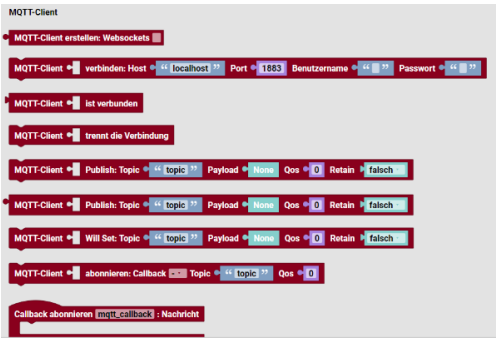


IoT mit MQTT und Node-RED

In der Software Robo Pro Coding gibt es u.a. den Punkt Cloud/MQTT mit den entsprechenden Blöcke. Während das Thema Cloud im Netz gut beschrieben ist, gibt es zum Thema MQTT kaum Informationen. In der Hilfe von Robo Pro Coding werden zwar die einzelnen Blöcke beschrieben, aber es fehlt ein komplettes Beispiel um alle Zusammenhänge zu erkennen. Weiterhin wird in der Hilfe beschrieben, dass der TXT 4.0 einen eigenen MQTT-Broker besitzt, es fehlt aber der Hinweis, dass auch ein Node-RED inklusive Dashboard zur Verfügung stehen. Dieser Artikel soll an zwei einfachen Beispielen den Zusammenhang aller o.g. Elemente zeigen. Dazu ist es allerdings notwendig, dass Grundkenntnisse von Node-RED vorhanden sind!

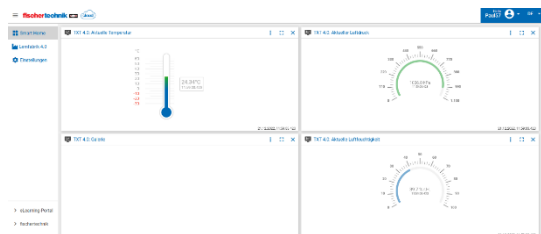


Begriffsbestimmung

IoT

Der Begriff Internet der Dinge oder Internet of Things (IoT) steht für eine vernetzte Welt aus "smarten", d.h. intelligenten, Geräten. Diese IoT -Geräte verhalten sich wie Computer und sind lokal oder über das Internet mit anderen Geräten vernetzt.

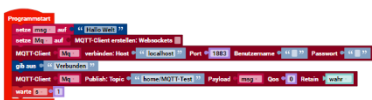
Hier das Beispiel von ft:
Robotics Add On: IoT



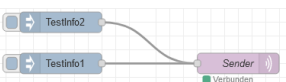
MQTT

MQTT steht für „Message Queuing Telemetry Transport“. Es ist ein offenes Netzwerkprotokoll für Machine-to-Machine-Kommunikation, dass die Übertragung von Telemetriedaten in Form von Nachrichten zwischen Geräten ermöglicht. MQTT funktioniert nach dem Publisher- / Subscriber-Prinzip und wird über einen zentralen Broker betrieben. Das bedeutet, dass Sender und Empfänger keine direkte Verbindung haben.

Veröffentlichen



Datenausgabe über TXT 4.0



Datenausgabe über Node-RED
192.168.x.x:1880



Datenausgabe über Microcontroller
z.B. ESP8266 mit MBE280

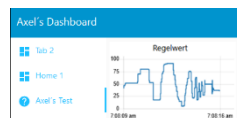


MQTT-Broker
127.0.0.1

Abonnieren



Datenanzeige über TXT 4.0



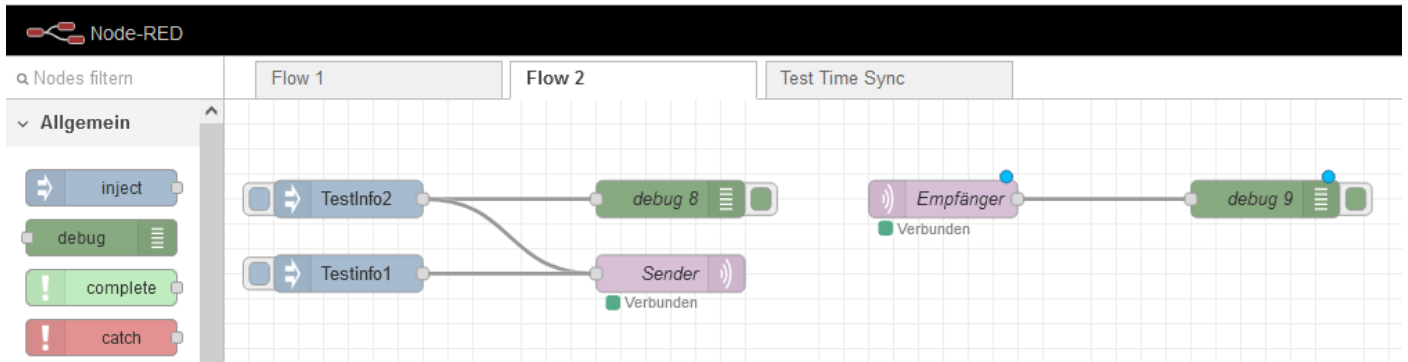
Datenanzeige über Dashboard
192.168.x.x:1880/ui



Datenanzeige über Dashboard
auf dem Smartphone

Node-RED

Node-RED ist ein von IBM entwickeltes grafisches Entwicklungswerkzeug. Die Software ermöglicht es, Anwendungsfälle im Bereich des Internets der Dinge mit einem einfachen Baukastenprinzip umzusetzen. Die einzelnen Funktionsbausteine werden durch Ziehen von Verbindungen verbunden. Zur Zeit gibt es von Fischertechnik noch keine Funktionserweiterungen für die Ein- und Ausgabe des TXT 4.0.



Dashboard

Das Dashboard ist eine grafische Oberfläche auf der die Daten, die in Node-RED erzeugt oder ausgelesen werden, visualisiert werden.



Zugriff auf die Software unter dem TXT 4.0:

MQTT-Broker: 127.0.0.1 oder Lokalhost

Node-RED: 192.168.x.x:1880

Dashboard: 192.168.x.x:1880/ui

Für .x.x muss die entsprechende Adresse des eigenen TXT 4.0 eingetragen werden.

Test des MQTT-Brokers

Als erstes sollte die Funktionsfähigkeit des MQTT-Brokers getestet werden. Dazu wird eine Nachricht an den Broker gesendet und gleich auch wieder abgerufen. Beide Texte werden dann im Debug-Fenster angezeigt.

26.5.2024, 08:16:48 node: debug 8
msg.payload : string[11]
"erster Test"

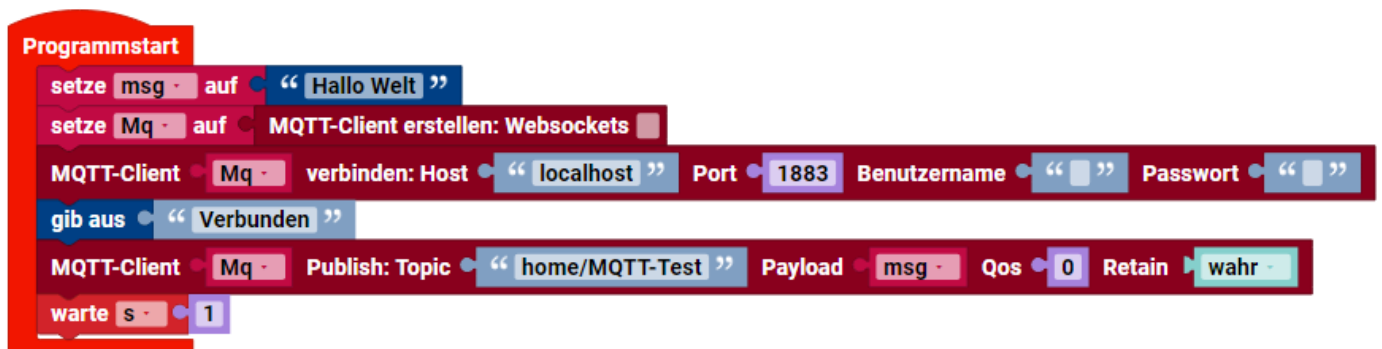
26.5.2024, 08:16:48 node: debug 9
home/MQTT-Test : msg.payload : string[11]
"erster Test"

Server: 127.0.0.1:1883
Topic: home/MQTT-Test

Wichtig ist dabei, dass in den beiden Nodes (Sender und Empfänger) die Adresse und der Port des TXT eigenen Brokers eingetragen sind. Ebenfalls muss der Topic-Name (Thema) bei beiden gleich sein.

Daten an MQTT senden (und über Node-RED auslesen)

- Die zu übertragene Nachricht wird in die Variable msg gespeichert.
- Zuerst wird der MQTT-Client erstellt und in einer Variablen abgespeichert.
- Im nächsten Schritt verbindet sich der Client mit dem angegebenen Host (Hier der TXT 4.0 unter Localhost) und dem entsprechenden Port. Ansonsten gibt es Broker auch als Open-Source-Produkte, wie Eclipse Mosquitto oder HiveMQ.
- Als Kontrolle der Verbindungsaufnahme erfolgt nun die Ausgabe eines kurzen Textes.
- Im letzten Schritt wird nun die vorbereitete Information (Variable msg) auf dem MQTT-Broker unter dem entsprechenden Topic abgelegt. Das ist ein „schwarzes Brett“ oder „Thema“, auf welches sich Sender und Empfänger einigen müssen. Dabei sind Topics von der Funktions- und Schreibsyntax Ordern in einem Filesystem sehr ähnlich.
- Qos (Quality of Service) ist eine Vereinbarung zwischen Sender und Empfänger für die Garantie der Nachrichtenübermittlung. 0 = höchstens einmal, 1 = mindestens einmal, 2 = genau einmal

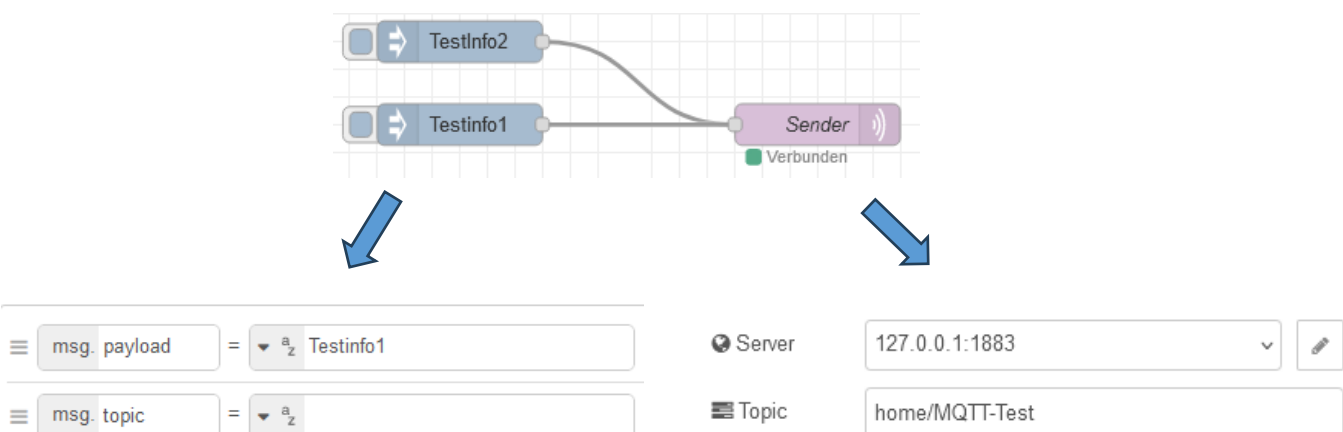


Auf der Empfängerseite (Hier Node-RED vom TXT 4.0) wird nun die empfangene Nachricht in der Empfänger- Node übernommen und über die Debug-Node im Debugfenster ausgegeben.



Daten vom MQTT empfangen (die von Node-RED gesendet wurden)

Hier wird die Nachricht von einem Inject-Node als String an die Node Sender übergeben, der die Nachricht auf dem angegebenen MQTT-Broker unter dem vordefinierten Topic ablegt.



- Zuerst wird wieder der MQTT-Client erstellt und in einer Variablen abgespeichert.
- Dann wird ebenfalls eine Verbindung mit dem Host hergestellt.
- In der nun folgenden Schleife wird der vereinbarte Topic angegeben und schließlich über die Funktion Callback die Nachricht auf der Konsole ausgegeben.

The image shows a RoboProCoding script for an MQTT client. It starts with a 'Programmstart' block, followed by 'setze Mq auf MQTT-Client erstellen: Websockets'. Then, 'MQTT-Client Mq verbinden: Host 127.0.0.1 Port 1883 Benutzername Passwort' is used to connect. A 'dauerhaft wiederholen' loop contains a 'make MQTT-Client Mq abonnieren: Callback mqtt_callback Topic home/MQTT-Test Qos 0' block and a 'warte s 0.5' block. Below this, a 'Callback abonnieren mqtt_callback : Nachricht' block is followed by 'gib aus Nachricht payload'. To the right, a 'Konsole' window shows the output: 'Programm startet...', 'Testinfo1', and 'TestInfo2'.

Fazit

Mit diesen Kenntnissen, sollte nun möglich sein, das Thema IoT umfangreich einzusetzen. So können mit dem Microcontroller ESP8266 z.B. Wetterdaten o.ä. mit RoboProCoding ausgewertet werden. Andersherum können auch die Eingangswerte vom TXT 4.0 auf dem Smartphone über das Dashboard beobachtet werden.