vários níveis no tópico. Para que o broker determine quais tópicos correspondem, o curinga de vários níveis deve ser colocado como o último caractere no tópico e precedido por uma barra.
  - Exemplo: SRS/Inatel/#

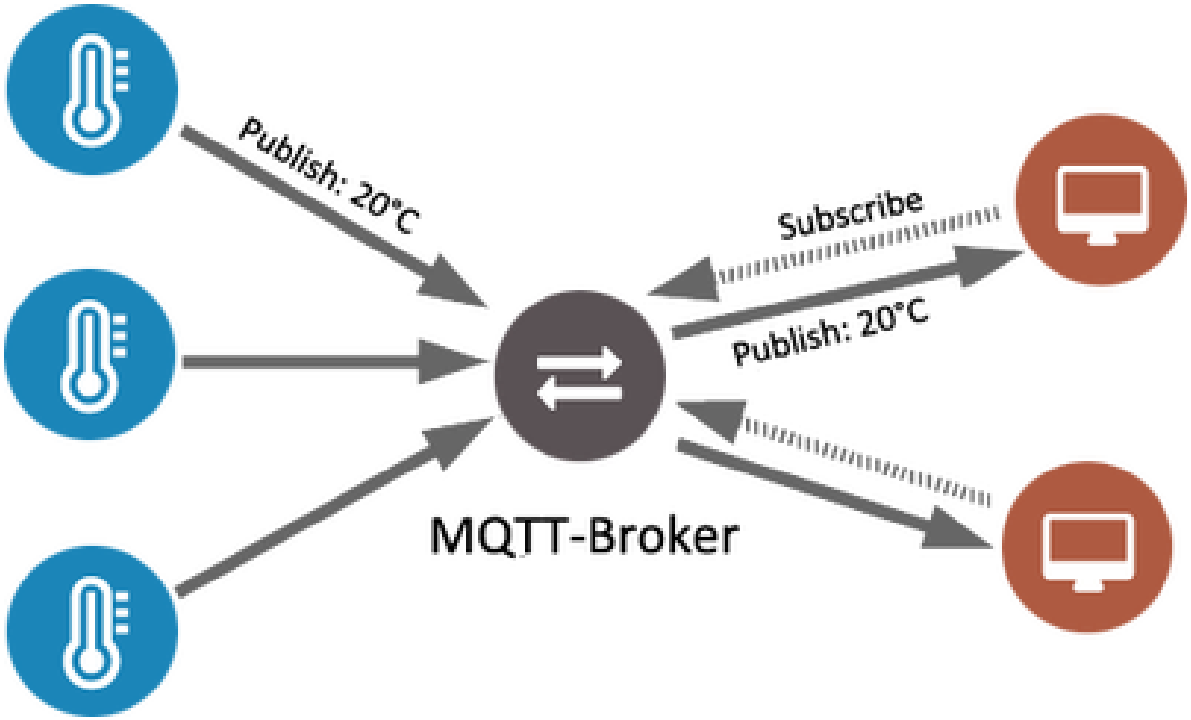
No protocolo MQTT nós temos 3 qualidades de serviço(QoS) e cada conexão com o broker pode especificar qual será utilizada., sendo estas: "no máximo uma vez", "no mínimo uma vez"e "exatamente uma vez".

- QoS 0 - No máximo uma vez
  - Conhecido como fire and forgot (atirar e esquecer), nesse QoS a mensagem é enviada apenas uma vez e não haverá passos seguintes, dessa forma a mensagem não será armazenada, nem haverá um feedback para saber se ela chegou ao destinatário.
  - Esse modo de transferência é o mais rápido, porém o menos seguro já que a mensagem será perdida caso o envio falhe ou o cliente esteja desconectado.

- QoS 1 - Pelo menos uma vez
  - Nesse modo de transferência, a mensagem é entregue pelo menos uma vez, havendo uma espera da recepção de feedback da entrega da mensagem, o chamado PUBACK. Não recebendo o PUBACK, a mensagem continuará sendo enviado até que haja o feedback. Nesse QoS pode acontecer da mensagem ser enviada diversas vezes e ser processada diversas vezes.
  - Para que haja o envio da mensagem mais de uma vez, a mensagem precisa ser armazenada. Ela será excluída após ter recebido o feedback de confirmação do envio.

- QoS 2 - Exatamente uma vez
  - Nesse modo de transferência, a mensagem é entregue exatamente uma vez, necessitando que a mensagem seja armazenada localmente no emissor e no receptor até que seja processada.
  - Para garantir a segurança desse QoS é necessário o envio de 2 pares de request-response(chamado de four-part handshake), onde temos o envio da mensagem(PUBLISH), a resposta de recepção(PUBREC), o aviso do recebimento do PUBREC(PUBREL) e a confirmação de que o processo foi concluído e pode ser feita a exclusão(PUBCOMP).
  - Após o recebimento do PUBREL, o receiver pode excluir a mensagem e ao sender receber o PUBCOMP ele poderá excluir a mensagem.

Broker local

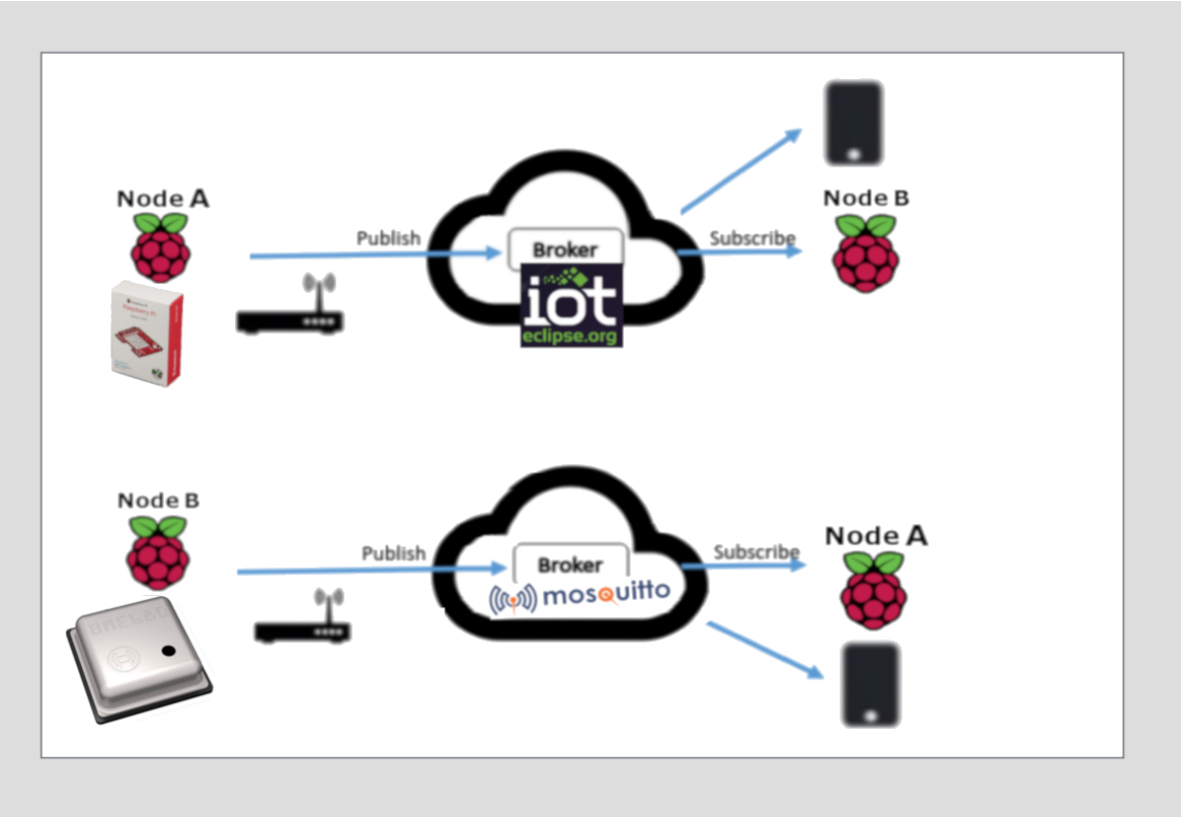


https://www.filipeflop.com/blog/broker-mqtt-com-raspberry-pi-zero-w/

Eclipse Mosquitto

`https://mosquitto.org/`

Broker na nuvem



Cloudmqtt

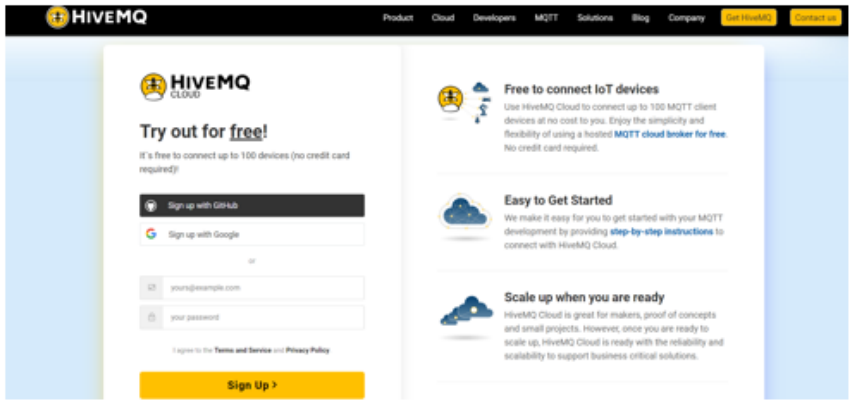
`https://www.cloudmqtt.com/`
Broker pago com planos a partir de 5 dólares por mês

Online Cloud Base MQTT Brokers/Servers

Broker Type	Broker Address and Port	Websocket Support	SSL support
Mosquitto	test.mosquitto.org 1883	Yes Encrypted port 8081 Un-encrypted 8080	Yes 8883 With Client certificate 8884
HiveMQ	broker.hivemq.com 1883	Yes 8000	
Mosquitto	iot.eclipse.org	Yes 80 and 443 (SSL)	Yes 8883
mosca	test.mosca.io 1883		

Broker MQTT com autenticação

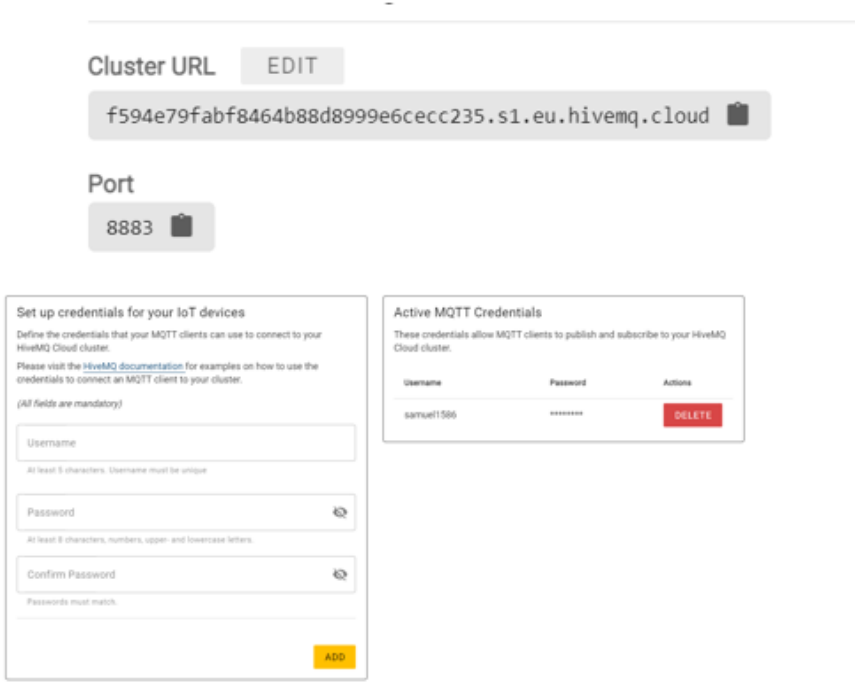
- https://www.hivemq.com/mqtt-cloud-broker/
- Permite criar uma conta grátis através do site, Google e GitHub.
- A comunicação é feita de forma segura usando usuário e senha. Estes dados são criados ao fazer cadastro no aplicativo.



https://www.hivemq.com/mqtt-cloud-broker/

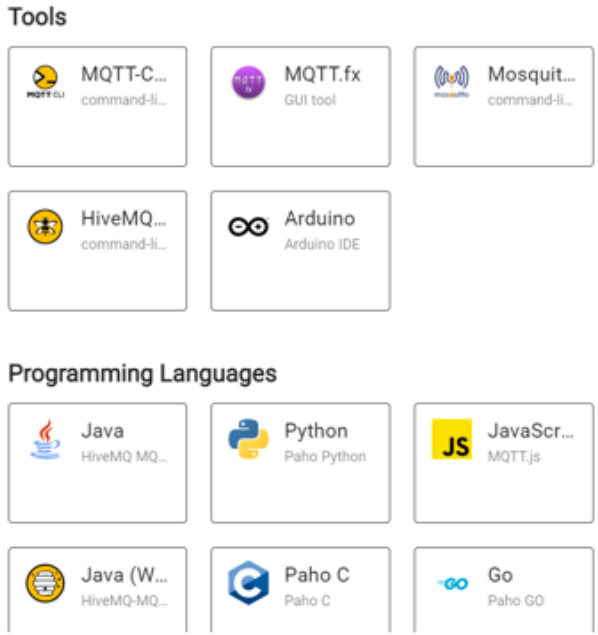
Broker MQTT com autenticação

- O site cria uma URL exclusiva para o usuário;
- Porta 8883.
- É possível criar mais de uma credencial para os usuários.



Broker MQTT com autenticação

- No site é possível obter vários exemplos para diferentes ferramentas e linguagens de programação.
- Infelizmente o código em Arduino não funciona na ESP32 apenas na ESP8266.



Paho Biblioteca python para MQTT

`https://www.eclipse.org/paho/`

```python
import paho.mqtt.client as mqtt
from random import randrange, uniform
import time

mqttBroker ="mqtt.eclipse.org"

client = mqtt.Client("Temperature_Inside")
client.connect(mqttBroker)

while True:
    randNumber = uniform(20.0, 21.0)
    client.publish("TEMPERATURE", randNumber)
    print("Just published " + str(randNumber) + " to topic TEMPERATURE")
    time.sleep(1)
```

https://medium.com/python-point/mqtt-basics-with-python-examples-7c758e605d4

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

def on_message(client, userdata, message):
    print("received message: " ,str(message.payload.decode("utf-8")))

mqttBroker ="mqtt.eclipse.org"

client = mqtt.Client("Smartphone")
client.connect(mqttBroker)

client.loop_start()

client.subscribe("TEMPERATURE")
client.on_message=on_message

time.sleep(30)
client.loop_stop()
```

https://medium.com/python-point/mqtt-basics-with-python-examples-7c758e605d4

mqttfx: Cliente MQTT Java.

`https://mqttfx.jensd.de/`

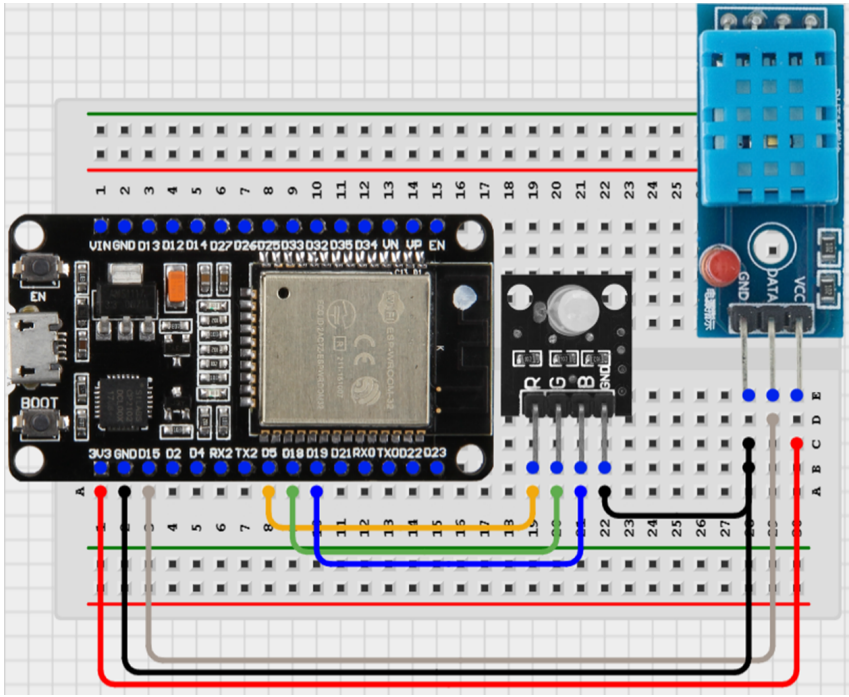
Cliente MQTTX.

MQTT X

`https://mqttx.app/`

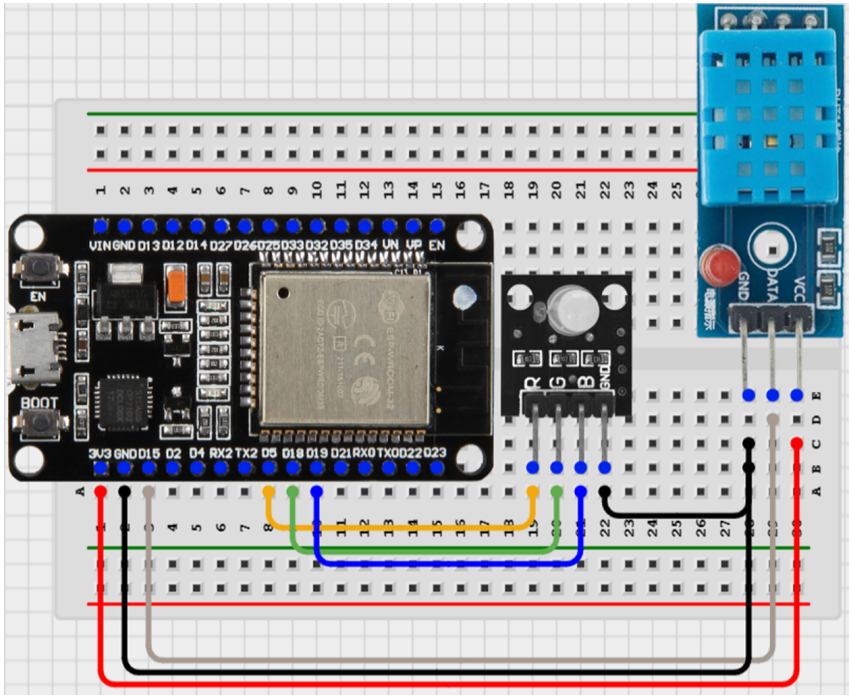
Exemplo mqtt pubsub ESP32

- O ESP32 se conecta à uma rede Wifi e à um broker MQTT;
- Realiza leituras de temperatura e umidade do DHT11;
- Publica as leituras de temperatura em um tópico (it012/temp/mac);



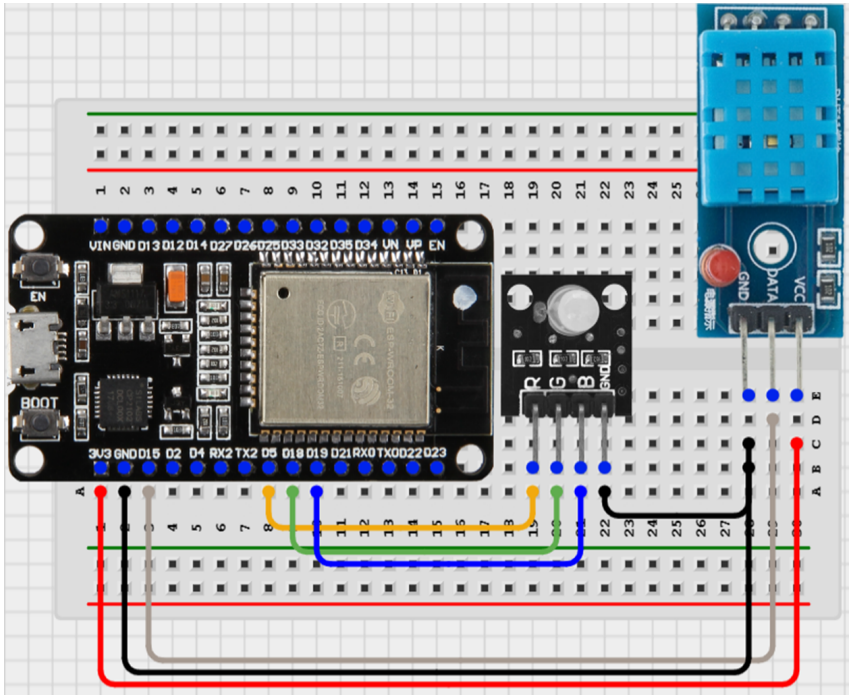
Exemplo mqtt pubsub ESP32

- Publica as leituras de umidade em um tópico (it012/hum/mac); mac é parte do endereço MAC único de cada modulo, usado aqui para gerar um nome de tópico MQTT único para cada dispositivo



Exemplo mqtt pubsub ESP32

- Assina um tópico, que será usado para tratar mensagens de controle recebidas (it012/control/mac); De acordo com o payload recebido, acende ou apaga os LEDs do LED RGB.



Blynk

`https://blynk.io/`