

FIAP

# 1. MQTT

# Objetivos da aula:

- Apresentação e uso do protocolo MQTT

# MQTT – MQ Telemetry Transport

- Protocolo muito simples para publicação e recebimento de mensagens, apropriado para dispositivos com alta latência e baixa largura de banda de comunicação
- A leveza do protocolo, que usa cabeçalhos de poucos bytes, o torna adequado para a comunicação de objetos no cenário da internet das coisas
- Um servidor MQTT, também conhecido como broker, faz o papel de servidor, que gerencia as mensagens publicadas , enviando-as aos clientes que se inscreveram para recebê-las
- Os clientes MQTT são as pontas da comunicação, podendo enviar ou receber mensagens através das operações:
  - Publish: um cliente MQTT publica uma mensagem com determinado tópico
  - Subscribe: um cliente se cadastra no servidor para receber cópias de mensagens com determinado tópico

# Usando o MQTT – conectando a um servidor

- O protocolo MQTT pode ser testado facilmente empregando simples aplicativos para celular
- Após a instalação do programa cliente basta configurar a conexão com o servidor MQTT, também chamado de “Message Broker”, fornecendo seu endereço IP ou URL, na porta padrão 1883
- Para uso em teste, um servidor público pode ser empregado, tais como:
  - [iot.eclipse.org](https://iot.eclipse.org)
  - [test.mosquitto.org](https://test.mosquitto.org)
  - [dev.rabbitmq.com](https://dev.rabbitmq.com)
  - [broker.mqttdashboard.com](https://broker.mqttdashboard.com)

▪ Também é possível instalar e configurar o próprio servidor MQTT

– No caso de uso em uma rede local, com poucas conexões, o servidor

mosquitto ([mosquitto.org](https://mosquitto.org)) é o mais apropriado, por consumir poucos

recursos, podendo ser executado em plataformas de IoT como Raspberry Pi

– Para uso no ambiente da Cloud Computing, onde podemos esperar

milhões de conexões e necessitamos de escalabilidade, precisamos de

um servidor do tipo RabbitMQ ([www.rabbitmq.com](https://www.rabbitmq.com))

# Testando com um webclient

- Alguns servidores MQTT permitem o uso do protocolo WebSocket para o transporte da mensagem
  - Com isso, é possível criar clientes da Web que enviam e recebem mensagens através de um servidor MQTT
- Para testar: <http://www.hivemq.com/demos/websocket-client/>

# Testando com um webclient

Crie um chat, com publish e subscribe.

Sugestão: O tópico pode ser juntando o primeiro nome com o último sobrenome, em “testeTeste”.

– Para enviar uma mensagem a alguém, use o tópico daquela pessoa

exemplo:

testeteste/#



**Need a fully managed MQTT broker?**

Get your own Cloud broker and connect up to 100 devices for free.

[Get your free account](#)

## Connection

● disconnected

Host

Port

ClientID

[Connect](#)

Username

Password

Keep Alive

SSL

☐

Clean Session

☐

Last-Will Topic

Last-Will QoS

Last-Will Retain

☐

Last-Will Message

**Publish****Subscriptions****Messages**

1. QoS (Quality of Service) – indica o nível de verificação de recebimento de mensagens pelo servidor (publish) ou pelo cliente (subscribe)

- QoS 0
- QoS 1
- QoS 2

2. Caracteres coringa na subscrição de tópicos:

- a. Níveis múltiplos (multi-level): o caractere ‘#’
- b. Nível simples (single-level): o caractere ‘+’

# Cliente MQTT no Node-RED

The image shows the Node-RED web interface. On the left, the 'network' category is expanded, displaying various nodes including 'mqtt in', 'mqtt out', 'http in', 'http response', 'http request', 'websocket in', 'websocket out', 'tcp in', 'tcp out', 'tcp request', and 'udp in'. The 'mqtt in' node is selected, and its configuration panel is open. The panel has a title 'Edit mqtt in node' and buttons for 'Delete', 'Cancel', and 'Done'. It contains a 'Server' dropdown menu with the text 'Add new mqtt-broker...' and a 'Topic' text input field. A blue arrow points to the edit icon (pencil) next to the 'Server' dropdown. Below this, there is a section titled 'mqtt in > Add new mqtt-broker config node' with 'Cancel' and 'Add' buttons. This section has four tabs: 'Connection', 'Security', 'Birth Message', and 'Will Message'. The 'Connection' tab is active, showing fields for 'Server' (with the value 'broker.hivemq.com'), 'Port' (with the value '1883'), 'Enable secure (SSL/TLS) connection' (unchecked), 'Client ID' (with the value 'Leave blank for auto generated'), 'Keep alive time (s)' (with the value '60'), 'Use clean session' (checked), and 'Use legacy MQTT 3.1 support' (checked). Blue arrows point to the 'Server' and 'Port' fields.



Edit mqtt out node > **Edit mqtt-broker node**

Delete Cancel Update

**Properties**

Name

**Connection** Security Messages

Server  Port

☒ Connect automatically

☐ Use TLS

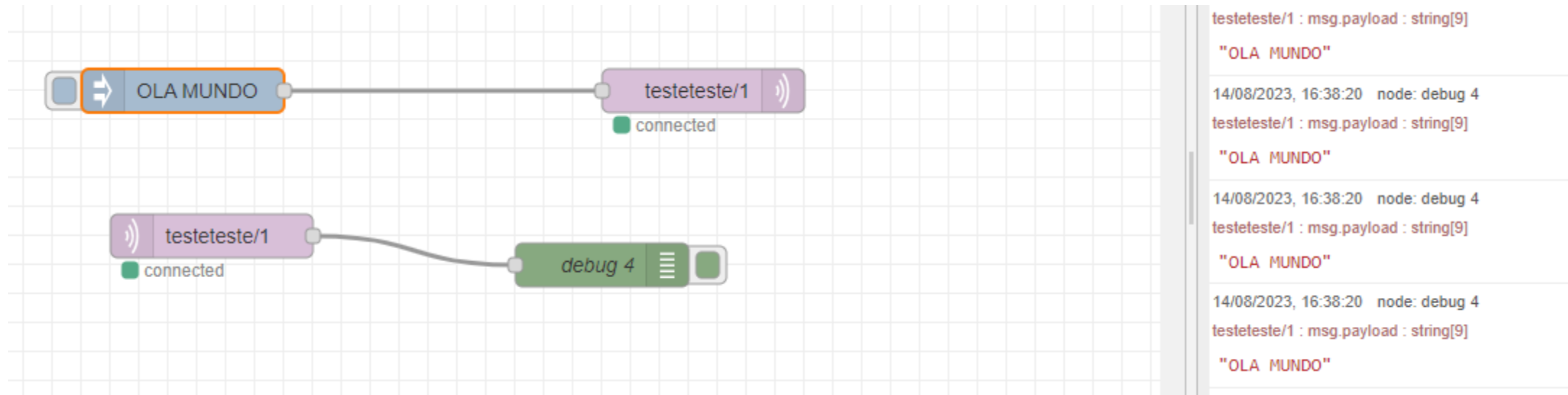
Protocol

Client ID

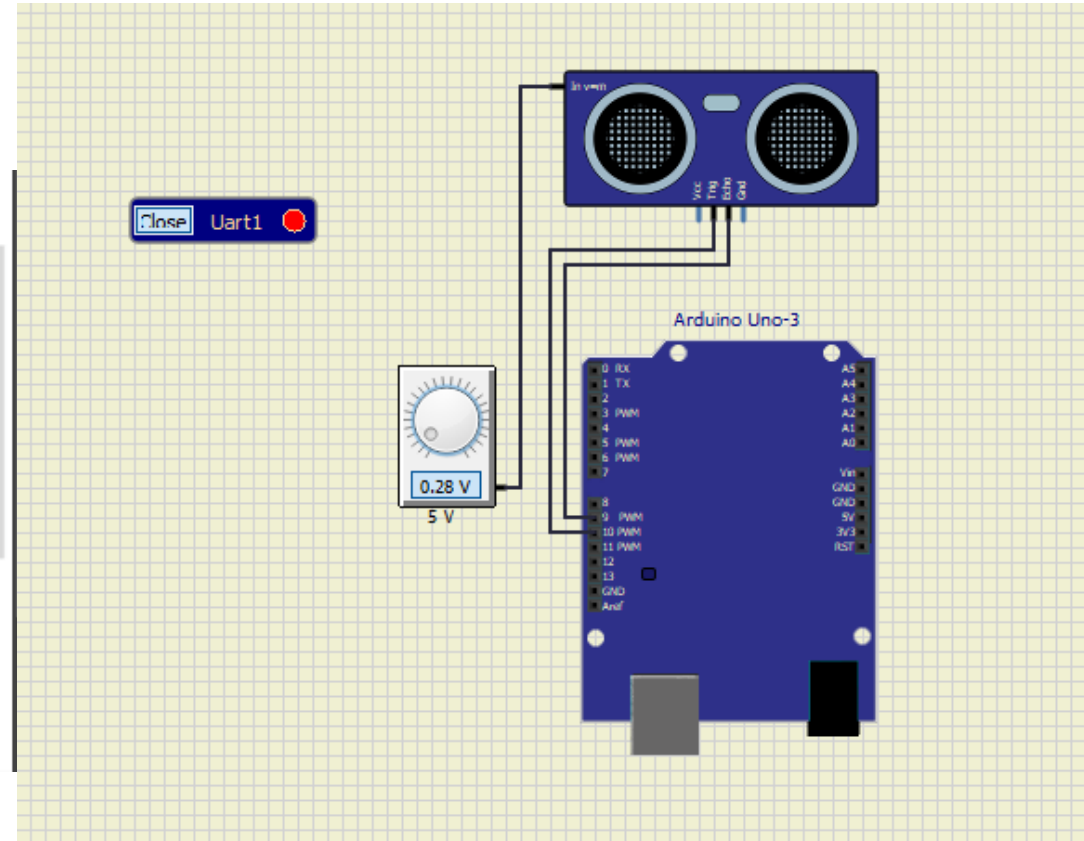
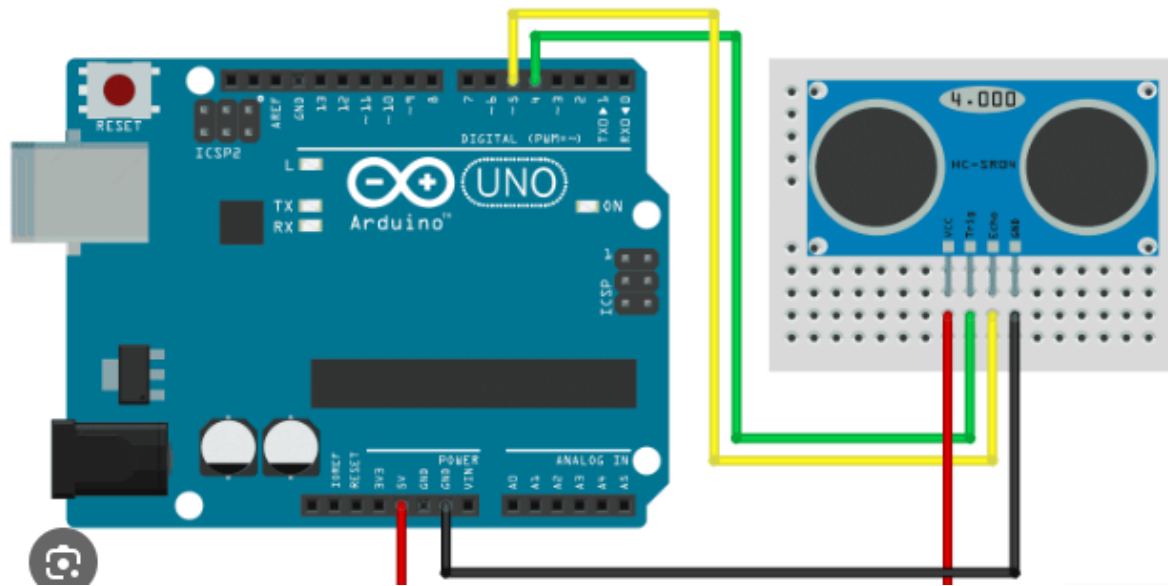
Keep Alive

Session ☒ Use clean session

# Cliente MQTT no Node-RED



# PRATICANDO!!!



# DESAFIO

- Exiba as informações captada pelo sensor ultrassônico e exiba através do protocolo MQTT no NODE-RED de outra bancada.

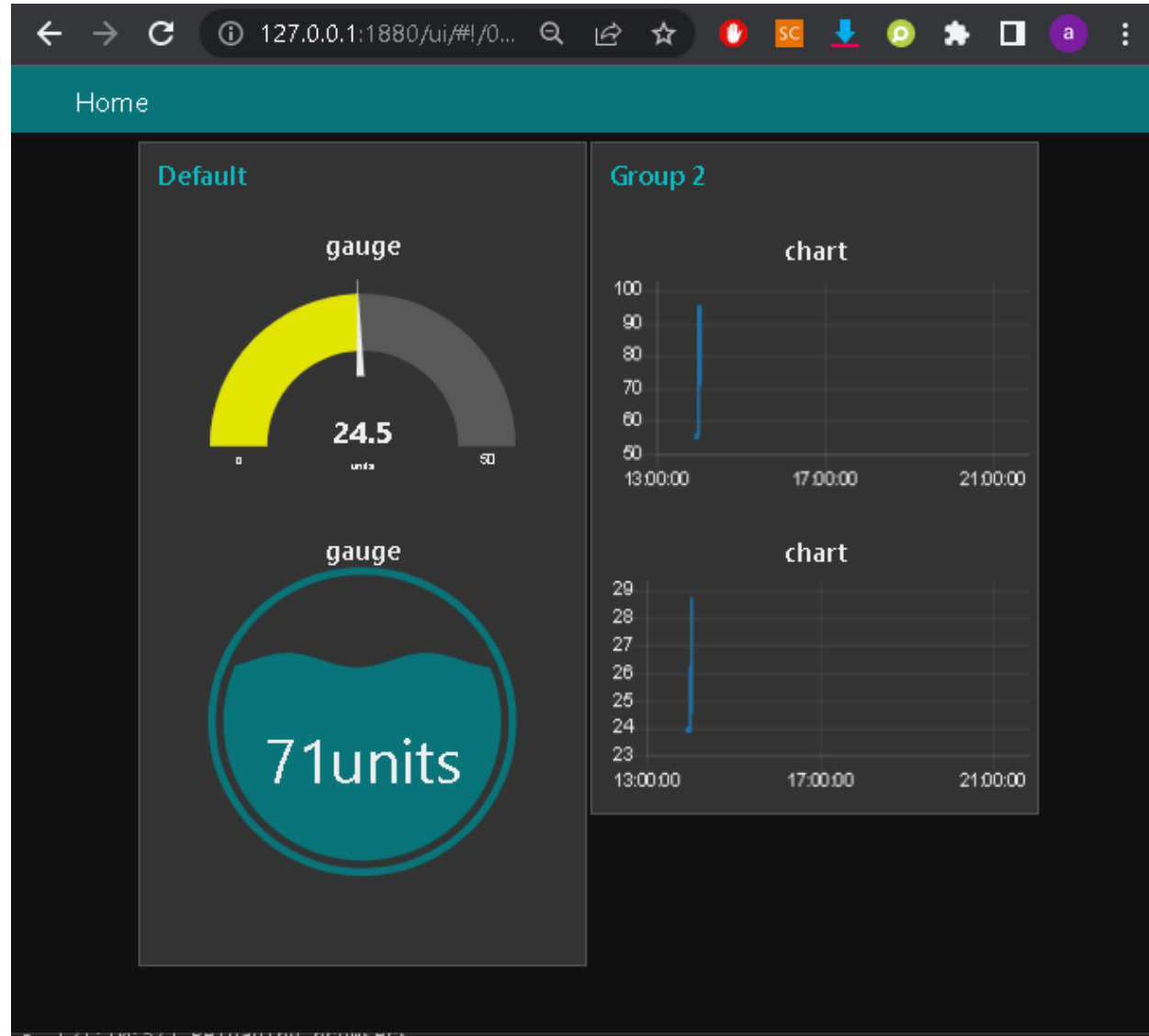


```
▶ { distance: 28.0016 }  
14/08/2023, 16:36:12 node: debug 4  
testeteste/1 : msg.payload : Object  
▶ { distance: 28.0016 }  
14/08/2023, 16:36:12 node: debug 4  
testeteste/1 : msg.payload : Object  
▶ { distance: 28.0016 }  
14/08/2023, 16:36:12 node: debug 4  
testeteste/1 : msg.payload : Object  
▶ { distance: 28.0016 }  
14/08/2023, 16:36:12 node: debug 4  
testeteste/1 : msg.payload : Object  
▶ { distance: 28.0016 }  
14/08/2023, 16:36:12 node: debug 4  
testeteste/1 : msg.payload : Object  
▶ { distance: 28.0016 }
```



- Para o desenvolvimento do sistema de supervisor ficar completo basta adaptar o fluxo que temos no node-RED para receber os tópicos de temperatura e umidade separados e enviar para o dashboard.

- Faça as adaptações necessárias para exibir os valores de temperatura e umidade em 2 gauge e 2 chart como na imagem abaixo:



**Copyright © 2024 Prof. Arnaldo Jr/Yan  
Coelho**

**Todos direitos reservados. Reprodução ou divulgação total ou parcial deste documento é expressamente proibido sem o consentimento formal, por escrito, do Professor (autor).**