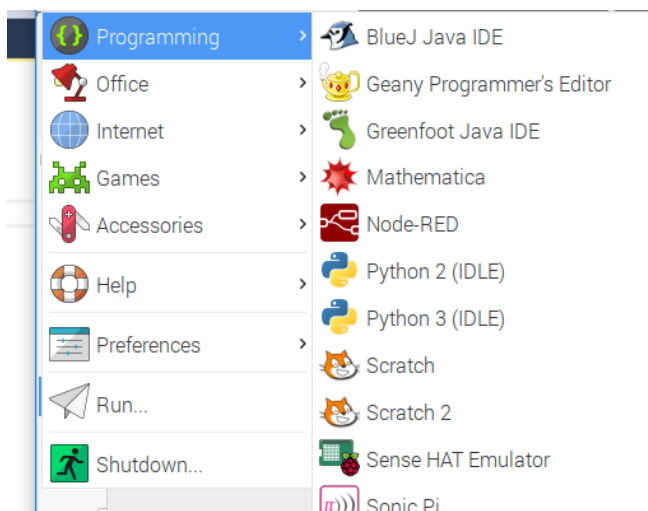


# NodeRED und SenseHat auf Raspberry Pi 3 für Industrie 4.0

von Mario Schnalzenberger

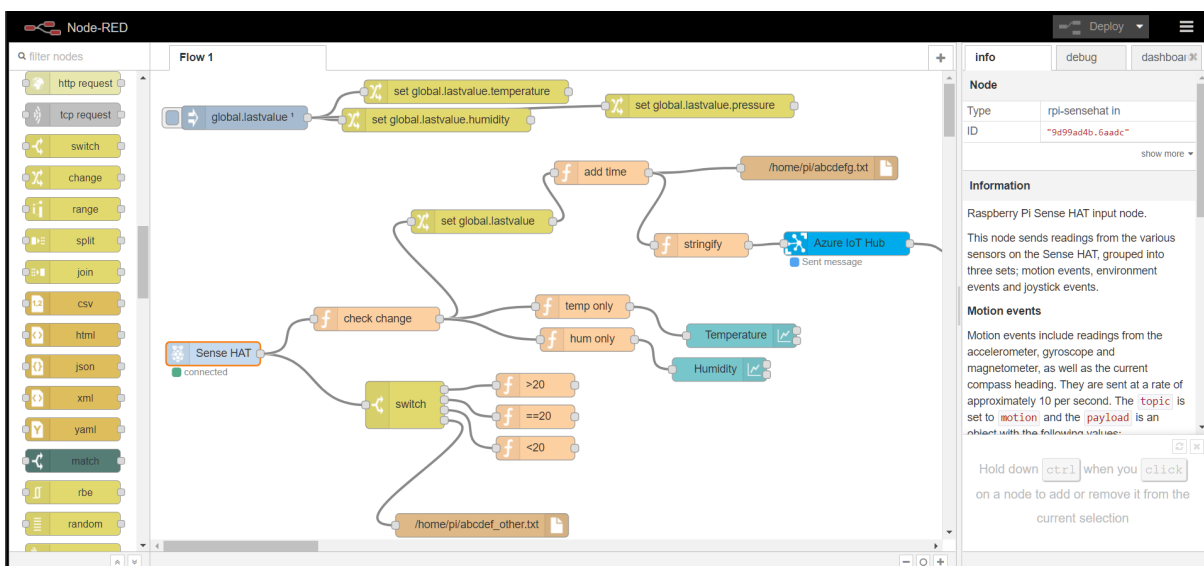
## Simulation und Python

Bereits installiert sind Node-RED und SenseHat Simulator für Python. Damit kann man mit Python tools schon arbeiten. Das ist als Basis und zum Kennenlernen ganz hilfreich.



## NodeRed

Node Red ist ein visualisiertes Ablauf-Tool (FLOWS) um Datentransfer zwischen automatisierten Elementen zu programmieren.



Der Ablauf läuft (hier) von links nach rechts. Die Quellen von Informationen (Input-Nodes) können mit Verarbeitungsknoten (nodes) verbunden werden um die Daten für die Outputs vorzubereiten.

Im Beispiel der Node „check change“ der die globale Variable „global.lastvalue“ verwendet um minimale absolute Veränderungen der Temperatur oder der Humidity zu prüfen. Nur in diesen Fällen wird die Nachricht weitergeleitet.

Der „Azure IoT Hub“ schickt die Daten dann in die Cloud.

Die Dashboard – Tools bieten eine geschickte, einfache Möglichkeit ein Dashboard für ein Device zu generieren (liegt alles am Raspi).

Bei mir 10.0.0.152:1880 => NODE RED

Bzw. 10.0.0.152:1880/ui => Temperatur – Dashboard

Um die Dashboards / IoT / zu installieren:

```
sudo apt install nodejs
cd ~/.node-red/node_modules/
npm install node-red-node-pi-sense-hat-simulator



# von microsoft
npm install node-red-contrib-azureiothubnode

# node-red dashboards
npm install node-red-dashboard
```

## In der Azure Cloud

Auf portal.azure.com kann man dann einen IoT Hub anlegen (Registrierung vorausgesetzt).

Der IoT Hub ist in einer Free-Version gratis (max. 8000 Msgs pro Tag)

<input type="checkbox"/>	NAME <small>↕</small>	TYP <small>↕</small>	STANDORT <small>↕</small>	
<input type="checkbox"/>	 Raspi3MarioStreamJob	Stream Analytics-Auftrag	Nordeuropa	...
<input type="checkbox"/>	 Raspi3TestHub	IoT Hub	Nordeuropa	...

Zusätzlich braucht man noch eine Stream Analytics Job mit dem man das STREAM Processing der Daten macht. Bei mir „RasPi3MarioStreamJob“, der die Daten (von allen Devices) aus dem IoT Hub nimmt und verarbeitet. Das wird mit SQL programmiert:

```

/* -----
First: Select Events to DB (cold path)

01.03.2018 15:35:52> Device: [RasPi3Mario], Data:
[{"temperature":34.49,"humidity":21.32,"pressure":929.5,"time":"2018-03-
01T14:35:52.760Z","device":"RasPi3Mario"}]

*/
SELECT
    EventProcessedUtcTime
    , EventEnqueuedUtcTime
    , PartitionId
    , IoTHub.IoTHub.MessageId AS MessageId
    , IoTHub.IoTHub.ConnectionDeviceId AS ConnectionDeviceId
    , IoTHub.IoTHub.EnqueuedTime AS EnqueuedTime
    , IoTHub.IoTHub.StreamId as StreamId
    , TRY_CAST([temperature] as FLOAT(53)) as [temperature]
    , TRY_CAST([humidity] as FLOAT(53)) as [humidity]
    , TRY_CAST([pressure] as FLOAT(53)) as [pressure]
    , TRY_CAST([time] as datetime) as timeValue
    , device
INTO
    [RaspberryTableDB]
FROM
    [IoTHub]
;

/* -----
First: Select Events to Power BI Dashboard (hot path)

01.03.2018 15:35:52> Device: [RasPi3Mario], Data:
[{"temperature":34.49,"humidity":21.32,"pressure":929.5,"time":"2018-03-
01T14:35:52.760Z","device":"RasPi3Mario"}]

*/
SELECT
    IoTHub.IoTHub.ConnectionDeviceId AS ConnectionDeviceId
    , IoTHub.IoTHub.StreamId as StreamId
    , TRY_CAST([temperature] as FLOAT(53)) as [temperature]
    , TRY_CAST([humidity] as FLOAT(53)) as [humidity]
    , TRY_CAST([pressure] as FLOAT(53)) as [pressure]
    , TRY_CAST([time] as datetime) as timeValue
    , device
INTO
    [PowerBi]
FROM
    [IoTHub]
;

```

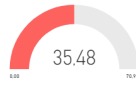
In Power Bi kann man dann mit dem konfigurierten Dataset (das durch die Anlage im Azure Stream Analytics erzeugt wird – erst mit dem Empfang der ersten Daten!) einen Bericht erzeugen:

Raspi3MarioStream

Seite 1



Durchschnitt von temperature



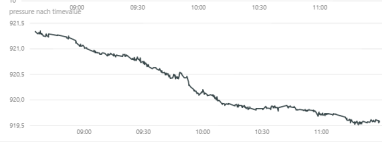
Durchschnitt von pressure



humidity



Anzahl von connectionsdeviceid



03.02.18 11:29:29

Spätestes Datum: timevalue

03.02.18 08:35:24

Frühestes Datum: timevalue