ESP8266 CLIENTE MQTT

Antônio Augusto Fasolo Quevedo
Fevereiro de 2019

Esta é a referência para o *firmware* MQTT_Client v1.0, escrito no Arduino IDE para módulos ESP-01 (com ESP8266 e pelo menos 4Mbit de *flash*), pelo Prof. Antonio Quevedo.
Este *firmware* recebe comandos próprios para comunicação com um AP WiFi e com um *Broker* MQTT, através dos pinos de interface serial, **a 19200 baud**. Cada comando para ser aceito deve terminar com "\r\n" (0x0D - 0x0A), ou pelo menos com "\n" (neste caso ignorando o caractere imediatamente anterior), e as respostas terminam sempre com "\r\n".
Há sempre um espaço entre o comando e o primeiro parâmetro, e uma vírgula (sem espaços adicionais) entre parâmetros. Alguns parâmetros devem ser colocados entre aspas (Em C, deve-se representar aspas dentro de uma string com o caractere de *escape*: **\"**)

Sintaxe da interface serial:

Resposta a comando inexistente: INVALID CMD
Resposta geral a parâmetros faltando em um comando válido: MISSING PARS

Mostra a versão de firmware:
VER
Resposta: "MQTT Client vx.x", onde x.x é a versão

Reinicia o módulo (se necessário):
RESTART
Resposta: "Restarting…" e depois uma série de dados do ESP8266, independentes do firmware

Conecta a uma rede WiFi:
CONNWIFI "SSID","password"
Resposta: CONNECT WIFI se conectado; ERROR WIFI se não conseguiu conectar

Retorna o IP atribuído:
GETIP
Resposta: O IP atribuído ou NOIP se não foi atribuído

Retorna o MAC Address:
GETMAC

Resposta: o MAC Address

Conecta a um broker MQTT:
(1) CONNMQTT “xxx.xxx.xxx.xxx”,porta,”clientID”
(2) CONNMQTT “xxx.xxx.xxx.xxx”,porta,”clientID”,”username”,”password”
“xxx.xxx.xxx.xxx” é o IP do Broker
porta é o número da porta do Broker (normalmente 1883)
clientID é o ID do cliente no Broker
“username” e “password” são a autenticação no broker. São opcionais.
A sintaxe (1) é para conexões anônimas (se permitidas). A sintaxe 2 é para conexões autenticadas. Caso falte apenas um dos parâmetros da autenticação (username ou password), o firmware ignora o parâmetro remanescente e considera conexão anônima.
Resposta: CONNECT MQTT se conectado; NOWIFI se não está conectado na wifi; “ERROR: n” se não conseguiu conectar, onde “n” é o código de erro (-4: Timeout; -3: Conexão interrompida; -2: Conexão falhou; -1: Desconexão limpa
Obs: *keepalive* é fixo em 60s, o *loop* de comunicação do *firmware* garante que não haja desconexão do MQTT a menos que o módulo seja desligado ou seja dado o comando de desconexão.

Assina um tópico:
SUBSCRIBE “tópico”
Resposta: OK SUBSCRIBE ou NOT CONNECTED ou ERROR SUBSCRIBE

Remove assinatura de um tópico:
UNSUBSCRIBE “tópico”
Resposta: OK UNSUBSCRIBE ou NOT CONNECTED ou ERROR SUBSCRIBE

Publica uma mensagem em um tópico:
PUBLISH “tópico”,”mensagem”
Resposta: OK PUBLISH ou NOT CONNECTED ou ERROR PUBLISH

Executa um PING:
PING
Resposta: PONG ou NOPONG (se não houve resposta)

Desconecta do Broker de forma “limpa”:
DISCONNECT
Resposta: OK DISCONNECT

Desconecta da rede WiFi:
QUIT

Resposta: OK QUIT

Mensagens recebidas (terminadas em \r\n):

WIFI_DISCONNECTED: Perda de conexão com WiFi

MQTT_DISCONNECTED: Perda de conexão com o Broker

MESSAGE [tópico],[mensagem]: Mensagem recebida do *Broker*

Referências de IDE Arduino em ESP8266 e biblioteca PubSubClient:

http://blogmasterwalkershop.com.br/embarcados/esp8266/instalando-o-firmware-nodemcu-no-esp8266-esp-01/
https://br-arduino.org/2015/11/mqtt-com-o-esp8266.html
https://learn.sparkfun.com/tutorials/esp8266-thing-hookup-guide/installing-the-esp8266-arduino-addon
https://github.com/knolleary/pubsubclient
https://pubsubclient.knolleary.net/api.html#state

Código-fonte do *firmware*:

https://drive.google.com/open?id=1FRmwIJPxlg1C-uQ-t1exyLP20OHqlKdR