Updated:September 1, 2022 / By steve

Arduino -Sending and Receiving JSON Data over MQTT

To send an receive JSON data with Arduino you can use the ArduinoJson library.

The online documentation contains several usage examples, and there is even a book that supports the project.

In this example we are going to send a JSON object using MQTT, receive that object and decode it.

The first step is to create a doc object of sufficient size. You can either create a static one or a dynamic one in the example we use static.

The difference between **StaticJsonDocument** and **DynamicJsonDocument** is covered here.

In version 5 of ArduinoJson we had **StaticJsonBuffer** and **DynamicJsonBuffer** the difference is here

You will see code on the Internet for both v5 and v6. The code in this example uses v6. The difference between v5 and v6 is explained here.

There is a detailed tutorial on calculating the size here.

Values of 128,256 are usually used. If you find that not all objects are serialised then you need to choose a larger value.

The next thing to note is how we add the elements using standard object notation doc["sensor"] = "gps";

For the array element we create a nested array object and then add in the array values.

Polls

What Security do you Currently use on Your Broker (s)

- ○ None
- ○ Username and Password
- ○ ACLs
- ○ Username and Password+ACLs
- ○ Other

Vote

View Results

Email Newsletter

Join me Up

This is what the output looks like using the mosquitto_sub tool.

```
Client mosqsub|8568-ws6 received PUBLISH (d0, q0, r0, m0, 'arduino-test',
{"sensor":"gps","time":1351824120,"data":[48.75608,2.302038]}
```

Once we are finished we use the **serializeJson(doc, out)** function using the doc object and an output buffer which I declared using **char out[128]**.

We then publish using the standard mqtt.publish command. The code is shown below:

```
StaticJsonDocument<256> doc;
doc["sensor"] = "gps";
doc["time"] = 1351824120;

// Add an array.
//
JsonArray data = doc.createNestedArray("data");
data.add(48.756080);
data.add(2.302038);
//doc["data"]=data;
// Generate the minified JSON and send it to the Serial port
//
char out[128];
int b =serializeJson(doc, out);
Serial.print("bytes = ");
Serial.println(b,DEC);
boolean rc = mqttClient.publish("arduino-test", out);
// The above line prints:
// {"sensor":"gps","time":1351824120,"data":[48.756080,2.302
```

Receiving JSON Data

The received data is handled by the callback function. This returns the data in a byte array.

Again we need to create a JSON doc of sufficient size.

Then we can either convert the payload into a string as it is a byte array of we can use it directly as the deserialize function will accept a byte array.

We then convert the received jSON string into a doc object and access the values using the same syntax we used for create them and print out a few just to confirm that it all works. The code is shown below:

```
void callback(char* topic, byte* payload, unsigned int lengt

char str[length+1];
```

Hi - I'm Steve and welcome to my website where you can learn how to build IOT systems using MQTT.

Search

Search ...

My Youtube Channel

```
  Serial.print("Message arrived [");
  Serial.print(topic);
  Serial.print("] ");
  int i=0;
  for (i=0;i<length;i++) {
  Serial.print((char)payload[i]);
  str[i]=(char)payload[i];

  }
  str[i] = 0; // Null termination
  Serial.println();
  //practise string
  //char json[] = "{\"sensor\":\"gps\",\"time\":1351824120,\"d

  StaticJsonDocument <256> doc;
  deserializeJson(doc,payload);

  // deserializeJson(doc,str); can use string instead of paylo
  const char* sensor = doc["sensor"];
  long time = doc["time"];
  float latitude = doc["data"][0];
  float longitude = doc["data"][1];

  Serial.println("latitude =");
  Serial.println(latitude,2);
  Serial.println(sensor);
```


Notes:

The latest version of the MQTT client (2.8) allows you to change the size of the MQTT output buffer using setBufferSize(). However When I tried this it gave compile errors for my Ethernet board and I need to go back to v2.7.

Download Sketch

DOWNLOAD

Related Tutorials and Resources

- JSON Basics For Beginners-With Examples and Exercises
- Using the Arduino PubSub MQTT Client
- Send and Receive Integers and Floats with Arduino over MQTT
- ArduinoJson
- How to Send and Receive JSON Data Over MQTT with Python
- MQTT Pubsub client

Please rate? And use Comments to let me know more

13 comments

Shreya says:
September 6, 2021 at 12:21 pm

I have a problem. I want to publish the live data I am feeding into an excel sheet into the cloud using MQTT. I do not know how to do it, but I want to do it this way. Could you please help me out?

Thanks and Regards

Reply

steve says:
September 6, 2021 at 12:23 pm

The script that comes with the tutorial should do that just change the address for the MQTT broker
rgds
steve

Reply

JorgeF says:
August 22, 2021 at 9:25 pm

Great explanation. It helped me a lot.
Thank you very much.

Reply

Mack says:
July 30, 2021 at 7:02 pm

Hi Steve,
Hi Steve,
So the problem is with JSON data? No

I have edited the names arduino-1 and arduino-2 for each device and used const char* declarations
const char* PubSubName = “arduino-2”; //each device needs a unique name
const char* PubSub_jdoc = “arduino-test”;

IT now WORKS correctly but after 30secs the MKR stopped subscribing.
This has been fixed by using the reconnect function from

https://github.com/knolleary/pubsubclient/blob/master/examples/mqtt_basic/mqtt_basic.ino

regards Mack

Reply

steve says:

July 31, 2021 at 3:26 pm

So it is working ok Now?

rgds

steve

Reply

Mack Richards says:

July 29, 2021 at 6:04 pm

I am able to successfully connect and publish and subscribe to my local network (192.168.100.38) Raspi Broker using your example code. mqtt-demo-1.ino.txt on a Arduino Mega. However I am unable to receive (subscribe) to the same published data from my Arduino MKR1010 which is on the same local network.

I don't need to publish any data from the MKR but only subscribe to my broker. The MKR1010 connects successfully to the broker in setup() but the callback function is never called by mqttClient.loop().

```
void loop(){
mqttClient.loop();
delay(5000);
}
```

Do I need any additional code in loop? Have I deleted too much code from loop().

What editing if any is required in the callback(... function?

Reply

steve says:

July 29, 2021 at 6:37 pm

Can you send me the entire script

rgds

steve

Reply

Mack says:

July 30, 2021 at 3:46 am

Yes.

If I enable the loop code then the code executes as a publisher with the callback

subscribing but I don't wish to publish from this device, only subscribe to my broker.

my code:

```
//from http://www.steves-internet-guide.com/arduino-sending-receiving-json-mqtt/

//#include //not required, part of Arduino15
//#include
#include //v1.8.10
#include //v2.8
#include //v6.8.13
#include "secrets.h"

// Update these with values suitable for your network.
//byte mac[] = { 0xDE, 0xED, 0xBA, 0xFE, 0xFE, 0xED };
//byte mac[] = {0x90,0xA2,0xDA,0x10,0xA5,0xF9}; // my Ethernet Shield #2 Arduino POE W5500

byte mac[6]; // the MAC address of your Wifi Module
IPAddress ip;
IPAddress gateway;

///////please enter your sensitive data in the Secret tab/secrets.h
char ssid[] = SECRET_SSID; // your network SSID (name)
char pass[] = SECRET_PASS; // your network password (use for WPA, or use as key for WEP)
//IPAddress ip(192, 168, 100, 61);
const char* server = "192.168.100.38"; //rpimqtt broker

//EthernetClient ethClient;
WiFiClient wifiClient;
//PubSubClient mqttClient(ethClient);
PubSubClient mqttClient(wifiClient);
///
//receive data
///
void callback(char* topic, byte* payload, unsigned int length) {

char str[length+1];
Serial.print("Message arrived [");
Serial.print(topic);
Serial.print("] ");
int i=0;
for (i=0;i<length;i++) {
Serial.print((char)payload[i]);
str[i]=(char)payload[i];

}
str[i] = 0; // Null termination
Serial.println();
//practise string
```

```
//char json[] = "{\"sensor\":\"gps\",\"time\":1351824120,\"data\":
[48.756080,2.302038]}";

StaticJsonDocument doc;
deserializeJson(doc,payload);

// deserializeJson(doc,str); can use string instead of payload
const char* sensor = doc["sensor"];
long time = doc["time"];
float latitude = doc["data"][0];
float longitude = doc["data"][1];

Serial.println(sensor);
Serial.print("latitude =");
Serial.println(latitude,2);

}//end callback

void setup()
{
// Open serial communications and wait for port to open:
Serial.begin(115200);
for (int i = 0; i < 4; i++){
delay(1000);
}
Serial.println();
Serial.println(__FILE__);
Serial.println();

//Ethernet.begin(mac);
// Allow the hardware to sort itself out
//delay(1500);

// attempt to connect to Wifi network:
Serial.print("Attempting to connect to WPA SSID: ");
Serial.println(ssid);
while (WiFi.begin(ssid, pass) != WL_CONNECTED) {
// failed, retry
Serial.print(".");
delay(5000);
}

Serial.println("You're connected to the network");
Serial.println();
// print the received signal strength:
long rssi = WiFi.RSSI();
Serial.print(F(" signal strength (RSSI):"));
Serial.print(rssi);
Serial.println(F(" dBm"));
//print the local IP address
```

```
ip = WiFi.localIP();
Serial.print(F("Local: "));
Serial.println(ip);

gateway = WiFi.gatewayIP(); // print your gateway address:
Serial.print(F("GATEWAY: "));
Serial.println(gateway);
mac_address();

mqttClient.setServer(server, 1883);
mqttClient.setCallback(callback);

if (mqttClient.connect("arduino-1")) {
// connection succeeded
Serial.println("Connected ");
boolean r= mqttClient.subscribe("arduino-test");
Serial.println("subscribe ");
Serial.println(r);
}
else {
// connection failed
// mqttClient.state() will provide more information
// on why it failed.
Serial.println("Connection failed ");
}

}//end setup

void loop(){
/*
StaticJsonDocument doc;
doc[“sensor”] = “gps”;
doc[“time”] = 1351824120;

// Add an array.
//
JsonArray data = doc.createNestedArray(“data”);
data.add(48.756080);
data.add(2.302038);
//doc[“data”]=data;
// Generate the minified JSON and send it to the Serial port.
//
char out[128];
int b =serializeJson(doc, out); //no of bytes in out[128]
Serial.print(“publishing bytes = “);
Serial.println(b,DEC);
Serial.println();
boolean rc = mqttClient.publish(“arduino-test”, out);
// The above line prints:
// {“sensor”:”gps”,”time”:1351824120,”data”:[48.756080,2.302038]}
```

```
*/
delay(5000);

mqttClient.loop();

}//end loop

void mac_address() {
WiFi.macAddress(mac);
Serial.print(F("MAC: "));
for (int i = 5; i > 0; i--) {

if (mac[i] < 16) Serial.print(F("0")); //less than F+1
Serial.print(mac[i],HEX);Serial.print(F(":"));
}
Serial.print(mac[0],HEX);
Serial.println();
}
```

Reply

steve says:

July 30, 2021 at 8:36 am

What are you using to publish. Does my script without editing work. That is the one that pubs and subs.
Rgds
Steve

Reply

Mack says:

July 30, 2021 at 2:17 pm

Hi Steve, I am using your example mqtt-json.ino (note in my first comment to you I used the wrong file name) as the publisher. It is running on a Arduino Mega with an Ethernet shield. It functions exactly as per your web site description. My Broker is a Raspberry pi with Mosquitto installed. The only publisher code change is the Broker address.
The MKR1010 has the same code installed but with wifi replacing ethernet. It too will publish and subscribe exactly as per your website using your code in the void loop().
I don't want the MKR to publish but only subscribe to the Mega. They are both on the same local network 192.168.100.xx.
I have previously used both devices and the same broker to publish from the Mega and subscribe from the MKR. The messaging was plain text. I am able to publish/subscribe several topics in sequence. This arrangement uses the ArduinoMqttClient.h library and examples from Arduino https://docs.arduino.cc/tutorials/mkr-1000-wifi/mkr-1000-mqtt-device-to-device.
I found your mqtt JSON example and tried to implement it as the data transfer is

structured.
regards Mack

steve says:
July 30, 2021 at 3:09 pm

So the problem is with JSON data?

Fred Verhoeckx says:
June 12, 2021 at 2:48 pm

Hi Steve, thanks for your very clear series of blogs! I always return to them as I am learning to program my ESP32 boards.

I have a 2 questions relating to the receiving part:
1)
Is it true that the byte-wise conversion of payload into str is equivalent to strcpy(str, payload) ? I use that and it looks that the results are the same. However, the rest of my code fails, so I ask this to be sure that it is OK.

2)
When I receive an MQTT message (I use AsyncMQTT) I do not want to act immediately on the content of each message. Rather, I want to keep the message payloads in e.g. a stack and handle them in the order of receipt just before my board is going into deep sleep. So I want to create an array of char arrays or maybe better: an array of pointers at char arrays.

Do you have any suggestions for that?

Reply

steve says:
June 12, 2021 at 7:35 pm

Hi
Not sure about the strcpy method but tou can pass it the payload in the deserialise function and it works
StaticJsonDocument <256> doc;
deserializeJson(doc,payload);

For the 2nd part I would use a queue
https://www.arduino.cc/reference/en/libraries/queue/

https://github.com/SMFSW/Queue

Rgds
Steve

Reply

Leave a Reply

Your email address will not be published. Required fields are marked *

Comment *

Name *

Email *

Website

Post Comment