IoT-Hackathon

Systemübersicht

MQTT - Kurzübersicht

MQTT – Was ist das?

MQTT → Message Queuing Telemetry Transport

Publish & Subscribe System
Ermöglicht Senden & Empfangen Nachrichten als Client

Ermöglicht einfache Kommunikation zwischen verschiedenen Geräten
Geringe Bandbreite

→ Gut geeignet für IoT Applikationen

Anwendungsfälle:

- Sende einen Befehl, um Geräte ein- oder auszuschalten
- Lese Daten von einem Sensor und publiziere sie

MQTT - Broker

- Empfängt alle Nachrichten
- Filtert die Nachrichten
- Publiziert Nachrichten an die Abonnenten (Subscriber)
- Bekannter Broker: Mosquitto

MQTT – Publish/Subscribe

Device 1 veröffentlicht Daten unter einem Topic (publish)

Device 2 hat dieses Topic abonniert (subscribe)

→ Device 2 erhält die Daten

MQTT - Topics

- Spezifizieren den „Titel“ unter dem Daten veröffentlicht werden
- Topics werden als String angegeben
- Topics können Hierarchien bilden – die Level werden durch "/" getrennt
- Beispiel: home/office/lamp
- **Topics sind case-sensitive:** home/office/lamp != home/office/LAMP

INFLUXDB – Kurzeinführung

INFLUXDB

- Timebase Database (NOSQL)
- Zur Speicherung von zeitbasierten Daten (Meßreihen)
- Eine Meßreihe wird identifiziert über
  - Name
  - Tags (Markierungen)
  - Meßwert(e)
  - Zeitstempel

Datenübertragung über Line-Protokoll

<measurement>[,<tag_key>=<tag_value>[,<tag_key>=<tag_value>]] <field_key>=<field_value>[,<field_key>=<field_value>] [<timestamp>]

Name | Tags | Meßwerte | Zeitstempel

https://www.influxdata.com/
https://docs.influxdata.com/influxdb/v1.6/

Grafana - Kurzeinführung

Grafana - Übersicht

- Analytik Plattform für jegliche Art von Metrik
- Abfragen, Visualisieren, Benachrichtigen egal wo die Daten gespeichert sind
- Erstellen, Erkunden und Teilen von Dashboards

https://grafana.com/
http://docs.grafana.org/

Darstellung von Temperatur und Luftfeuchtigkeit

- Graph Panel mit 2 Sensor Daten
- Herkunft der Daten aus einer InfluxDB

Neues Dashboard anlegen
Auswahl eines geeigneten Panel

Neues Dashboard anlegen
Globale Dashboard Einstellungen

Neues Dashboard anlegen
Panel Konfigurieren

Neues Dashboard anlegen
Datenquelle auswählen

- Beispiel Daten sind in einer InfluxDB gespeichert.
- Konfigurierte Datenquelle nennt sich hier *telegraf*

Neues Dashboard anlegen
Auswahl Daten

- Im Beispiel wurden die Sensordaten im Measurement mit Name *sensors* gespeichert

Neues Dashboard anlegen
Mehrere Metriken hinzufügen

Neues Dashboard anlegen
Anpassung der Einheiten

- Im Beispiel sollen 2 Sensor Daten (Temperatur in Grad Celcius und Luftfeuchtigkeit in %) mit unterschiedlichen Skalen dargestellt werden

Neues Dashboard anlegen
Verschieben einer Query auf die rechte Y-Achse

Neues Dashboard anlegen
Darstellungseigenschaften

- Einstellung des Zeitstrahl (hier die letzten 6 Stunden)
- Automatischer Refresh (hier alle 30 Sekunden)

Node-RED - Kurzeinführung

Node-RED

- grafisches Entwicklungswerkzeug von IBM
- Umsetzung von Anwendungsfällen mit Baukastenprinzip
- Funktionsbausteine werden durch Ziehen von Verbindungen verbunden
- Aufteilung verschiedener Abläufe in einzelne Arbeitsbereiche (Flows) möglich
- Datenverarbeitung und -manipulation mit JavaScript möglich

Node-RED

Node-RED : hap
Nicht sicher | hap:1880/#flow/79101932.d19748
Node-RED
Deploy
filter nodes
TQ
TQ
Flow 1
info
debug
dashboard
input
inject
catch
status
link
mqtt
http
websocket
tcp
udp
Watson IoT
sun events
serial
Information
Flow "79101932.d19748"
Name Flow 1
Status Enabled
Flow Description
None
Flow-Leiste
Node-Bereich
Nodes können per Drag&Drop in den Arbeitsbereich geschoben werden
Infobereich
Informationen zum ausgewählten Node, Debug-ausgaben

Node-RED

Node-RED : hap
Nicht sicher | hap:1880/#flow/79101932.d19748
Node-RED
Deploy
filter nodes
Hue
input
inject
catch
status
link
mqtt
udp
Watson IoT
sun events
serial
mqtt
Doppelklick öffnet das Editierfenster
Edit mqtt in node
Delete
Cancel
Done
node properties
Server
HAP MQTT
Topic
/mein/topic
QoS
2
Name
Name
Das Editierfenster variiert je nach Node.
Hilfe zum Node im Infobereich.
node settings
info
debug
Information
Node
"4af075c3.56ceec"
Type
mqtt in
show more
Node Help
Connects to a MQTT broker and subscribes to messages from the specified topic.
Outputs
payload
string | buffer
a string unless detected as a binary buffer.
topic
string
the MQTT topic, uses / as a hierarchy separator.
qos
number
0, fire and forget - 1, at least once - 2, once and once only.
retain
boolean
true indicates the message was retained and may be old.
Details

Node-RED

Node-RED : hap
Nicht sicher | hap:1880/#flow/79101932.d19748
Node-RED
Deploy
Full
Deploys everything in the workspace
Modified Flows
Only deploys flows that contain changed nodes
Modified Nodes
Only deploys nodes that have changed
filter nodes
Hue
TQ
Flow 1
Watson IoT
sun events
serial
output
debug
link
mqtt
http response
websocket
tcp
udp
Watson IoT
play audio
/mein/Topic
msg.payload
meine Datenverarbeitung
/anderer/Topic
Inputs
payload
topic
qos
Ein Nodeausgang kann mit beliebig vielen Nodeeingängen per Drag&Drop verbunden werden.
Änderungen am Flow werden mittels Deploy-Button gespeichert und aktiviert.

Node-RED

Node-RED : hap
Nicht sicher | hap:1880/#flow/79101932.d19748
Node-RED
Deploy
filter nodes
Hue
TQ
TQ
info
debug
dashboard
all nodes
input
output
function
social
mobile
storage
analysis
advanced
Raspberry Pi
logic
boolean logic
HueMagic
formats
dashboard
/mein/Topic
msg.payload
meine Datenverarbeitung
3.9.2018, 17:39:52 node: Wetter
msg.payload : Object
{ weather: "Clouds", detail: "Wolkenbedeckt", icon: "04d", tempk: 295.25, tempc: 22.1 … }
3.9.2018, 17:39:53 node: b16cf7db.6e7af8
msg.payload : number
22.1
3.9.2018, 17:39:54 node: 1092e647.a9ab9a
msg.payload : Object
{ weather: "Clouds", detail: "Wolkenbedeckt", icon: "04d", tempk: 295.25, tempc: 22.1 … }
3.9.2018, 17:49:52 node: Wetter
msg.payload : Object
{ weather: "Drizzle", detail: "Leichtes Nieseln", icon: "09d", tempk: 295.23, tempc: 22 … }
3.9.2018, 17:49:53 node: b16cf7db.6e7af8
msg.payload : number
22
3.9.2018, 17:49:54 node: 1092e647.a9ab9a
msg.payload : Object
{ weather: "Drizzle", detail: "Leichtes Nieseln", icon: "09d", tempk: 295.23, tempc: 22 … }
3.9.2018, 17:59:55 node: Aktuelles Wetter Lünen
Ausgaben können mit Hilfe des Debug-Nodes im debug-Fenster beobachtet werden

Node-RED

Im UI-Fenster können Werte beobachtet und Buttons und andere Steuerelemente geklickt werden.
Neue Elemente können mit den Dashboard-Nodes hinzugefügt werden.

Hardware

IT Architektur

Node A - Hardware

Schaltplan

D1mini
WeMos.cc
RESET
5V
R1
R2
S1
S2
GND
WAKEUP
GPIO16/WAKE
GPIO14/CLK
GPIO12/MISO
IO13/MOSI/RXD2
D5
D6
D7
H1
BME280
Sensor
D1
D2
GND
5V
GPIO5/SCL
GPIO4/SDA
H2
JP1
WS2812b RGB LED breakout connector
DIN
JP4
DOUT - not connected
(just to solder the pins)

Umsetzung

TX
RX
D1
D2
VIN
GND
SCL
SDA
BME/BMP280
3V3
D8
D7
D6
D5
D0
A0
RST
E1
E2
TQ 1337
SQS 2018

SQS 2018

Firmware

- Datenübertragung per WiFi

- MQTT Publish
  - /campus/nodeA → NodeId, Sensorwerte (T, P, H), Zustand Taster, Farbe (alle 2 sec.)
    Beispiel: sensors,node=home temp=21.02,hum=43.12,press=10200.13,s0=1,s1=1,color=255_0_0
  - /campus/Button0 → NodeId, Zustand Taster 0, Farbe („sofort“)
    Beispiel: CubusX#0#120_10_30
  - /campus/Button1 → NodeId, Zustand Taster 1, Farbe(„sofort“)
    Beispiel: CubusOne#1#220_43_130

- MQTT Subscribe
  - /campus/feedback → node=Farbe
    Beispiel:
    1=240_32_120 – setze Farbe LED 0 von allen Nodes
    CubusOne=0_255_0 – setze Farbe von LED 1 von Node CubusOne

Node B - Hardware

Schaltplan

D1mini
WeMos.cc
RESET
5V
R1
R2
S1
S2
GND
WAKEUP
GPIO16/WAKE
GPIO14/CLK
GPIO12/MISO
IO13/MOSI/RXD2
D5
D6
D7
H1
BME
VIN
GND
SCL
SDA
D1
D2
GPIO5/SCL
GPIO4/SDA
H2
JP1

NodeB – Aufbau, Schritt 1 (Drahtbrücken und Widerstände)

NodeB – Aufbau, Schritt 2 (Kabel)

NodeB – Fertig

Entwicklungsumgebung

Programmierung mit der Arduino Entwicklungsumgebung

Entwicklungsboard auswählen – WEMOS D1 mini

Zusatzbibliotheken installieren

Aufgaben

DIY – Vorschläge für erste Schritte

Programmierung ESP8266

- ☐ **NodeB entsprechend Plan aufbauen**
- ☐ **Sensor auslesen**
- ☐ **Taster auswerten**
- ☐ **LED Ring ansteuern**
- ☐ **WiFi Verbindung herstellen**
- ☐ **MQTT – Daten senden**
- ☐ **MQTT – Daten empfangen**

Top-Level Programme

- ☐ **Node-RED – MQTT Daten auswerten**
- ☐ **Node-RED – MQTT Daten senden**
- ☐ **GRAFANA – Datenauswerten**

WLAN:
SSID: IoT Campus
PW: campus2018

(mqttbroker.fritz.box = 10.0.0.1)

Node-Red: http://10.0.0.1:1880
Grafana: http://10.0.0.1:3000 U:admin, P:admin
MQTT: IP: 10.0.0.1 U: tq, P: campus2018

Links

- GITHUB Repo mit Beispielen - https://github.com/sqs-dach/IoT-on-Campus-2018

- MQTT - https://www.heise.de/developer/artikel/MQTT-Protokoll-fuer-das-Internet-der-Dinge-2168152.html

- INFLUX DB – Line Protokoll
  https://docs.influxdata.com/influxdb/v1.6/write_protocols/line_protocol_tutorial/

- NODE RED - https://nodered.org/

- GRAFANA - https://grafana.com/
  Getting started - http://docs.grafana.org/guides/getting_started/

Idee zur Umsetzung

- Bewertungssystem implementieren, welche auf Tastendruck die Ring-Anzeige anpasst
- Ringanzeige in Abhängigkeit von Umweltdaten steuern

Vielen Dank für Ihre Aufmerksamkeit.

sqs.com

SQS Software Quality Systems AG

Stollwerckstraße 11
51149 Köln, Deutschland
Telefon: +49 2203 9154-0
Telefax: +49 2203 9154-15
info-germany@sqs.com

Unsere Werte

Leidenschaft für Qualität	Professionelle Partnerschaft	Exzellente Mitarbeiter	Verantwortung	Ehrlichkeit & Integrität