

ChatGPT



Überprüfe die Grammatik und Rechtschreibung des Textes. Des weiteren schreibe einen kleinen Absatz zur Reconnect Funktion. Außerdem gib mir den Text im Markdown Format zurück:

MQTT

Wie bereits in der theoretischen Ausarbeitung besprochen, ist MQTT ein Protokoll zur Übertragung von Daten, bei welchem der Publisher, der Prototyp, ein Topic abonnieren muss um genau auf dieses Topic die Daten zu versenden.

```
{caption="Senden der Sensordaten an den MQTT Server" .cpp}
const char *mqtt_server = "mqtt.contrude.eu";
const char *mqtt_username = "contrude";
const char *mqtt_password = "HaG1$Vk62!cWv";
const char *mqtt_domain = "contrude/";
const int ship_number = 1;
const int mqtt_port = 1883;
// Number for this node and also number for the container
int nodeNumber = 1;

String mqtt_publish = String(mqtt_domain) + ship_number + "/" + nodeNumber;

void publishSensorData() {
  sensors_event_t a, g, temp;
  mpu.getEvent(&a, &g, &temp);

  client.publish((String(mqtt_publish) + "/temperature").c_str(), String(bme.readTemperature()).c_str());
  client.publish((String(mqtt_publish) + "/pressure").c_str(), String(bme.readPressure()).c_str());
  client.publish((String(mqtt_publish) + "/humidity").c_str(), String(bme.readHumidity()).c_str());
  client.publish((String(mqtt_publish) + "/vibration").c_str(), String(a.acceleration.x).c_str());
}

void reconnect() {
  while (!client.connected()) {
    Serial.print("Attempting MQTT connection...");
    if (client.connect("ESP32Client", mqtt_username, mqtt_password)) {
      Serial.println("connected");
    } else {
      Serial.print("failed, rc=");
      Serial.print(client.state());
      Serial.println(" retrying in 2 seconds...");
      delay(2000);
    }
  }
}
```

In unserem Fall habe ich das Abonnieren des Topics so skalierbar wie möglich gemacht. Soll heißen, dass ich das Topic aufgesplittet habe.

Das eigentliche Topic hat diese Form:
contrude/Schiffsnummer/Containernummer/Sensordatenart
Um nun es möglich zu machen, den Prototypen in jeden Schiff und jeden Container zu plazieren, habe ich die einzelnen Teile in folgende variablen ausgelagert.

1. ``const char *mqtt_domain = "contrude/";``
2. ``const int ship_number = 1;``
3. ``const int mqtt_port = 1883;``
4. ``int nodeNumber = 1;``

Diese einzelnen Teile, habe ich dann in einem String zusammengefasst:
``String mqtt_publish = String(mqtt_domain) + ship_number + "/" + nodeNumber;``

Hier ist der korrigierte Text mit einem zusätzlichen Absatz zur **Reconnect-Funktion** im **Markdown-Format**:

Mqtt Documentation

MQTT

Wie bereits in der theoretischen Ausarbeitung besprochen, ist MQTT ein Protokoll zur Übertragung von Daten, bei dem der Publisher, also der Prototyp, ein Topic abonnieren muss, um genau auf dieses Topic die Daten zu versenden.

```
const char *mqtt_server = "mqtt.contrude.eu";
const char *mqtt_username = "contrude";
const char *mqtt_password = "HaG1$Vk62!cWv";
const char *mqtt_domain = "contrude/";
const int ship_number = 1;
const int mqtt_port = 1883;
// Number for this node and also number for the container
int nodeNumber = 1;

String mqtt_publish = String(mqtt_domain) + ship_number + "/" + nodeNumber;

void publishSensorData() {
  sensors_event_t a, g, temp;
  mpu.getEvent(&a, &g, &temp);

  client.publish((String(mqtt_publish) + "/temperature").c_str(), String(bme.readTemperature()).c_str());
  client.publish((String(mqtt_publish) + "/pressure").c_str(), String(bme.readPressure()).c_str());
  client.publish((String(mqtt_publish) + "/humidity").c_str(), String(bme.readHumidity()).c_str());
  client.publish((String(mqtt_publish) + "/vibration").c_str(), String(a.acceleration.x).c_str());
}

void reconnect() {
  while (!client.connected()) {
    Serial.print("Attempting MQTT connection...");
    if (client.connect("ESP32Client", mqtt_username, mqtt_password)) {
      Serial.println("connected");
    } else {
      Serial.print("failed, rc=");
      Serial.print(client.state());
      Serial.println(" retrying in 2 seconds...");
      delay(2000);
    }
  }
}
```

In unserem Fall habe ich das Abonnieren des Topics so skalierbar wie möglich gemacht. Das bedeutet, dass ich das Topic aufgesplittet habe.

Das eigentliche Topic hat diese Form: **contrude/Schiffsnummer/Containernummer/Sensordaten**

Um es zu ermöglichen, den Prototypen in jedes Schiff und jeden Container zu platzieren, habe ich die einzelnen Teile in folgende Variablen ausgelagert:

1. ``const char *mqtt_domain = "contrude/";``
2. ``const int ship_number = 1;``
3. ``const int mqtt_port = 1883;``
4. ``int nodeNumber = 1;``

Diese einzelnen Teile habe ich dann in einem String zusammengefasst: ``String mqtt_publish = String(mqtt_domain) + ship_number + "/" + nodeNumber;``

Reconnect-Funktion

Die ``reconnect()``-Funktion stellt sicher, dass die Verbindung zum MQTT-Server bei Verbindungsabbrüchen wiederhergestellt wird.

Falls die Verbindung unterbrochen wird oder der Client nicht verbunden ist, versucht die Funktion in einer Schleife kontinuierlich, die Verbindung wieder aufzubauen.

Bei jedem Verbindungsversuch wird überprüft, ob eine erfolgreiche Verbindung hergestellt werden kann. Falls dies gelingt, wird eine Bestätigung im Serial-Monitor ausgegeben ("connected"), andernfalls erfolgt ein neuer Versuch nach einer Wartezeit von zwei Sekunden. Dieses Verhalten gewährleistet eine zuverlässige Datenübertragung, auch bei Netzwerkproblemen oder Neustarts des Geräts.

Ich habe die Grammatik- und Rechtschreibfehler korrigiert und einen erklärenden Abschnitt zur **Reconnect-Funktion** hinzugefügt. Falls du weitere Anpassungen möchtest, gib einfach Bescheid!

ChatGPT kann Fehler machen. Überprüfe wichtige Informationen.