Programming your own IoT device

EDUDEMOS

How Connected Devices Are Shaping Tomorrow

Created By

IoTB Zukunftswerkstatt

UE: Programming your own IoT device (LoRaWAN)

Brief description

In this teaching unit, learners programme their own IoT devices using a low-code environment.

Contact person

Fabian Reifler |

1. General conditions

Framework conditions	
Time frame	10 - 14 lessons
Target group	people interested in technology
Prerequisites	enjoyment and talent in working with computers

2. Objectives of the teaching unit

Learners should familiarise themselves with the most important components of an IoT system and also establish a link to the hardware.

Competence-orientated learning objectives

You will lern with the article Quickstart with the IoT Cube ...

- how to create a project in makeCode
- how to load extensions (e.g. for sensors and actors)
- how to edit and download a program to the iot cube.
- how to create dashboards on ThingsBoard to visualise data.

3. Teaching material

All teaching materials are listed here in chronological order.

Links For learners	Content / activity	Links for teachers
Quickstart with the IoT Cube	quick start guide	
https://paeber.github.io/lora-at-interface/	soft- and hardware documentation, it is in German, sorry...	

4. Teacher's preparations for implementation

4.1 Hardware

- LoRaWAN Gateway to TTN (if there is no Gateway reachable)
- IoT Cube Class set

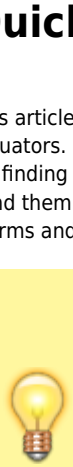
4.2 Software

- NodeRED Instances (one for each student)

4.3 Other

Quickstart with the IoT Cube

This article explains the first steps in programming the LoRaWAN-capable IoT cube. The IoT cube comes in a box with various sensors and actuators. Each cube is labelled with a number when you look at the IoT Cube from the side, eg. **eui-2023-c-05**. The number is important for finding your measurement data on the internet. With the IoT (internet of things), we want to measure „things“ in our environment and send them to the internet. We can then process our measurement data online and display it on a map, for example. We can also trigger alarms and send something back to our IoT cube, for example to control a greenhouse or top up feed in a birdhouse.



Competence-orientated learning objectives

You will lern with the article Quickstart with the IoT Cube ...

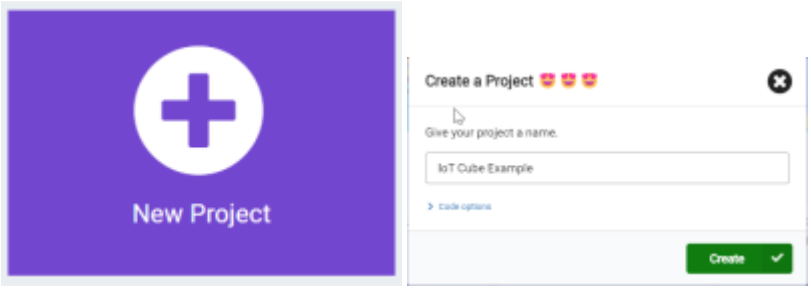
- how to create a project in makeCode
- how to load extensions (e.g. for sensors and actors)
- how to edit and download a program to the iot cube.
- how to create dashboards on ThingsBoard to visualise data.

1. Create a project and load extensions

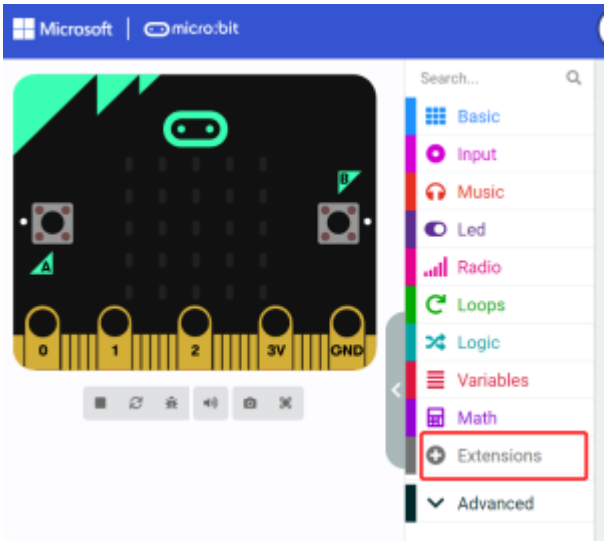
To program the IoT Cube, you need the following web application:

https://makecode.microbit.org

Create a new project and choose any name:



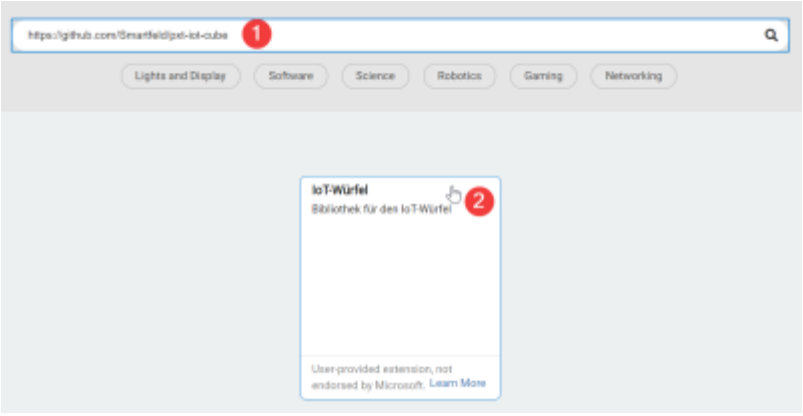
To support the IoT Cube, you need to load an extension. Proceed as follows:



Copy the following path into the search field and press ENTER.

```
https://github.com/Smartfeld/pxt-iot-cube
```

Click on the extension to install it.



To support more sensors and actors, that are included in the IoT- Cube Box, repeat the procedure with this two paths:

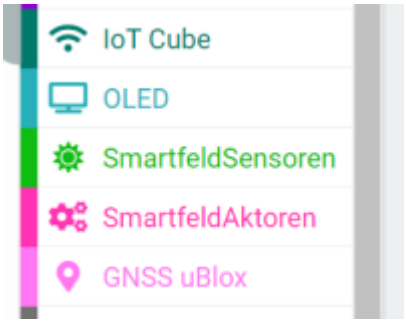
Some sensors and actors:

```
https://github.com/Smartfeld/pxt-sensorikAktorikSmartfeld
```

GPS Module:

```
https://github.com/Smartfeld/pxt-ubx-gnss/
```

After loading the extensions, the following blocks should be available:



2. Example with light level (example 1)

2.1 Goal (example 1)

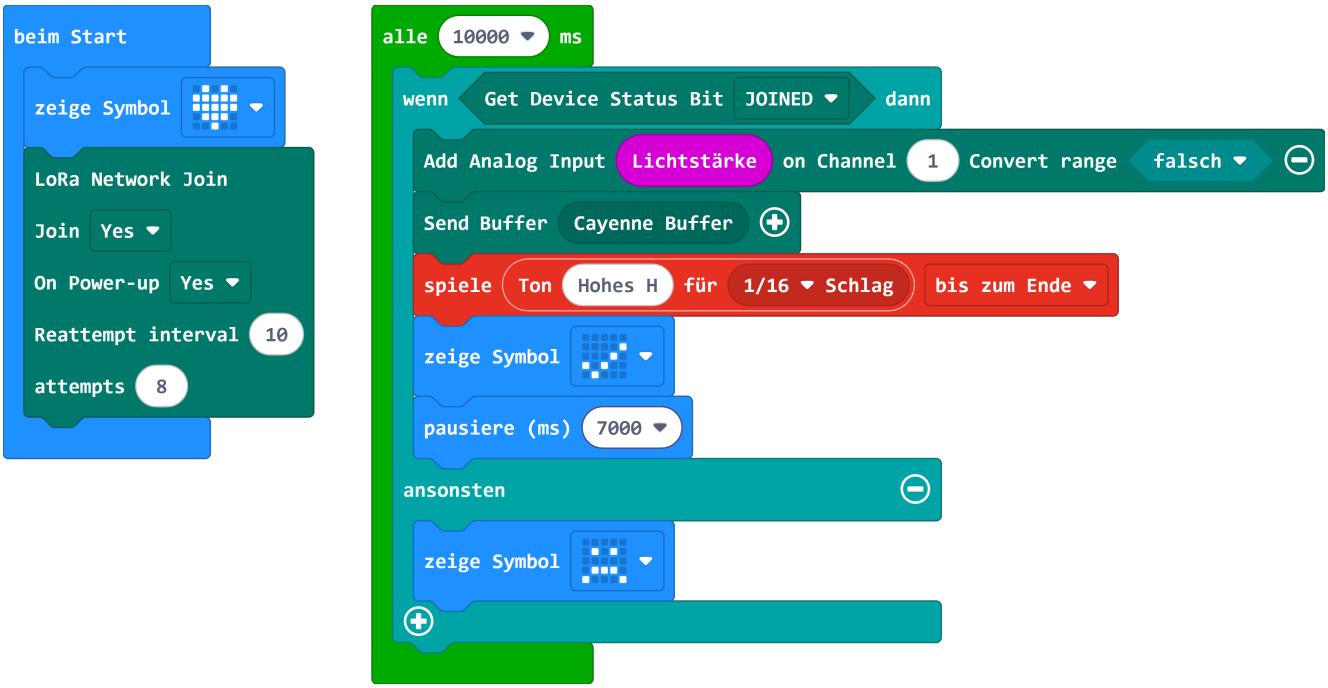
As a first example, we want to send a light level via LoRaWAN. The MicroBit can measure the light intensity via the LEDs on the front. We will transmit the value every 30 Seconds.

2.2 Hardware (example 1)

- Connect the Y USB Cable to the IoT Cube
  - the plug marked red on the back of the housing (only power)
  - the other one to the micro:bit on the top of the housing (power + data)
- Connect the USB-A Connector with your computer

2.3 Software for the IoT Cube (example 1)

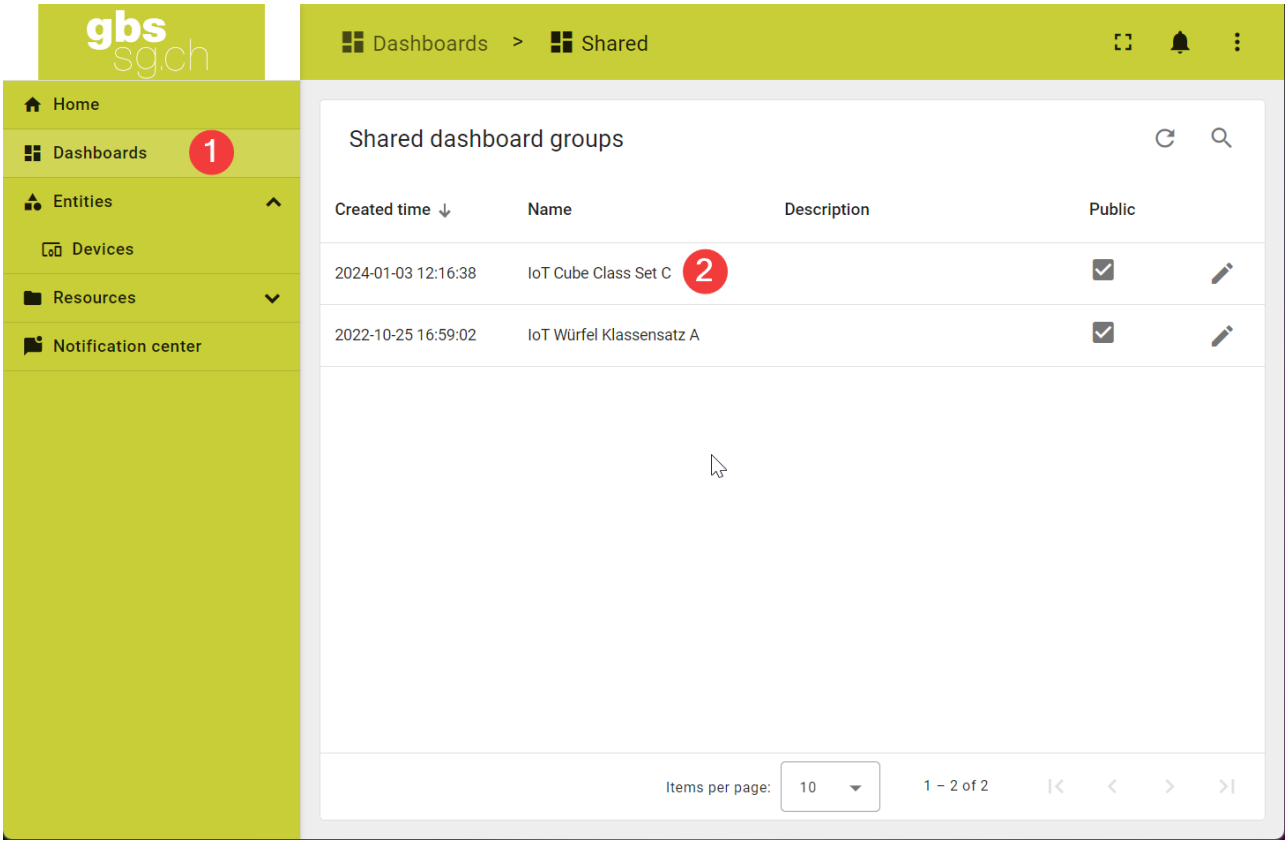
Try to recreate this example. You will need nodes from the following categories. The colours help you to choose the right category.



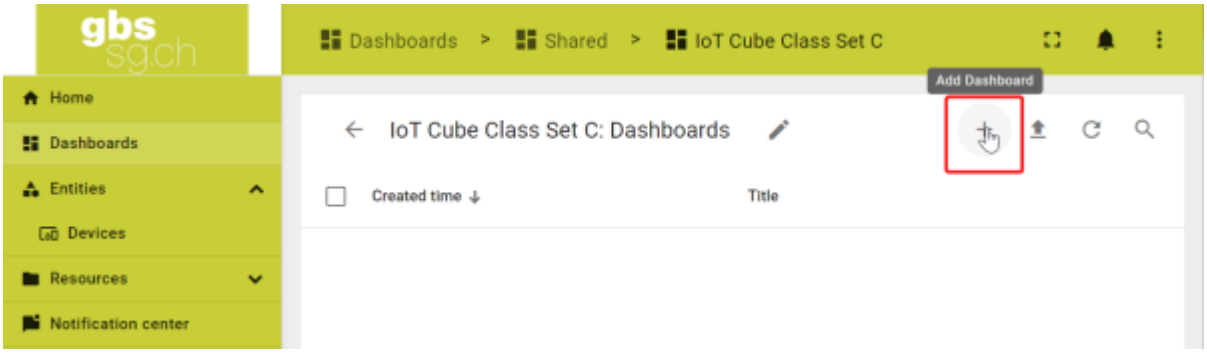
2.4 Integration on things board (example 1)

You now create a dashboard to display the values. Proceed as follows:

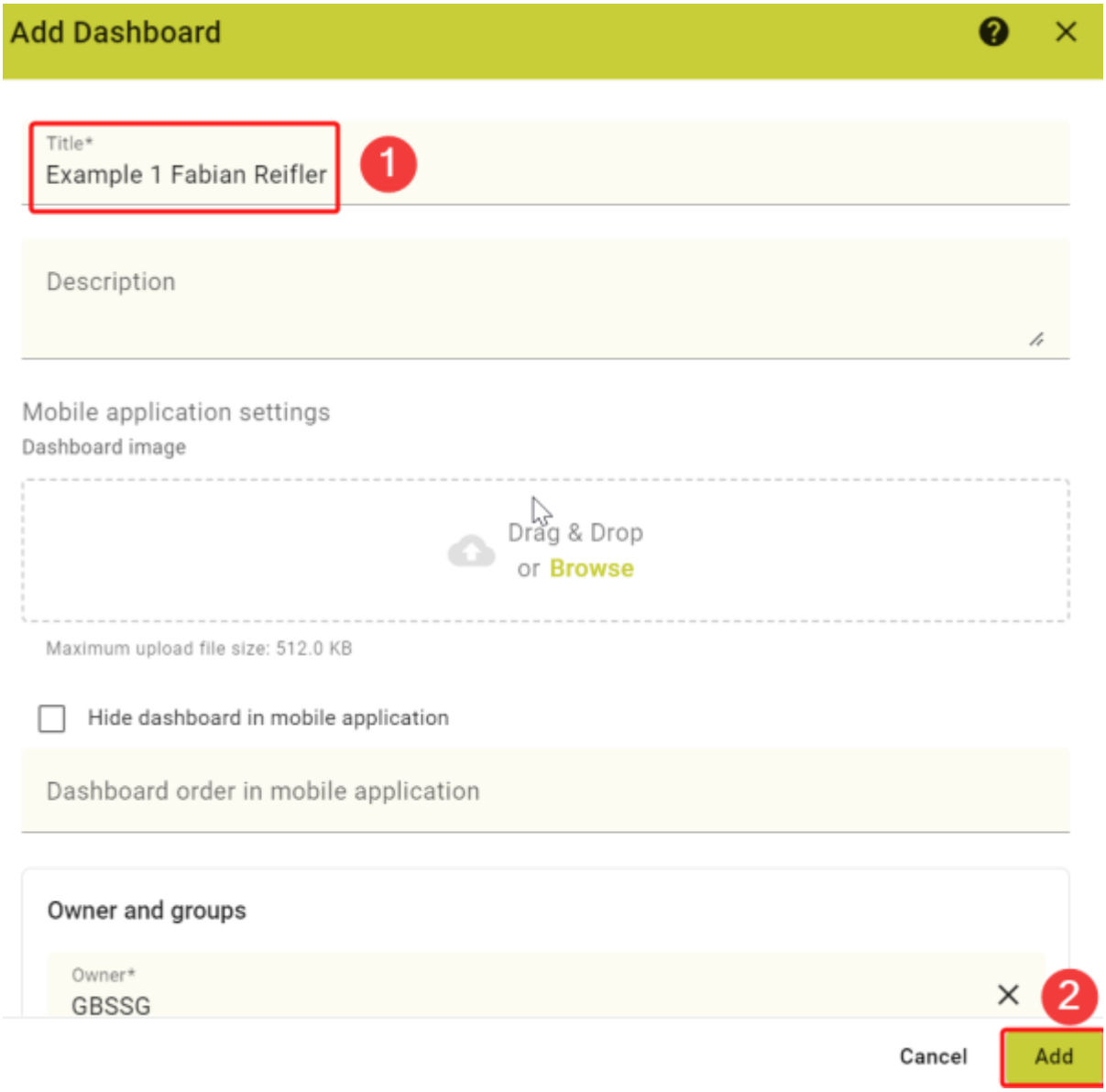
- Click here to open our cloud platform: Clavis Cloud (ThingsBoard)
- Login with the Email **iot@smartfeld.ch**. You will get the password from Fabian Reifler.
- Click on **Dashboards** and than on **IoT Cube Class Set C**



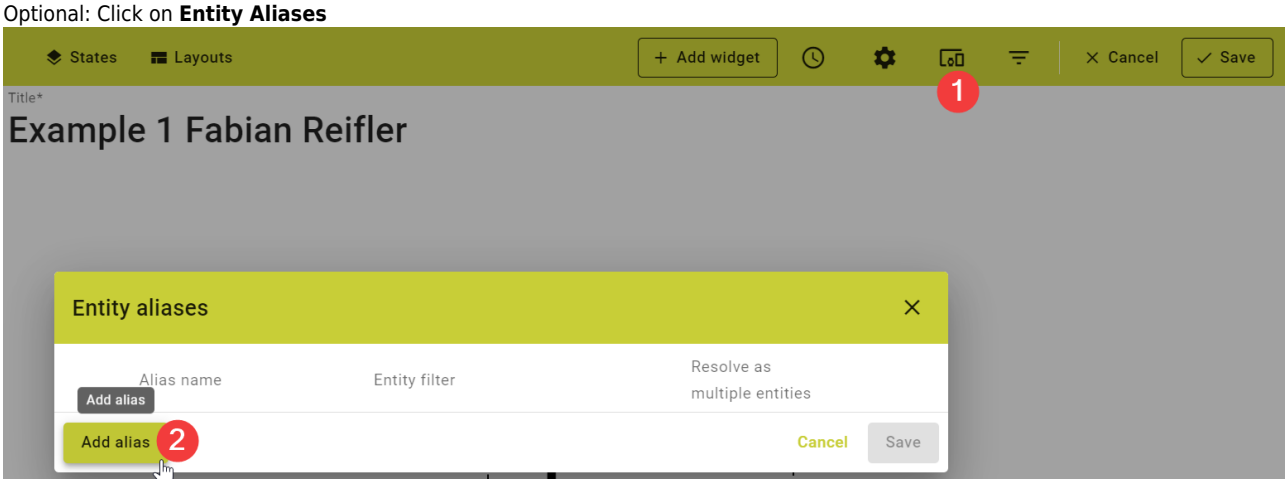
- Click on **Add Dashboard**



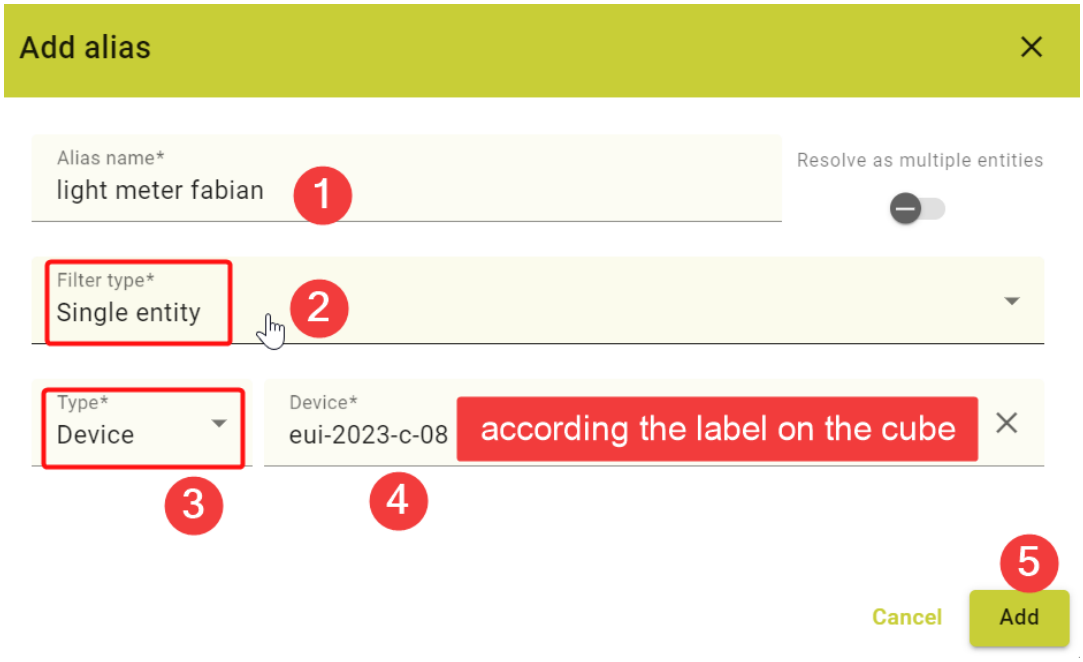
- Give a Title, like **Example 1 Your Name** and click **Add**



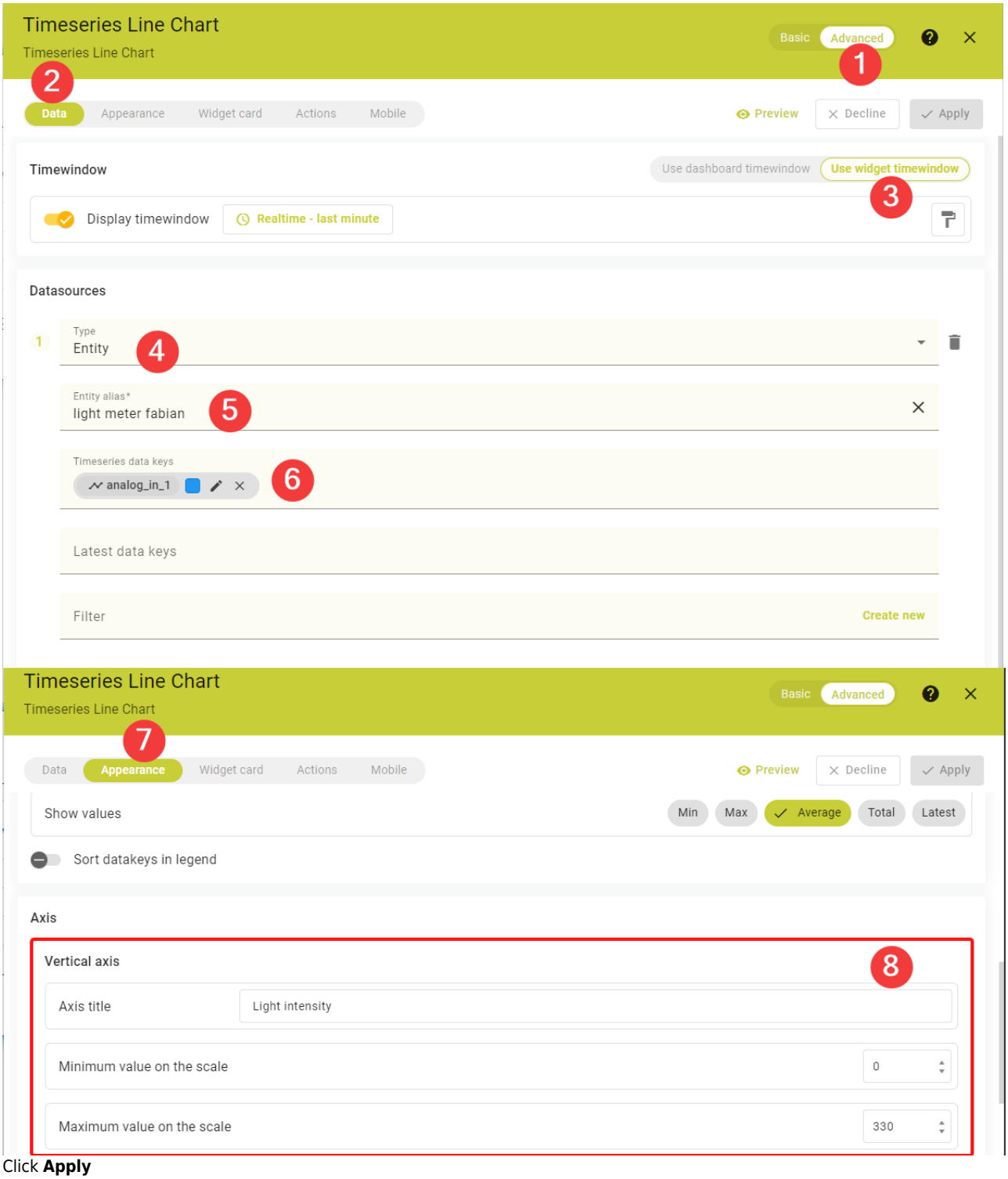
- Optional: Click on **Entity Aliases**



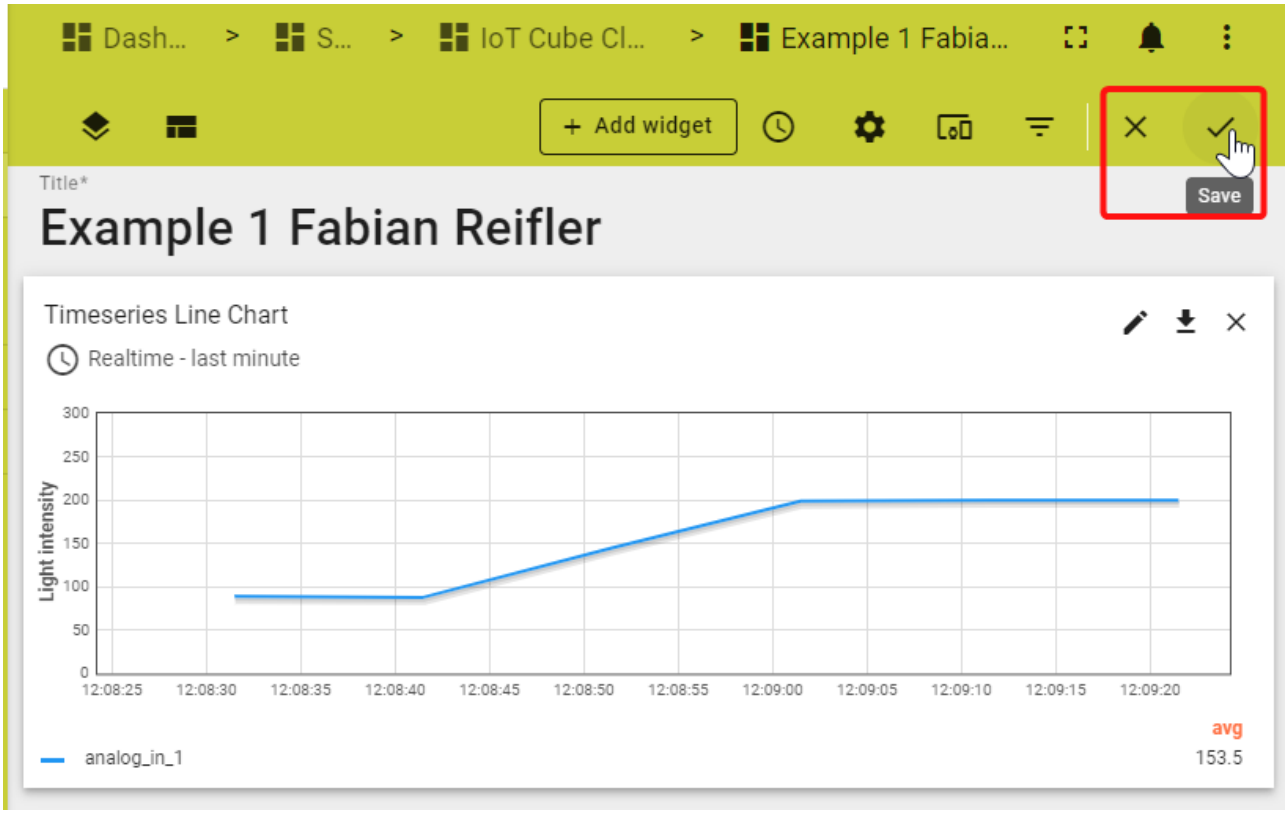
  - Here you can assign a kind of nickname (alias) for your IoT Cube. I will choose „light meter fabian“.
  - Select Filter type: **Single entity**, Type: **Device** and select your device according to the label on the cube. Note: The device must have already sent data, otherwise it will not appear.



- Click on **Add new widget**
- Select widget bundle **Charts**
- Select **Timeseries Line Chart**
- Settings like that:



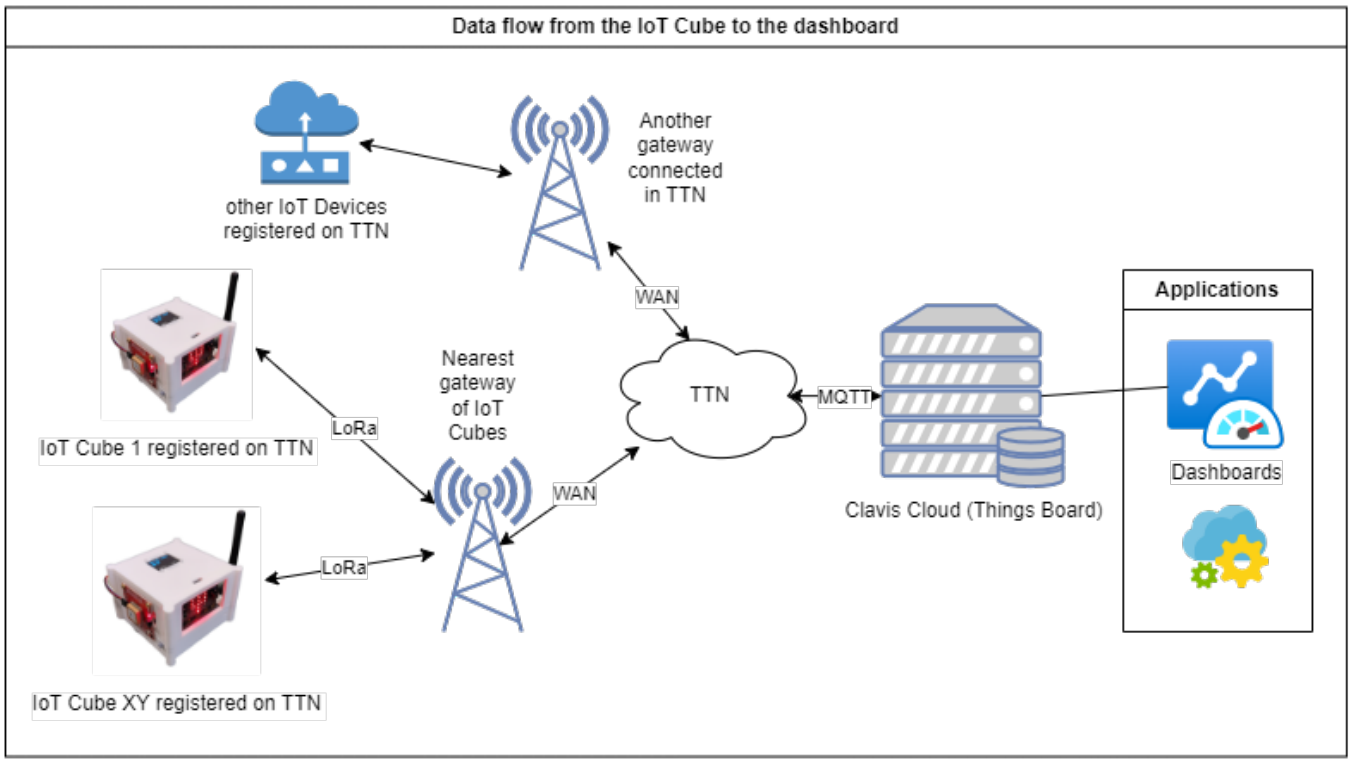
- Click **Apply**
- Don't forget to click **Save**:



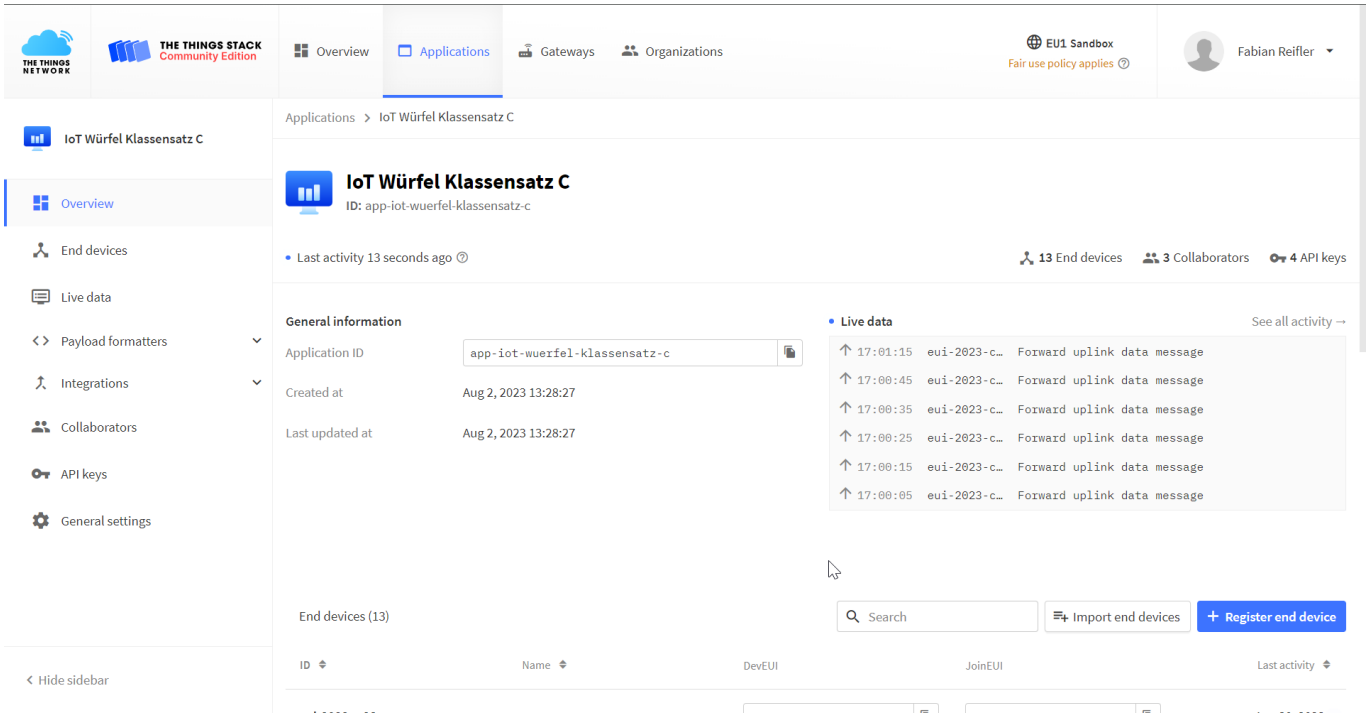
So, this was the short version. But why does the data arrive here so easily?

2.5 Data flow - IoT Cube - TTN - Things Board

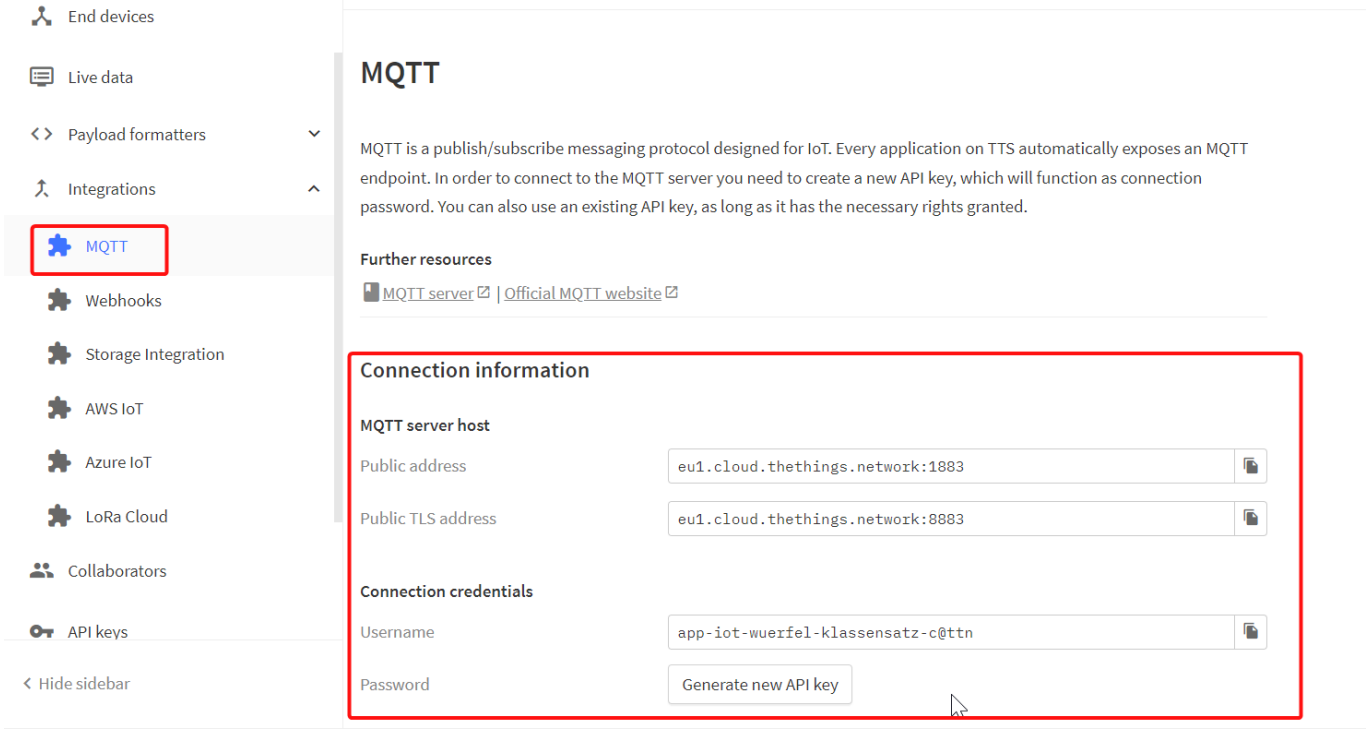
Simplified, the data flow can look like this:



Your measurement data from the IoT Cubes has been transmitted to the Internet, as these devices are already registered with The Things Network (TTN). You can access and view your live data on the TTN Console. If you're interested in this aspect, please visit the TTN Console. If you do not have a TTN account, create one TTN Login and contact Fabian Reifler to be added as a collaborator. The TTN Console looks like that:



TTN [1] serves as a LoRaWAN Network Server, connecting all LoRaWAN gateways and edge devices. It's important to note that data is not permanently stored on TTN but is forwarded. Currently, TTN forwards your measurement data using MQTT [2]:



The data is retrievable via MQTT, which we implement using ClavisCloud. ClavisCloud is a hosted version of the open-source software ThingsBoard. It archives every message and provides functionality to create dashboards for visualizing the data.

3. Example with Slider (example 2)

3.1 Goal (example 2)

Now we want to connect a sliding Potentiometer to the IoT Cube. Whenever the slider is moved, a value is to be transmitted via LoRaWAN. The value can be used for various scenarios, e.g. to dim a remote smart light. The value is also shown on the OLED display.

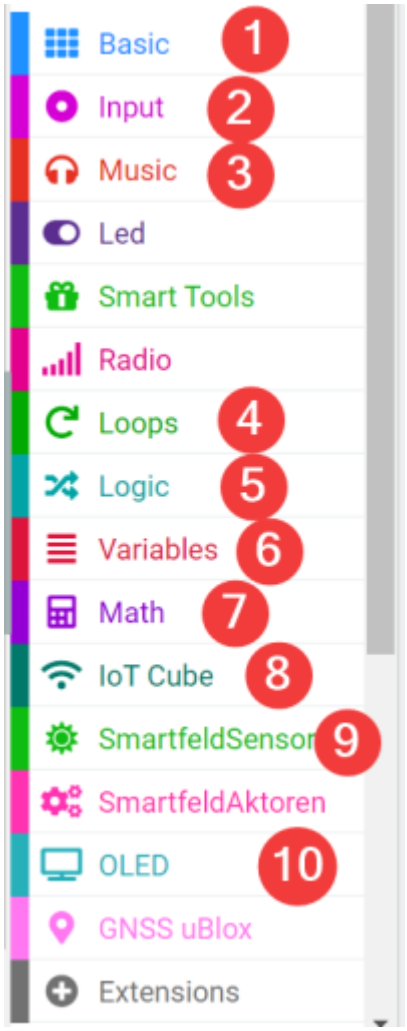
3.2 Hardware (example 2)

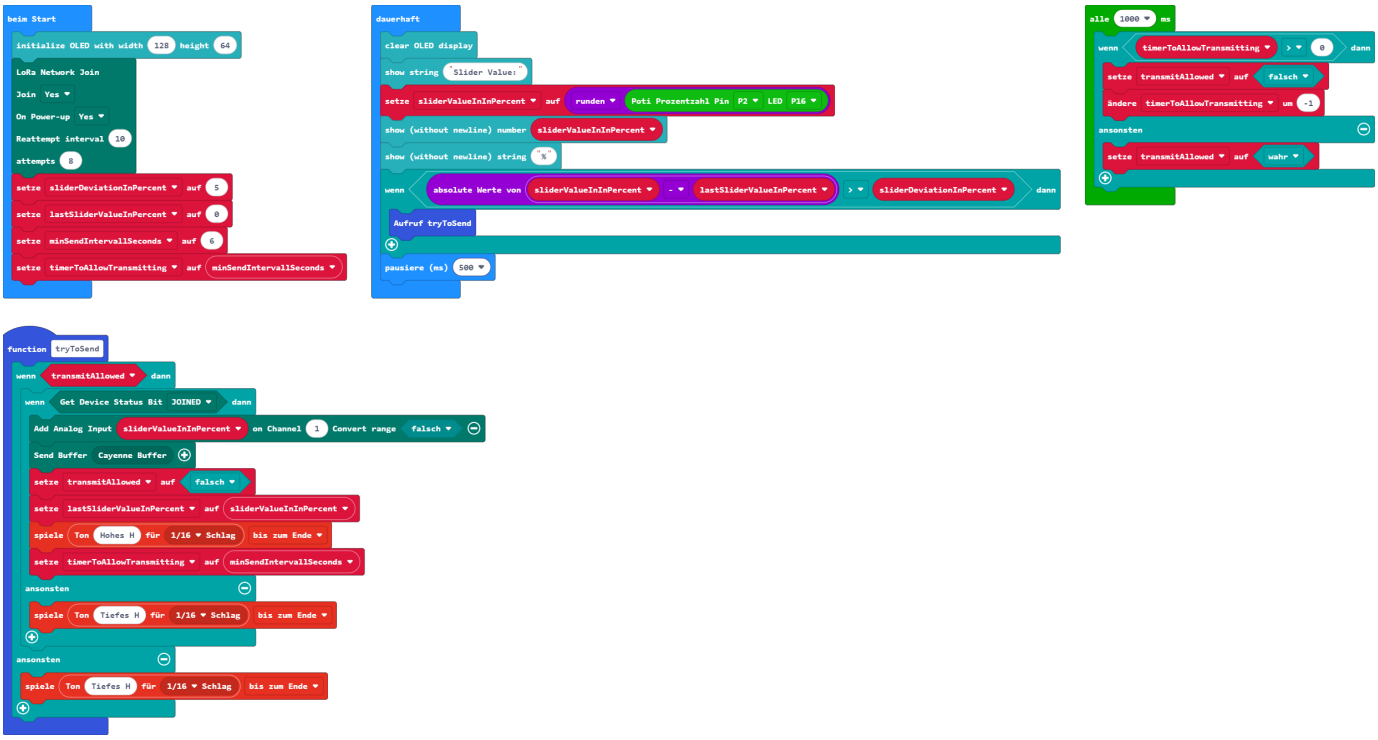
- Connect the display to **J5 (I2C 5V)**
- Connect the **Sliding Potentiometer** to the IoT Cube. You can use **J3 (Pin P2 3.3V)**. Pin P2 can be used to read analog values.
- Connect the Y USB Cable to the IoT Cube
    - the plug marked red on the back of the housing (only power)
    - the other one to the micro:bit on the top of the housing (power + data)
- Connect the USB-A Connector with your computer

Be careful when connecting other sensors to J5, J6 and J7. J5 to J7 have a 5V supply voltage. 3.3V devices can be damaged. Use J3, if you are using analog sensors, like the UV- Sensor in the box.

3.3 Software for the IoT Cube (example 2)

Try to recreate this example. You will need nodes from the following categories. The colours help you to choose the right category.





Explanations

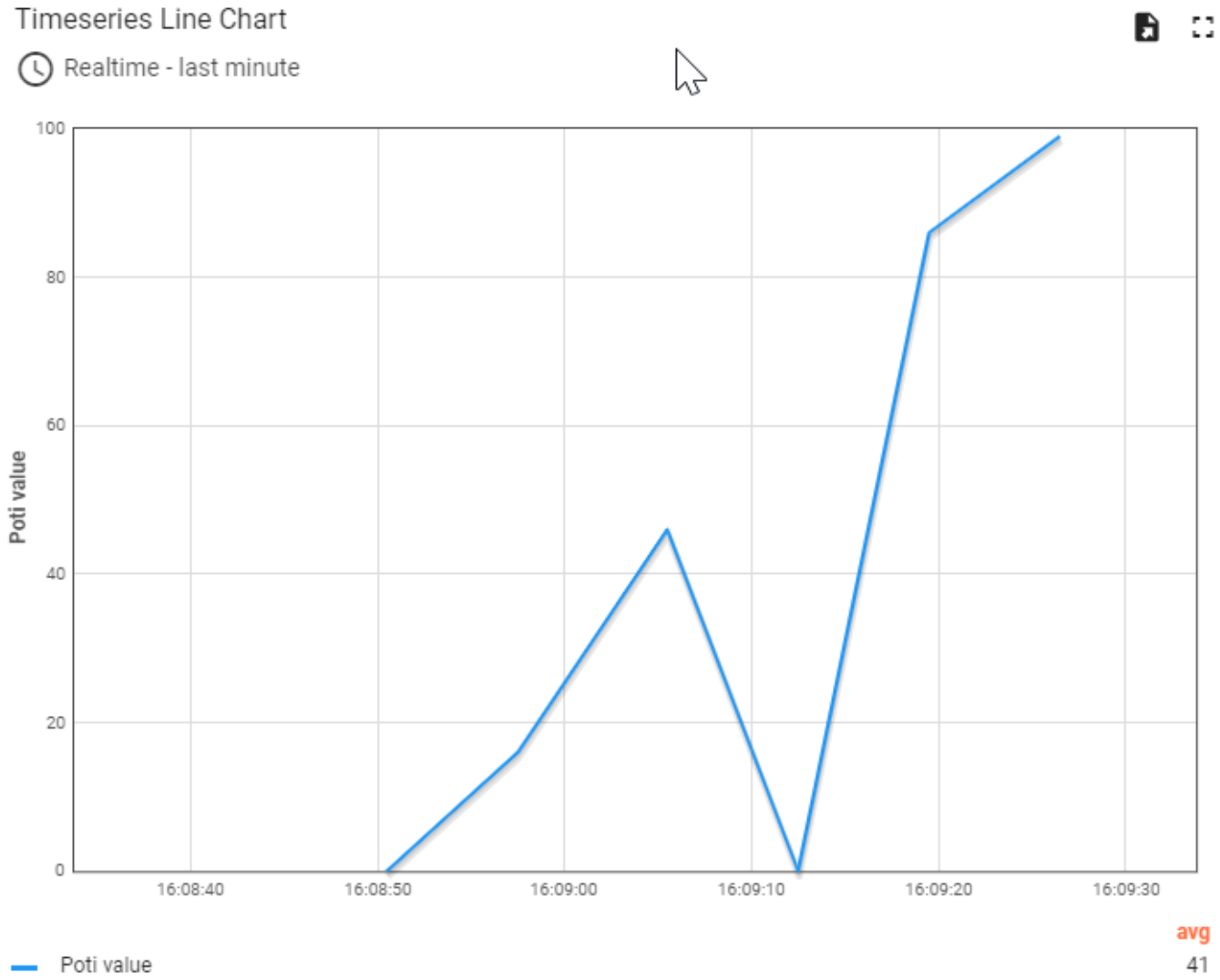
In the current configuration of the IoT Cubes, transmission is only possible approx. every 6 seconds. Shorter intervals are therefore prevented with a simple timer. When moving the slider, the system first checks whether the value has changed. If so, the program attempts to send the data. However, this is only possible if the IoT Cube has joined the network and nothing has been sent in the last six seconds. Compared to the first example, the edge device acts more intelligently here. Data is only sent if something has changed. This saves energy and extends the battery life.

> Intelligent edge devices in IoT (Internet of Things) are crucial components for modern smart systems. These devices have several key features that make them „intelligent“ and efficient for various applications.
>
> - Energy Efficiency: Think about when it is really necessary to send data. Every transmission costs energy.
> - Data Processing Capabilities: Unlike traditional IoT devices that rely solely on cloud servers for data processing, intelligent edge devices have the capability to process data locally. This reduces latency, as data doesn't need to travel to a central server and back, and it can also operate in environments with limited or no connectivity.

Shared Example 2

3.4 Integration on things board (example 2)

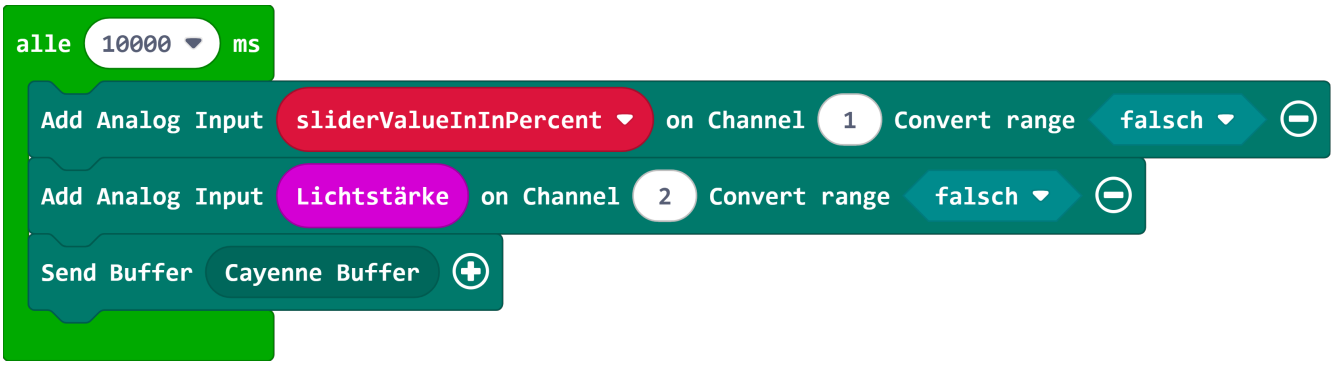
Proceed in the same way as in example 1: Go to the website againClavis Cloud (ThingsBoard) and create a suitable dashboard!



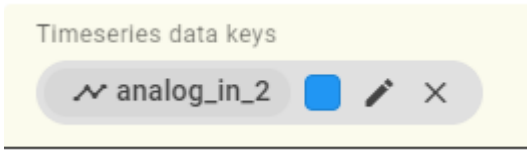
4. Exercise

Try to combine the two examples so that the light measurement and the potentiometer measurement are made on the same IoT cube.

Tipp: You can add more than one sensor value to the buffer and transmit them together. Like this:



If you use Channel 2 instead of Channel 1, the data key looks like the one in the Clavis Cloud (ThingsBoard):



Tags

ue-programming-iot-cube-lora

1)
The Things Network

2)
Message Queuing Telemetry Transport

UE: Introduction with IoT Dollhouses and System Knowledge

Short Description

In a half-day session, students experiment with a smart home model (dollhouse). The edge devices are pre-programmed and provided to ensure a smooth introduction.

Contact Person

Fabian Reifler | Copyright © 2023 GBSSG

Framework Conditions

Framework Conditions	
Time Frame	4 lessons
Target Group (Professions)	Technicians
Requirements	

Goals of the Teaching Unit

The students should get to know initial IoT applications as an introduction to the course and describe their system structure.

Competency-Oriented Learning Objectives

After completing the section Getting to Know Initial IoT Applications with Node-RED, you will be able to...

- describe what Node-RED can be used for.
- create your own dashboards on your smartphone.
- use the MQTT protocol.
- describe some possibilities of IoT.
- explain the message flow in NodeRED.

After studying the content Remote Access with RealVNC, you will be able to...

- accomplish remote access using RealVNC.

Teaching Materials

Here all teaching materials are listed in chronological order.

Links for Students	**Content / Activity**	**Links for Teachers**
Remote Access with RealVNC	How to access the Dollhouses remotely	
Getting to Know Initial IoT Applications with Node-RED	Introduction with Dollhouses	

An MQTT broker must run on the Raspberry Pi, which is in the local network and allows the following user:

- user: psn
- password: posten_node_RED

Topics to subscribe:

- meinHaus/1OG/Temperatur
- meinHaus/1OG/Luftdruck
- meinHaus/1OG/Luftfeuchtigkeit
- meinHaus/1OG/HeizungStatus

- meinHaus/2OG/FensterStatus
- meinHaus/Dach/Solarzelle/Leistung

Topics to publish:

- meinHaus/1OG/Heizung → „on“, „off“, „1“, „0“ possible

NodeRED Flow for Configuring the Wio Terminals

A SD card with the following data is inserted in the Wio Terminal:

```
{
    "SSID": "IoTB",
    "PW": "",
    "ID":"01",
    "mqtt_bootstrap_broker": "172.20.1.51",
    "mqtt_bootstrap_broker_user":"",
    "mqtt_bootstrap_broker_password":"",
    "default_functionality":"solar"
}
```

The WLAN and the MQTT broker can be configured. The functionality can be set to „solar“ or „heatingControl“. „solar“ is used for the Wio Terminal on the 2nd floor, „heatingControl“ for the 1st floor.

The Wio Terminals can be reconfigured via MQTT. The Wio with ID 01 listens to the following topic:

`setup/wioTerminals/configTerminal-01`

The following can be configured:

```
{
    "brokerToConnect": "172.20.1.71",
    "portToConnect": "1883",
    "mqtt_bootstrap_broker_user": "",
    "mqtt_bootstrap_broker_password": "",
    "functionality": "heatingControl",
    "mqtt_prefix": "meinHaus/"
}
```

This NodeRED Flow takes care of switching between two brokers:

wioterminalconfigurationovermqtt.zip

Docker Container

Here is a Docker container containing all necessary flows and dependencies:

node-red-schnpperkurs-v1.0.tar.gz

If the container is downloaded and Docker is installed on the system, the image can be loaded with the following command:

```
docker load < node-red-schnpperkurs-v1.0.tar.gz
```

Help with Docker containers:

Docker Container (am entstehen)

Additionally, pigpiod must be installed on the Raspberry Pi:

```
sudo apt-get install pigpiod
```

In addition, the pigpiod daemon must be started (entry in /etc/rc.local):

```
/usr/bin/pigpiod -n 127.0.0.1 &
```

Miscellaneous

Remote Access with RealVNC

VNC stands for **V**irtual **N**etwork **C**omputing. It allows remote control of computers in a network. It is a widely used tool. VNC communicates over the dedicated port 5900.

Competency-Oriented Learning Objectives

After studying the content Remote Access with RealVNC, you will be able to...

- accomplish remote access using RealVNC.

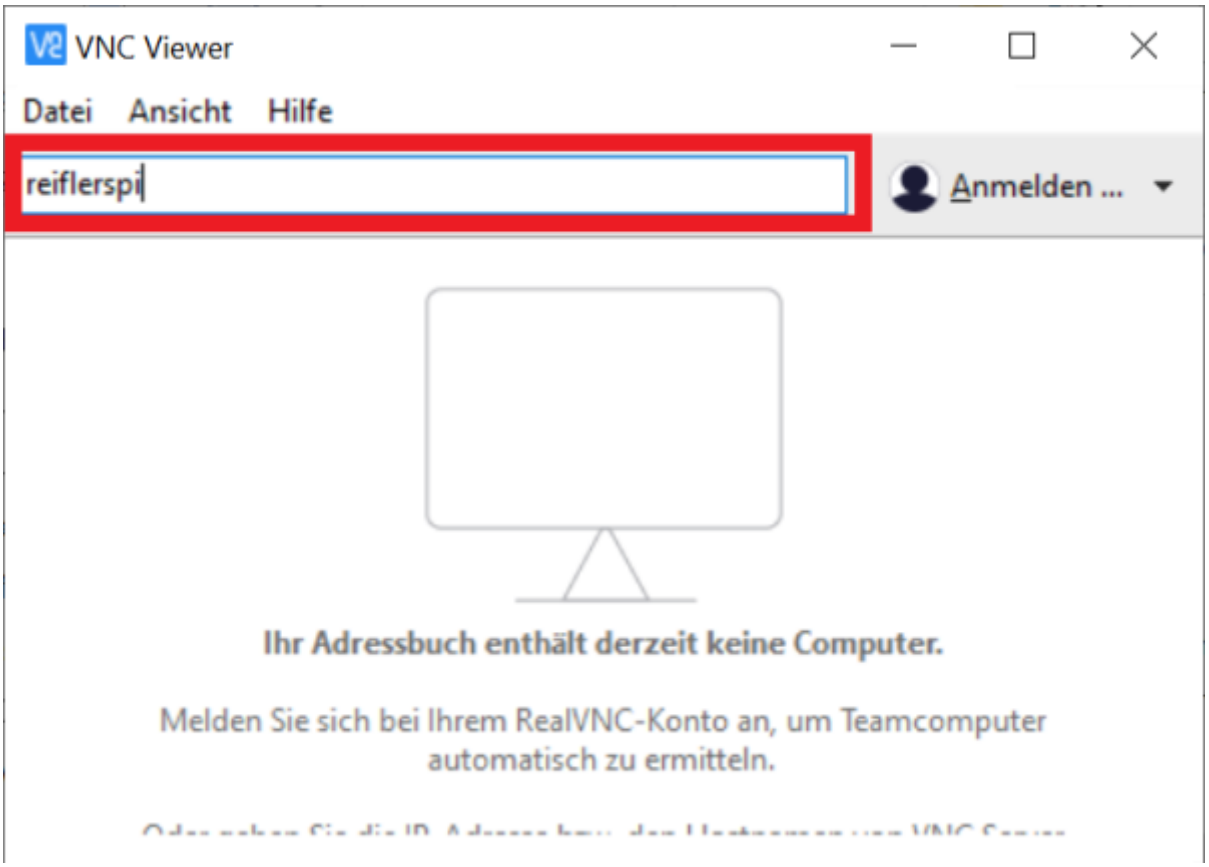
Installing VNC Viewer on the PC

You need the RealVNC Viewer on your PC to enable remote access to the Raspberry Pi via RealVNC. Download it from the following website and install the software:

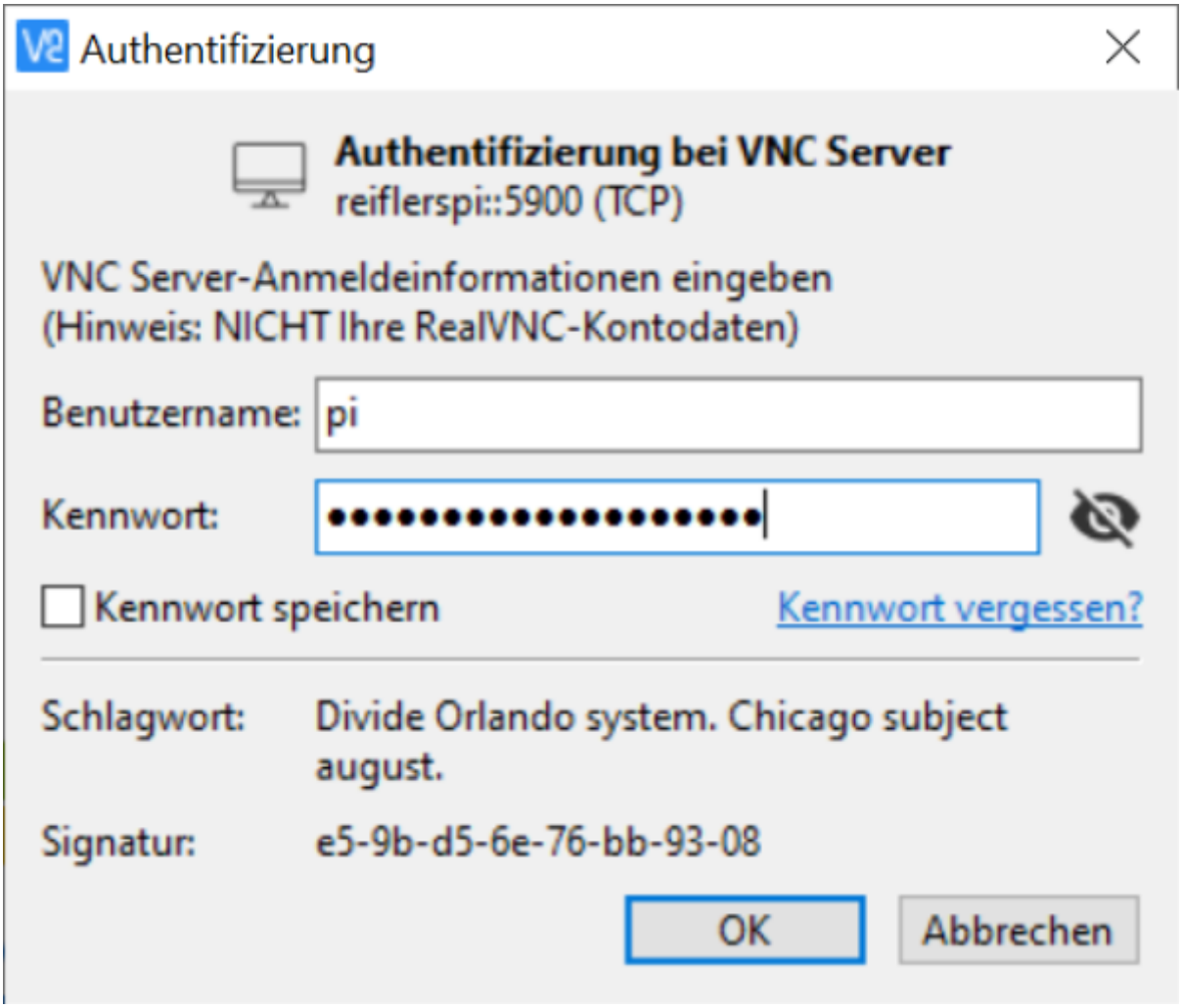
RealVNC Viewer.

Accessing the Raspberry Pi

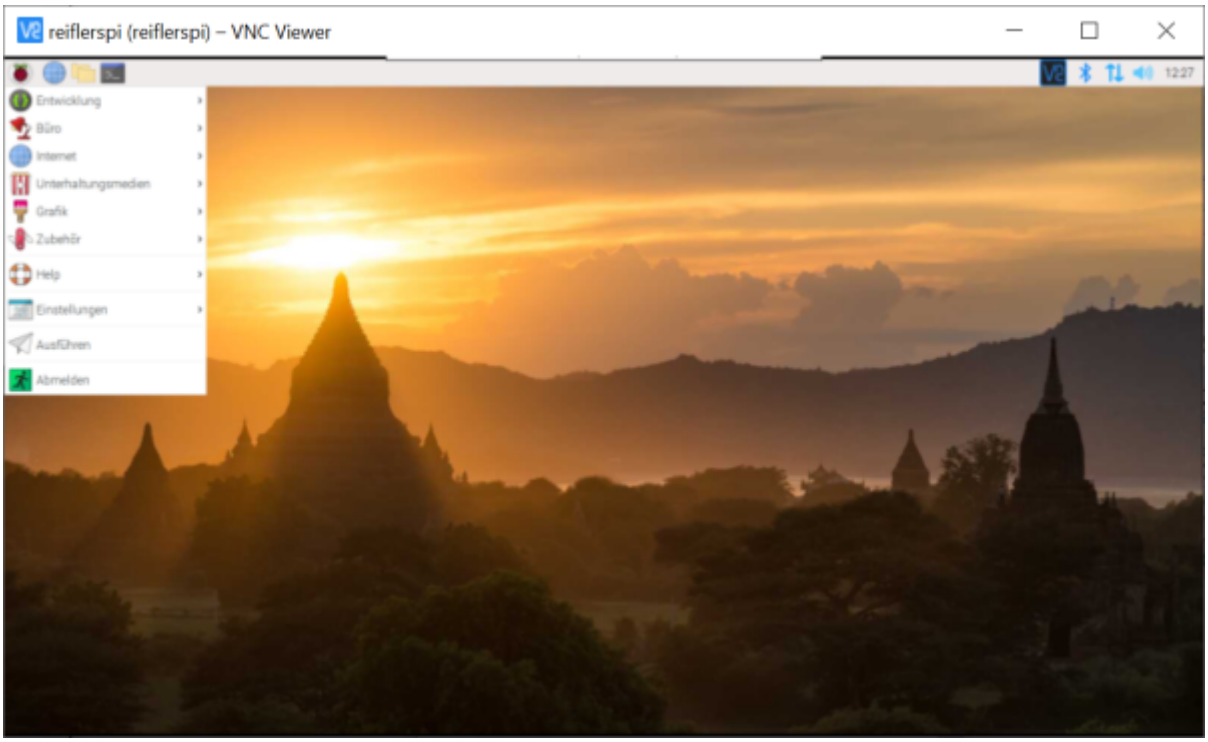
- Start the VNC Viewer
- Enter the host name of the Raspberry Pi (IP also works). Look at the label on the Raspberry Pi case. There you can see the host name, for example: koffer-1a.



- Enter the user on the Raspberry Pi, User: pi
- Enter the password (ask your teacher)

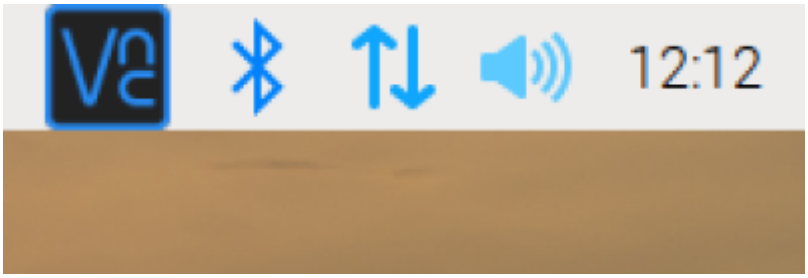


- Voilà, now you can comfortably control the Raspi remotely over the network:



Additional Information: Installing VNC Server on the Raspberry Pi (already done)

A VNC server from RealVNC is usually already installed on the Pi. If you see the VNC icon in the top right corner, it's already there:



If not, follow these steps in the console:

~ $

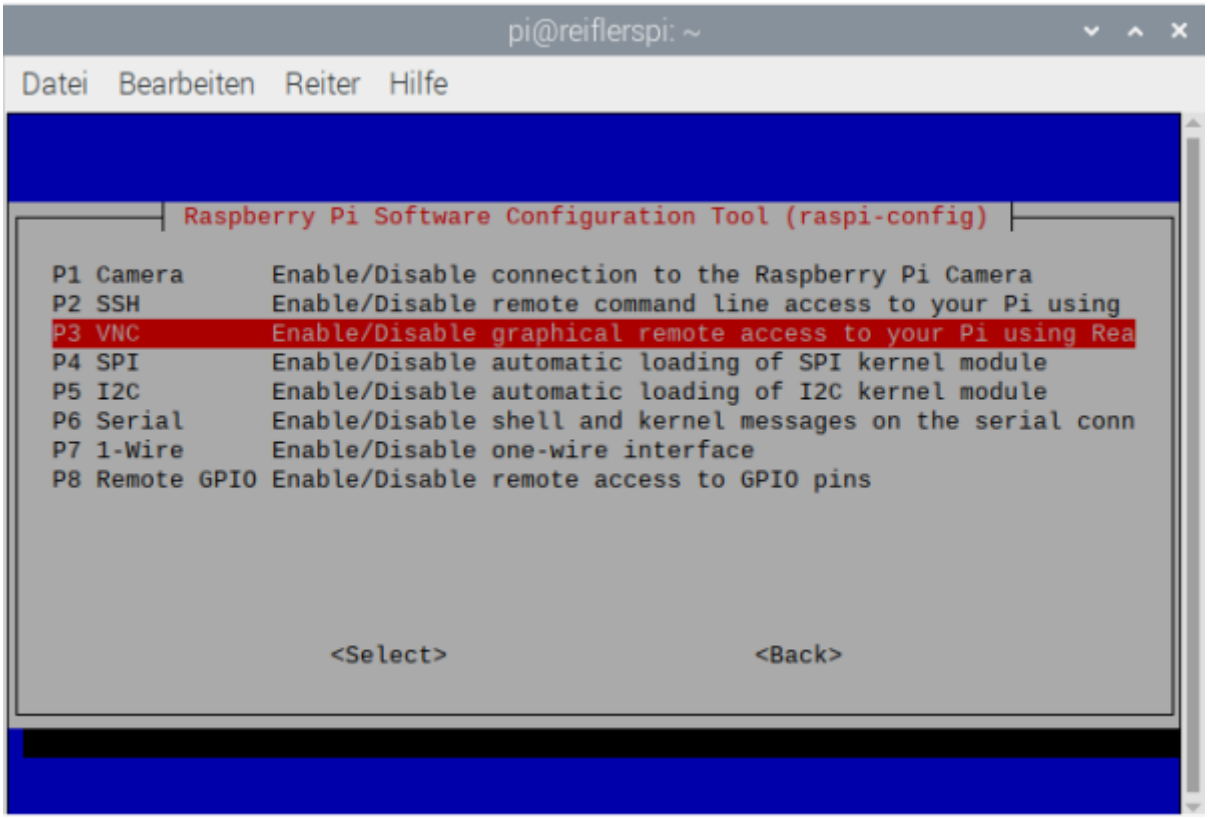
```
sudo apt-get update
sudo apt-get install realvnc-vnc-server
```

An important setting is hidden in the raspi-config menu:

~ $

```
sudo raspi-config
```

Navigate there to „Interfacing Options“ and turn on VNC:



Tags

If the page is used for a teaching unit, please leave a tag here. The tags are used for the automatic compilation of learning objectives.

ue-node-red-introduction

Getting to Know Initial IoT Applications with Node-RED

Here you have the opportunity to get to know the open-source software Node-RED to network some sensors and actuators. Node-RED runs in your dollhouse on a single-board computer named Raspberry Pi, but could also be operated on a PC or in the cloud.

After Node-RED is started on the Raspberry Pi, a web server on the single-board computer is accessible. The web server provides a graphical programming interface, which can be opened with a web browser.

Competency-Oriented Learning Objectives

After completing the section Getting to Know Initial IoT Applications with Node-RED, you will be able to...

- describe what Node-RED can be used for.
- create your own dashboards on your smartphone.
- use the MQTT protocol.
- describe some possibilities of IoT.
- explain the message flow in NodeRED.

1. Starting NodeRED

- Use the Raspberry Pi, either via remote access (RealVNC) or directly with screen and keyboard. For Remote- Access ask your teacher.
- Close the web browser on the Raspberry Pi if it is still open.
- On the **Desktop** of the Raspberry Pi, we have created an executable script file named **start.sh** for you. By **double-clicking** on it, commands are executed to start NodeRED on the Raspberry Pi and load some example flows. After double-clicking, please wait **about 5 seconds**, then the web browser with the loaded Node-RED frontend should appear.
- For your information: You can't do much harm in this environment, clicking on **start.sh** again will restore the original state. Be aware: this process will discard your changes.

2. Access to the NodeRED Webinterface with your computer

If you are in the same WLAN (SSID: IoTB) as the dollhouses, the NodeRED web interface can also be accessed from your computer. The url looks like that, but **change the Xy** according to the label on the raspberry pi:

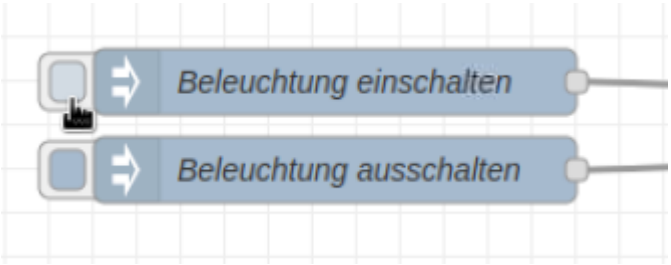
`http://koffer-Xy:1880/`

Control via your own browser offers better performance than operating the web interface via RealVNC.

3. Messages in NodeRED

3.1 Example 1: Lighting

In the first tab, I have prepared an example flow for you to turn on or off lighting in the dollhouse. Click alternately on the two buttons for a test. Report to the course instructor if the light does not turn on in the dollhouse:



Node-Red is based on the object-oriented programming language JavaScript. Therefore, the messages exchanged between the nodes are JavaScript objects. The message flow is from left to right.

The objects that are forwarded over the connections, for example, have the following properties:

```
var msg = {payload: "Some data", topic: "Topic of the message", _msgid: "99dbeb61.db6b28"};
```

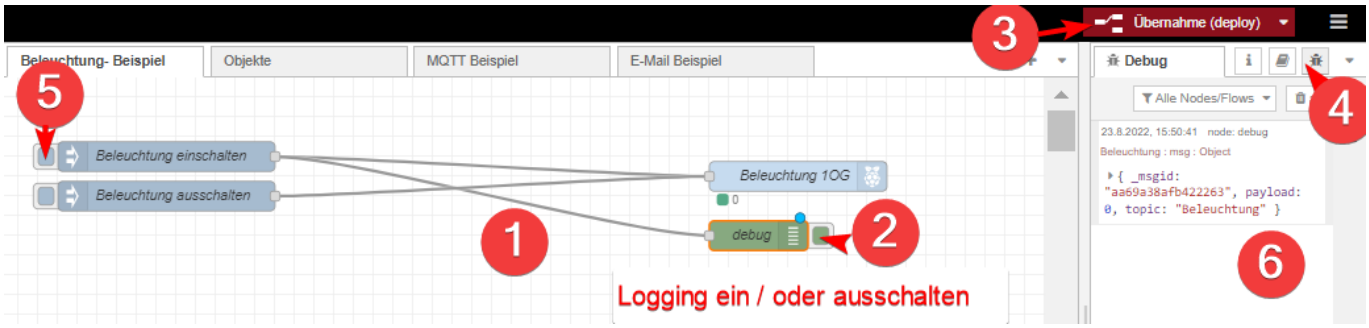
The properties mean the following:

Properties of the msg object	
payload	stores the payload of a message to be transmitted and can take any conceivable form and complexity
topic	stores the overarching topic of a message, so that filtering / classification is possible at the receiver
_msgid	contains an internal ID for identifying the message

To see during development what messages are exchanged between the nodes, the green debug nodes can be used.

Assignment (numbers correspond to the screenshot below):

1. Connect the left two nodes to the debug node.
2. Make sure that logging is turned on (green button must be extended)
3. Press „Übernahme (deploy)“
4. Click on the bug in the sidebar on the right
5. Trigger a message by pressing the upper or lower switch
6. Observe the messages in the debug window



Here are some remarks on this exercise:

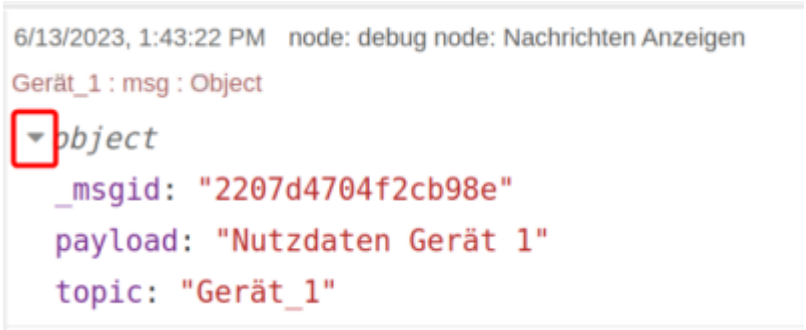
- Take a look at the configuration of all nodes by double-clicking:
  - The two blue *inject*- nodes on the left generate a message when pressed. The message content is 0 for the upper button and 1 for the lower button.
  - The light blue *GPIO*- node on the right establishes a connection to the digital inputs and outputs (GPIOs) of the Raspberry Pi. A relay is connected to pin 40, which turns on the lighting in the dollhouse.

3.2 Example 2: Objects

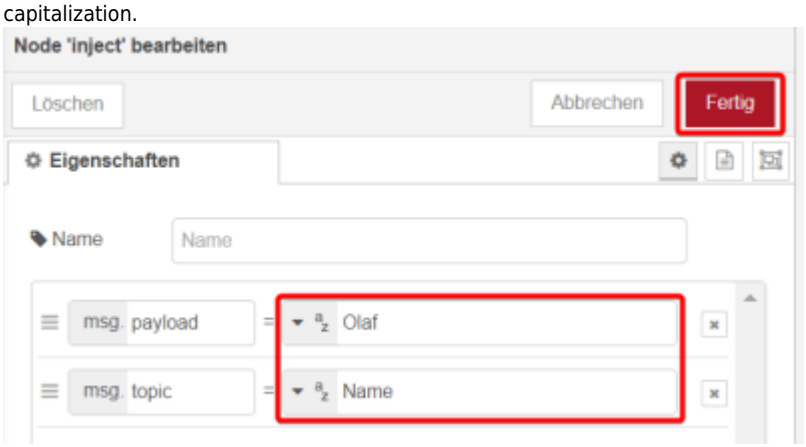
Now click on the „Objects“ tab.

Complete and examine the flow in the following steps:

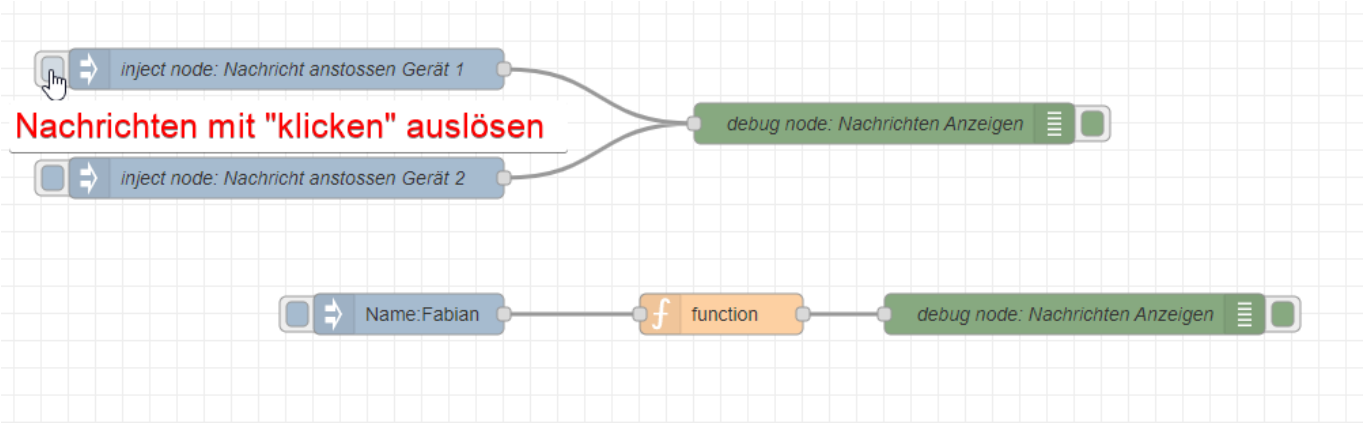
1. Connect the two inject nodes to the debug node.
2. Click on **Übernahme (deploy)**
3. Click on the button of the inject node on the left to trigger a message.
4. Observe what happens in the debug window. You can expand the message with the small arrow:



5. Take a look at how the nodes are configured by double-clicking on them.
6. Drag a new inject node into the flow area (found under „General“ on the left). Double-click to access the configuration.
7. Next to *msg.payload*, enter your name. You should set the data type to `string`. Next to *msg.topic*, enter *Name*. Pay attention to capitalization.



8. Connect the inject node to the existing *function* node.
9. Click on **Übernahme (deploy)**
10. Observe again what happens in the debug window when you press the button of the new inject node. If you did it right, the following should be displayed: „Well done! So your name is …!“
11. Take a look with a double-click on the „function“ node to see what is being done there. In Node-RED, it is thus possible to intervene at any point with your own Java Script functions, for example, to change messages.



4. The MQTT Protocol

NodeRED supports many network protocols, such as UDP, TCP, HTTP, and **MQTT**. The MQTT protocol is very popular in IoT devices because it is lightweight and easy to use. For example, the MQTT protocol allows controlling actuators over the Internet or reading the values of remote sensors.

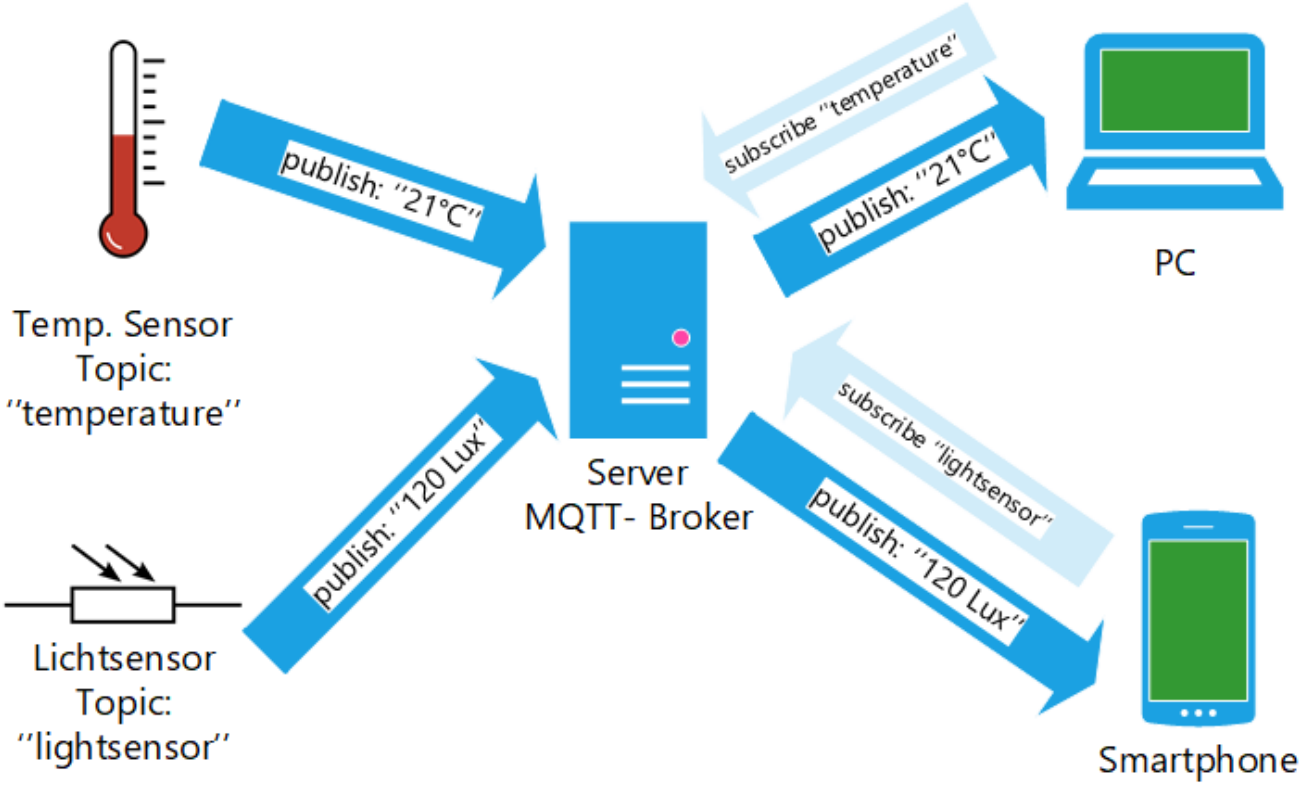
4.1 Publish / Subscribe

Unlike HTTP with its request/response method, MQTT implements a publish/subscribe architecture. In HTTP, a point-to-point connection with request and response is common. The transmission in MQTT works differently: devices that have something to report simply send their data to the broker (central server in the IoT system), that is, they publish their messages. The broker forwards these messages to other devices that have subscribed to these messages.

Devices that can be networked with MQTT include:

- Network-capable microcontroller board (e.g. Wio Terminal or ESP32)
- PC (Windows)
- MAC (OS X)
- Raspberry Pi (Linux)
- Smartphone (Android or iOS)
- Generally: Network-capable, intelligent units in which the MQTT protocol is implemented

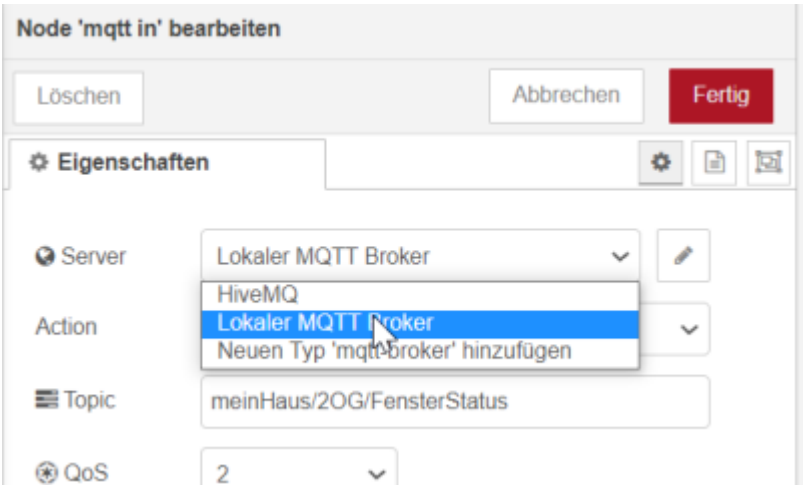
All devices connected to an MQTT broker are also referred to as **MQTT clients**. An MQTT network thus consists of several MQTT clients and one **MQTT broker**.



4.2 Example in the Local Network

It is not always necessary for the data to be accessible on the Internet. For example, if you only need sensor signals within your own home, networking in the local network is sufficient.

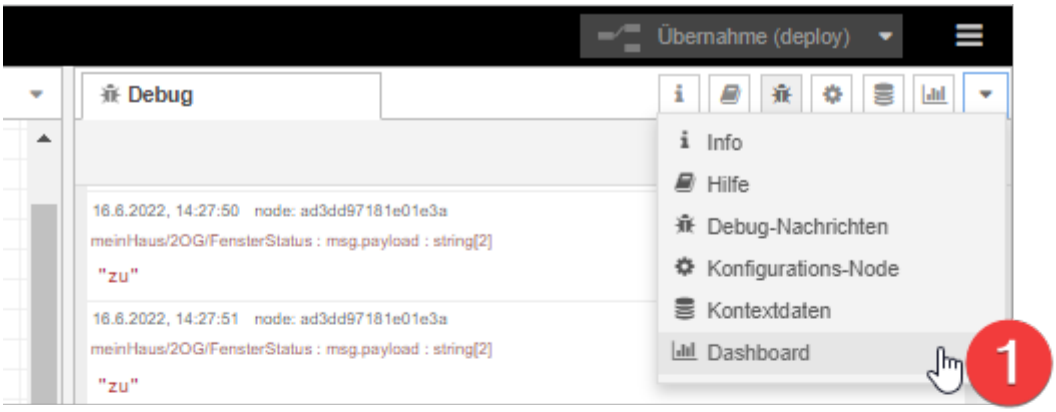
A local MQTT broker runs in the dollhouse. This is only accessible within the same network (in this case, the IoTB WLAN).

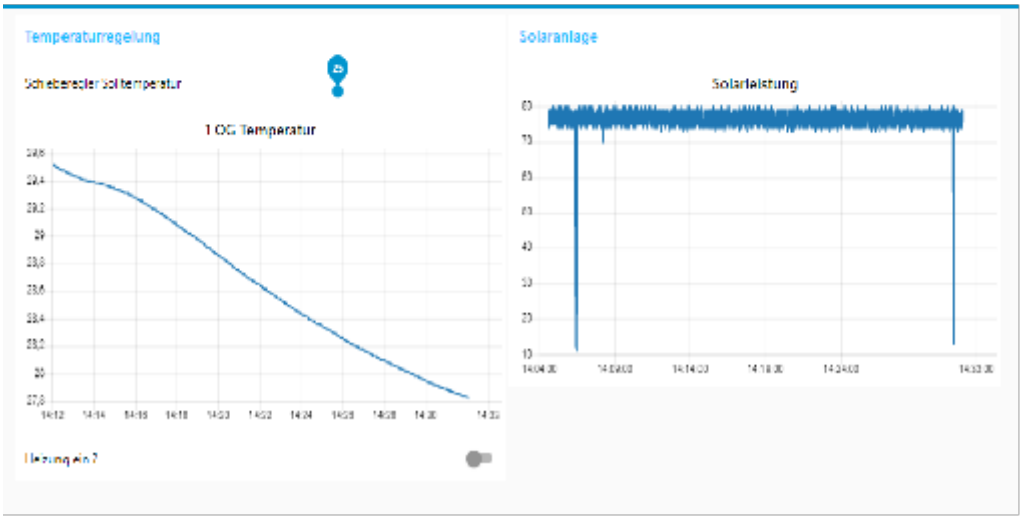


4.2.1 Investigating the Flow

- Click on the „MQTT Beispiel“ (MQTT Example) tab
- Investigate with the help of the green debug nodes:

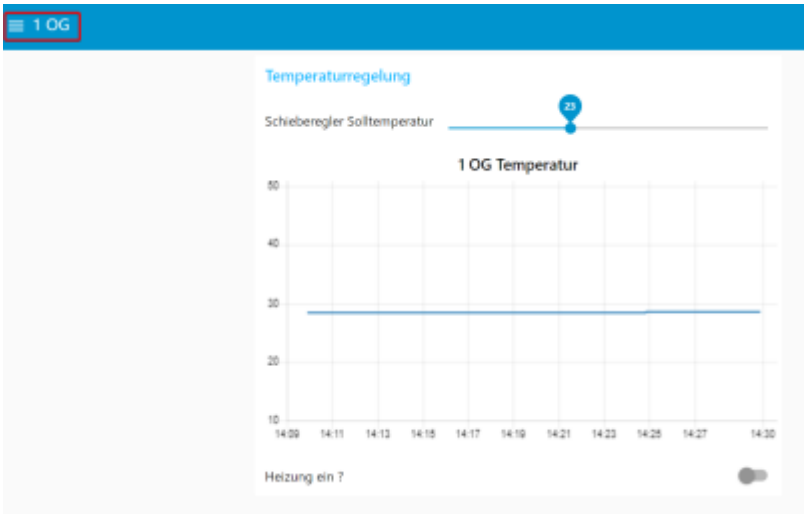
    - What data is sent from the window of the dollhouse?
    - Do the values of the solar cell power change when you cover them with your hand?
    - What temperature does the sensor measure on the 1st floor of the dollhouse?
- Some blue nodes are used to create a user interface (dashboard). Open the dashboard as follows:
    - Click the small arrow ∇ in the top right corner
    - Click on Dashboard
    - Click on the small arrow symbol in the top right corner





4.2.2 Heating Control Task

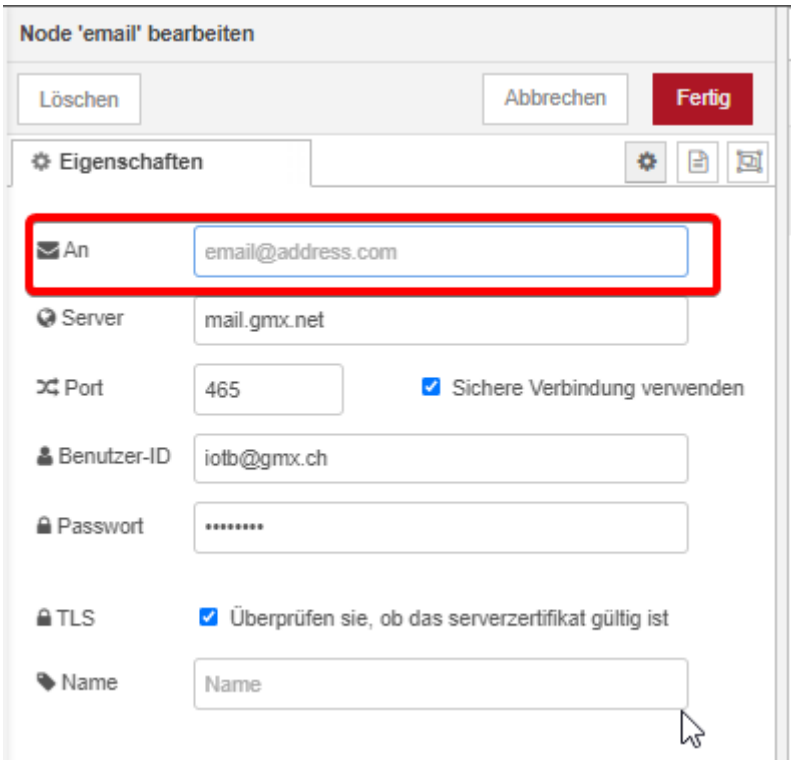
- The flow **MQTT Beispiel** (MQTT Example) is not yet completely wired. Try to implement a two-point control for the temperature by correctly connecting the nodes. The following is still missing:
    - The actual temperature is not yet evaluated by the controller
    - The controller is not yet sending control commands to the heating in the dollhouse.
- Test the controller by changing the desired temperature on the dashboard. You need some patience, the control loop is rather sluggish.



5. Window Monitoring

Barbie and Ken forgot to close the window on the 2nd floor last week and left for the weekend. A storm completely destroyed the room. Let's make sure this doesn't happen again! In the following example, we want to send an email as soon as the window remains open for a certain amount of time.

- Switch to the „Email Example“ tab
- For testing purposes, we want to set the trigger time to 5 seconds (in practice, several hours would be more practical). Configure the trigger node accordingly.
- Configure the email node so that the warning email lands in your account.



- Finally, click on **Übernahme (deploy)**
- Test whether an email arrives if the window remains open longer than the set time.
- Try to figure out how you can change the content of the email.

6. Clever Boiler Control

6.1 Switching on at High Solar Power

The dollhouse is equipped with a **photovoltaic system**. The current power is measured and can be subscribed to in the local network under the topic **`meinHaus/Dach/Solarzelle/Leistung`**.

Electric energy is notoriously difficult to store. This leads to many homeowners with photovoltaic systems having to sell their surplus electricity very cheaply to their grid operator. However, there may be devices in the house that can serve as energy storage:

- Electric car in the garage
- Electric boiler (for hot water)
- Freezers

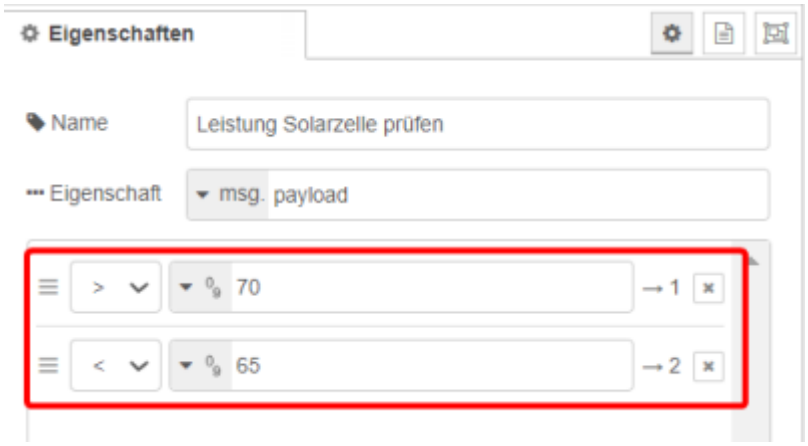
Or devices that can be switched on flexibly in terms of time:

- Battery-operated devices
- Washing machine
- Tumbler

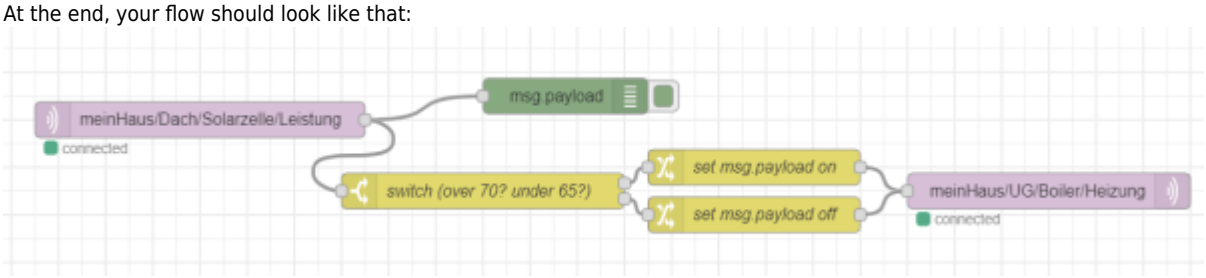
We want to turn on the electric boiler in the dollhouse when more than 70% power is available from the solar cells. Of course, in practice, the current consumption in the house would also have to be measured, but we simplify this in this school example.

Proceed as follows:

1. Take a look at the boiler dashboard (Tab Basement)
    1. Manually switch the boiler heating on and off (click on the lightning bolt).
    2. Turn the shower on and off.
    3. Observe what happens.
2. In Node-RED, go to the „Boiler Charging“ tab
3. Drag an `mqtt in` node into the flow area and configure it:
    1. Under Server, select Local MQTT Broker.
    2. Subscribe to the topic `meinHaus/Dach/Solarzelle/Leistung`
4. Use a debug node to examine what comes from the solar cell.
5. Drag a `switch` node into the flow area.
    1. Double-click on it to get to its configuration menu
    2. Forward the message from the solar cells to output 1 if the solar cell power exceeds 70%. Note that you must define the comparison value as a `number` (default is `string`) and specify it without '%'.
    3. Forward the message from the solar cells to output 2 if the solar cell power falls below 65%. Note that you must define the comparison value as a `number` (default is `string`) and specify it without '%'.



6. Drag two `change` nodes into the flow area, connecting them to output 1 and output 2 of the `switch` node, respectively. Change the payload of the current message to „on“ or „off,“ whichever makes sense to you.
7. Check with a debug node whether the logic works.
8. Now drag an `mqtt-out` node into the flow area.
9. Send „on“ or „off“ via MQTT to the boiler with the topic `meinHaus/UG/Boiler/Heizung`. Choose the local MQTT Broker for this.
10. Test with a flashlight and by covering the solar cell to see if anything happens at the boiler.
11. At the end, your flow should look like that:



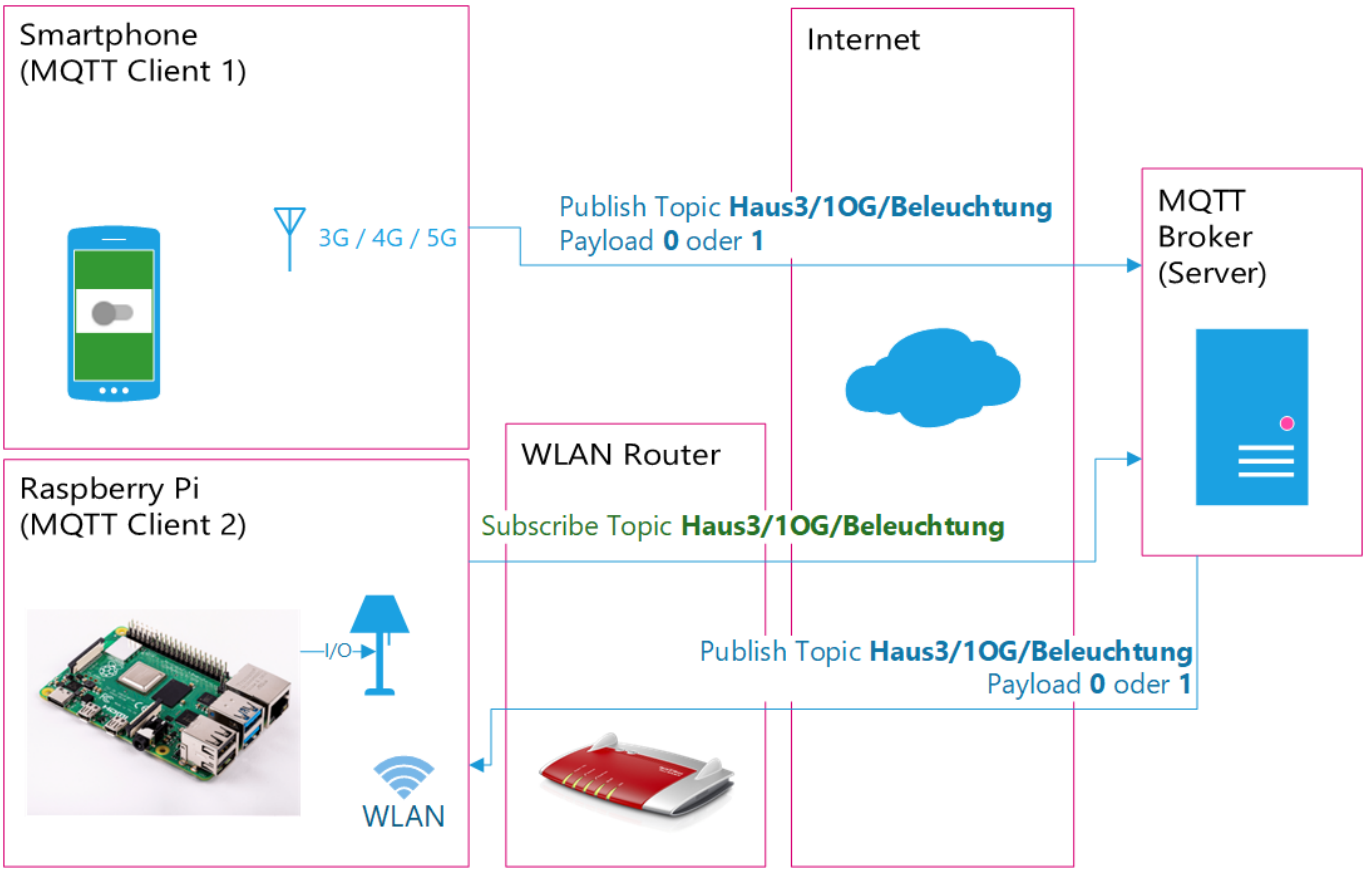
7. Out into the World!

So far, we have only exchanged data between different devices within our local network (our own WLAN). Now we want to make it possible to access the dollhouse from anywhere in the world.

7.1 Actuator Switching via MQTT Broker on the Internet

7.1.1 Overview

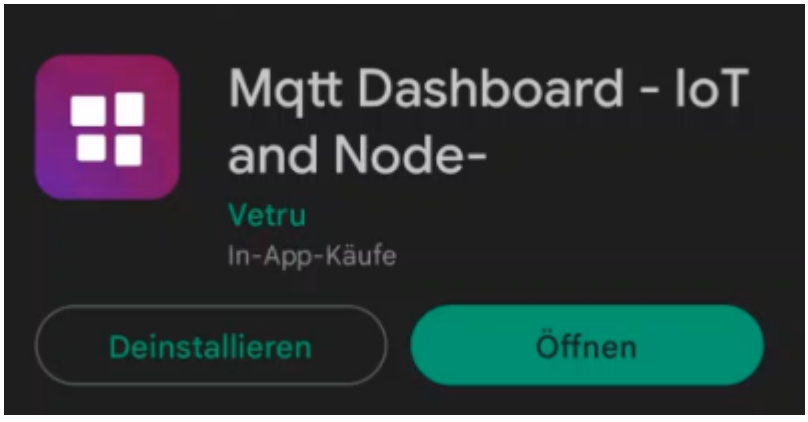
With a smartphone, you should be able to switch the light (or another actuator) on or off in the dollhouse. This endeavor could be simplified via MQTT as follows:



Implementation on the Smartphone (MQTT Client 1)

We now want to create a small app with a single switch to turn the light on or off. There are many MQTT client apps available for installation on a smartphone. Install the following app if you have **Android**:

https://play.google.com/store/apps/details?id=com.app.vetru.mqttdashboard



Unfortunately, I haven't found a comparable app for iOS that is free. For testing purposes, EasyMQTT is sufficient. EasyMQTT is severely limited in the free version. Unfortunately, without paying for the full version (a one-time fee of 6 CHF), it's not possible to create buttons, graphs, and the like.

So, install this app on iOS (the free version is sufficient for testing, but those who want more will have to pay):

https://apps.apple.com/de/app/easymqtt/id1523099606?nolink

7.1.1.1 MQTT Broker

In the app, you need the access data of an MQTT broker. Scan the following QR code with your smartphone so that you can conveniently copy out the access data (get the password from the course instructor):

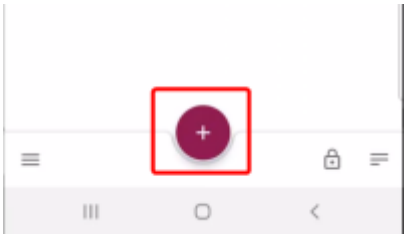
After opening the MQTT Dashboard app, enter the access data.

Proceed as follows on Android:

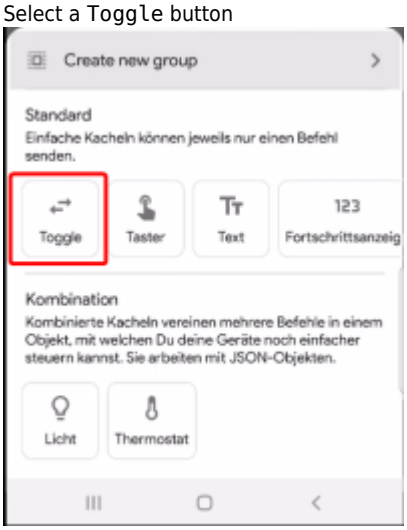
1. Open the app you just installed
2. Click on `ADD FIRST BROKER`
3. Fill in all the empty fields with the access data:
    1. Broker Name: `HiveMQ`
    2. Address: `ssl://schnupperkurs-mqtt-broker.s2.eu.hivemq.cloud`
    3. Port: `8883`
    4. Unique Client ID: leave as it is
    5. Click the checkbox `Activate broker protection` or `Brokerschutz` (German)
        1. Username: `iot-schnupperkurs`
        2. Password: see access data
    6. Click to use SSL connection
        1. Click the checkbox `Use SSL connection`
    7. Click save at the bottom

7.1.1.2 Create Dashboard on the Smartphone

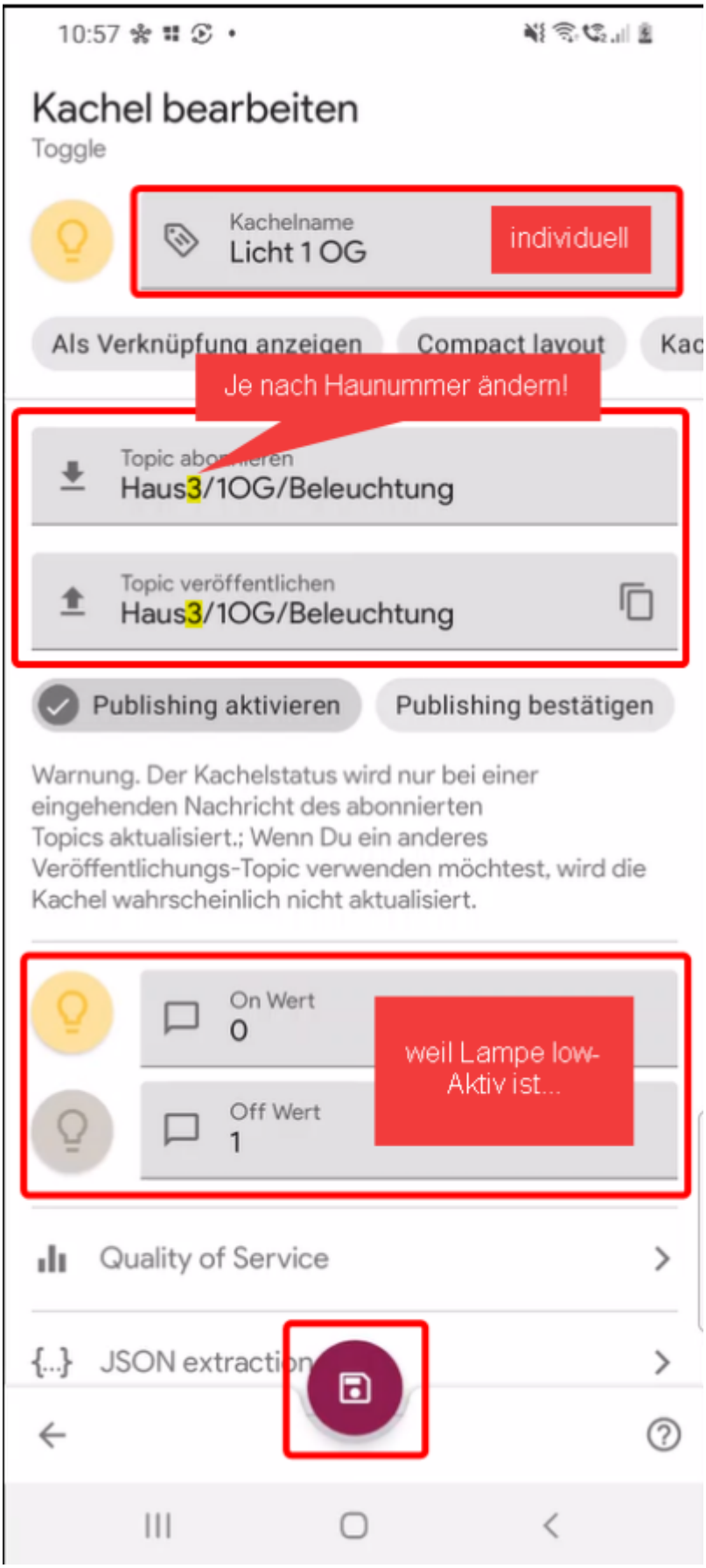
- Click on the + in the app



- Select a `Toggle` button



- Configure everything exactly as in the screenshot, but change the number at `Topic subscribe` and `Topic publish` according to the label you see on the top left of the dollhouse wagon.

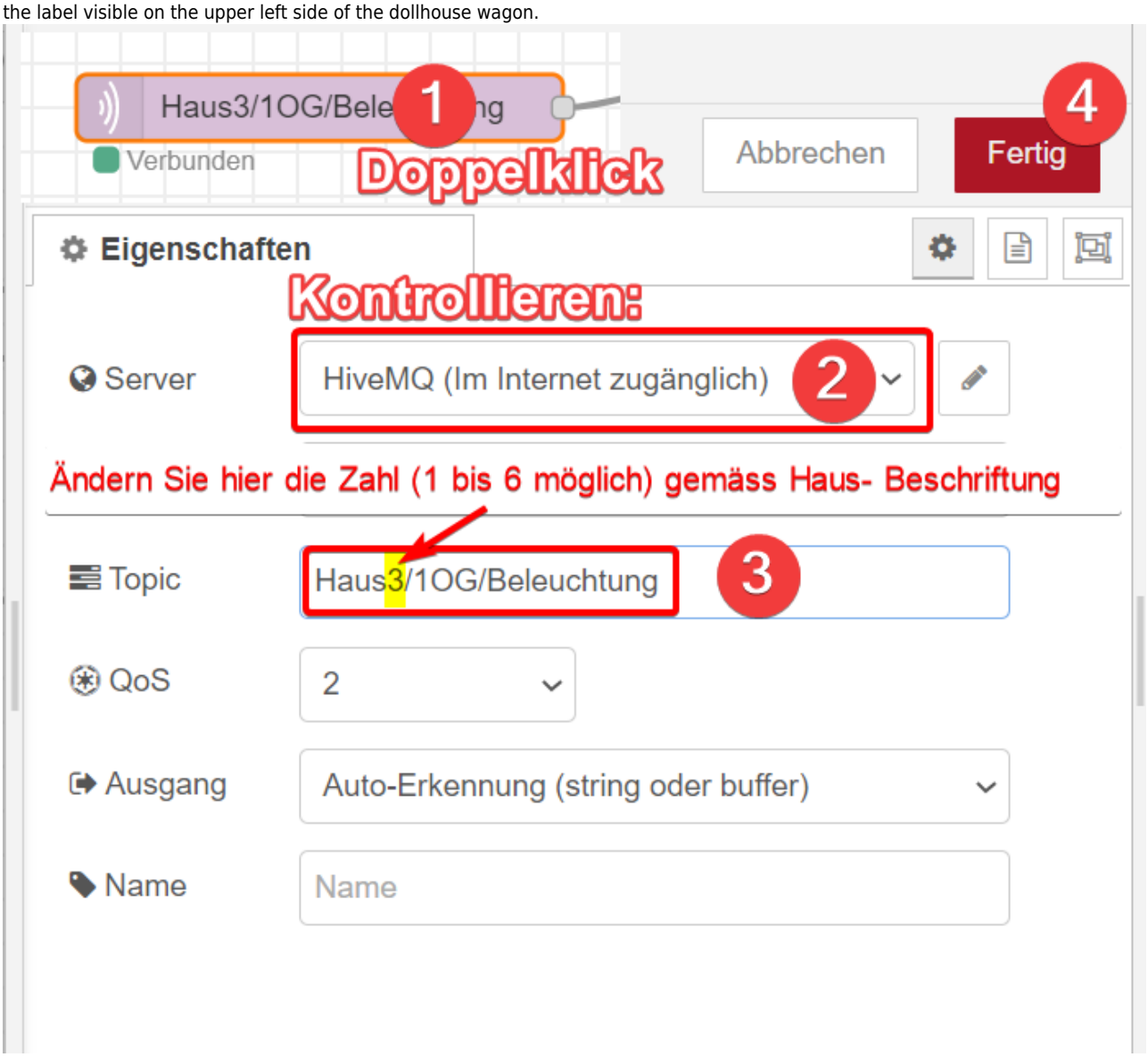


- By clicking on the question marks, you can individually select symbols and colors that influence the appearance of the button. I searched for `bulb` and defined different colors for on and off.

7.1.2 Implementation on the Raspberry Pi (MQTT Client 2)

- Return to the Node-RED user interface.

- Click on the „Lighting Example“ tab.
- In this flow, you will find a purple 'mqtt in' node. It is used to subscribe to any topic at a broker. This node has been pre-configured by the instructor. Therefore, a secure connection to an MQTT broker on the internet is already established. For authentication with this broker, a username and password are already set. You are now connected to the same broker as your smartphone.
- Double-clicking on this node will display the following settings. Change the number here (unless you have position 3), according to the label visible on the upper left side of the dollhouse wagon.



- Connect the output of the purple node to the input of the lighting in the 1st floor.
- Also, draw a connection to a debug node to check later if data is coming from the MQTT Broker.

7.1.3 Testing and Reflecting!

Now, the remote control with the smartphone should work. Discuss with your neighbor:

- What path do the data take from the smartphone to the dollhouse?
- Do the data stay within the same room?
- Are the data transmitted securely?
- Who else has control over your lighting?

Tip: You can locate the MQTT Broker by entering the broker's URL 'schnupperkurs-mqtt-broker.s2.eu.hivemq.cloud' here:

https://www.handy-sofort-orten.de/ip-adresse-orten/ip-adresse-und-standort-von-webservern-zeigen/

7.2 Expanding the Dashboard on the Smartphone (limited with iOS App)

Switch to the „MQTT Example“ tab in Node-RED.

For the following tasks, drag the following node into the flow area, which can be found under the „Network“ category:

- mqtt out (for outgoing messages)

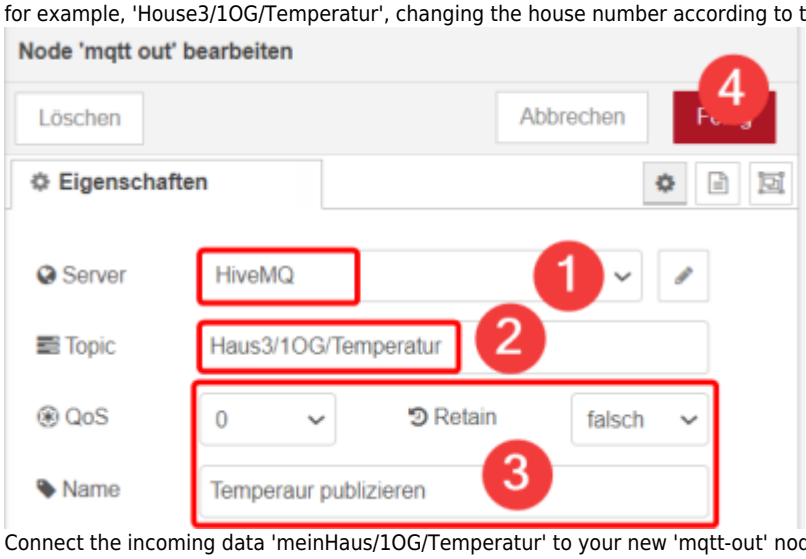
The **mqtt-out** node is used to publish messages.

Category	Network
Description	For first-time use, an MQTT Broker must be configured. Topic, QoS, and Retain flag can be specified. Incoming data is then published under the defined topic.

The **mqtt-in** node is used to subscribe to messages.

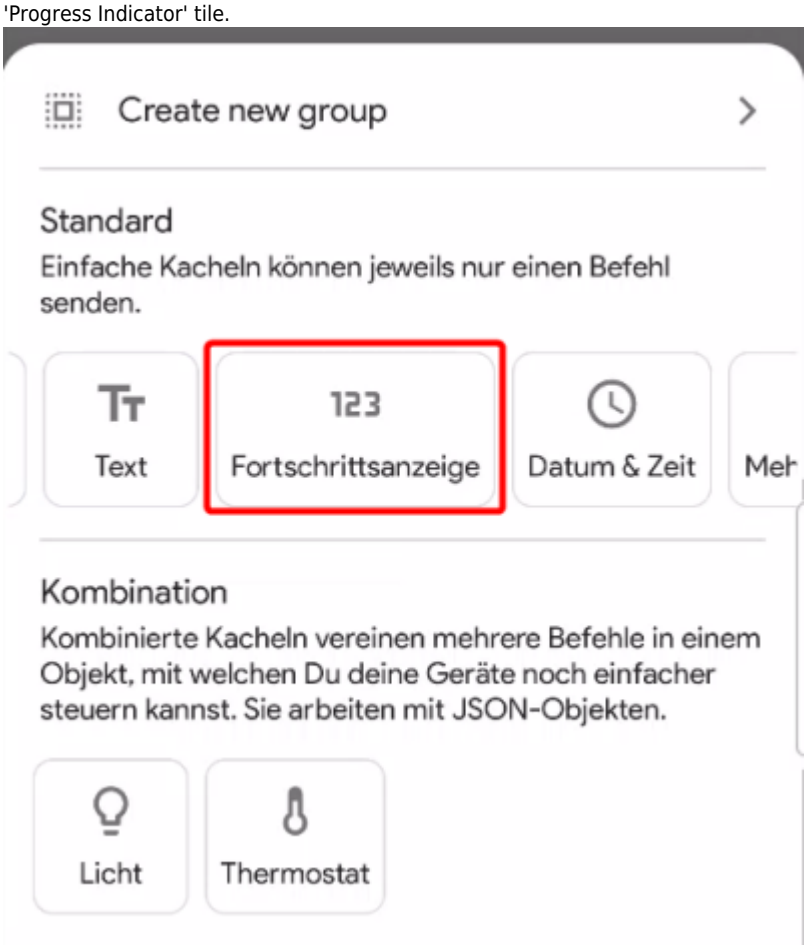
Category	Network
Description	For first-time use, an MQTT Broker must be configured. Topic, QoS, and Retain flag can be specified. When data is published under the topic, this data is output at the exit of the mqtt-in node. The data can be processed for output. For example, Node-RED can convert a JSON string into a parsed JSON object.

- Publish the actual temperature via MQTT so that you can retrieve the data later with your smartphone:
  - Edit the 'mqtt-out' node with a double-click
  - Choose 'HiveMQ' as the server (It is accessible on the internet, and also connected to your smartphone). Choose the topic, for example, 'House3/1OG/Temperatur', changing the house number according to the label on your house.

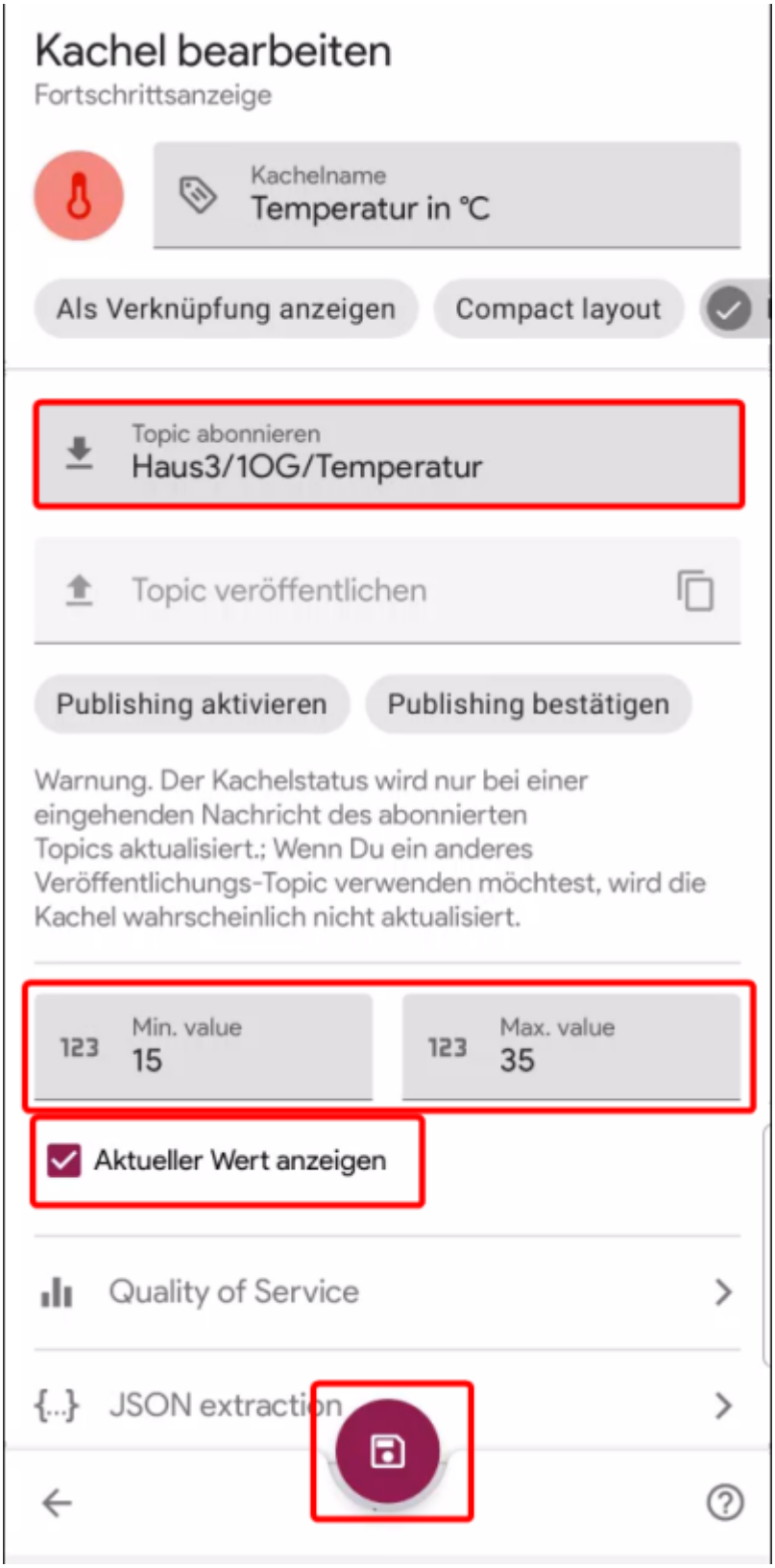


  - Connect the incoming data 'meinHaus/1OG/Temperatur' to your new 'mqtt-out' node.
- Add a display for the temperature on the smartphone app. This task can only be done with the Android app. You can use the

'Progress Indicator' tile.



- Subscribe to the same topic (in my case 'Haus3/1OG/Temperatur') that you are publishing in NodeRED



- Additional task: Expand the NodeRED flow and your smartphone app so that you can also change the set temperature via smartphone. Tip: Pull in and configure an mqtt-in node, then feed the data to the control circuit (set temperature). So, can you set the temperature for the 1st floor with your smartphone?

Tags

ue-node-red-introduction

UE: LoRa-Mapping and TTN

Short Description

Participants will be tasked with activating a LoRa antenna and a gateway. They will then examine the network coverage using IoT devices, specifically the Dragino Tracker D and IoT Cubes. Optionally, the Tracker D devices can be mounted on drones, which will automatically survey an area following programmed waypoints.

Contact Person

Fabian Reifler | Copyright © 2023 GBSSG

Framework Conditions

Framework Conditions	
Time Frame	5 lessons
Target Group (Professions)	people interested in technology, technicians
Requirements	enjoyment and talent in working with computers

Goals of the Teaching Unit

Participants will gain a comprehensive understanding of what a LoRaWAN infrastructure entails and the effectiveness of its radio technology in terms of power consumption and range. Through a hands-on approach, they will create their own infrastructure and conduct measurements to evaluate its performance.

Competency-Oriented Learning Objectives

After completing the section Register a GPS Tracker Dragino TrakerD on TTN, you will be able to...

- how to register a GPS Tacker (Model Tracker D) on TTN.

After studying the content Send GNSS- Data with the IoT Cube, observe RSSI and SNR, create a dashboard for Mapping, you will be able to...

- to programme the Cube so that it transmits its position via LoRa
- explain the term SNR (Signal-to-Noise Ratio)
- explain the term RSSI (Received Signal Strength Indicator)
- use RSSI and SNR to estimate how good the reception of an IoT device is.

After studying the content LoRaMapping with a drone, you will be able to...

- equip a drone with a LoRa device
- set way points, to fly a drone automatically
- map out the LoRa network coverage around the antenna

Teaching Materials

Here all teaching materials are listed in chronological order.

Links for Students	Content / Activity	Links for Teachers
Register a GPS Tracker Dragino TrakerD on TTN	How to register a tracker D to TTN[3)]	
LoRaMapping with a drone	using a drone	
Send GNSS- Data with the IoT Cube, observe RSSI and SNR, create a dashboard for Mapping	Programm IoT Cubes to get RSSI[4)] and SNR[5)]	

3)

The Things Network
[4]
Received Signal Strength Indication
[5]
Signal-to-Noise Ratio

Register a GPS Tracker Dragino TrakerD on TTN

Shows an example of how to register a Dragino Model Tracker D on TTN.

Competency-Oriented Learning Objectives

After completing the section Register a GPS Tracker Dragino TrakerD on TTN, you will be able to...

- how to register a GPS Tacker (Model Tracker D) on TTN.

1. Official Documentation

TrackerD Official Documentation

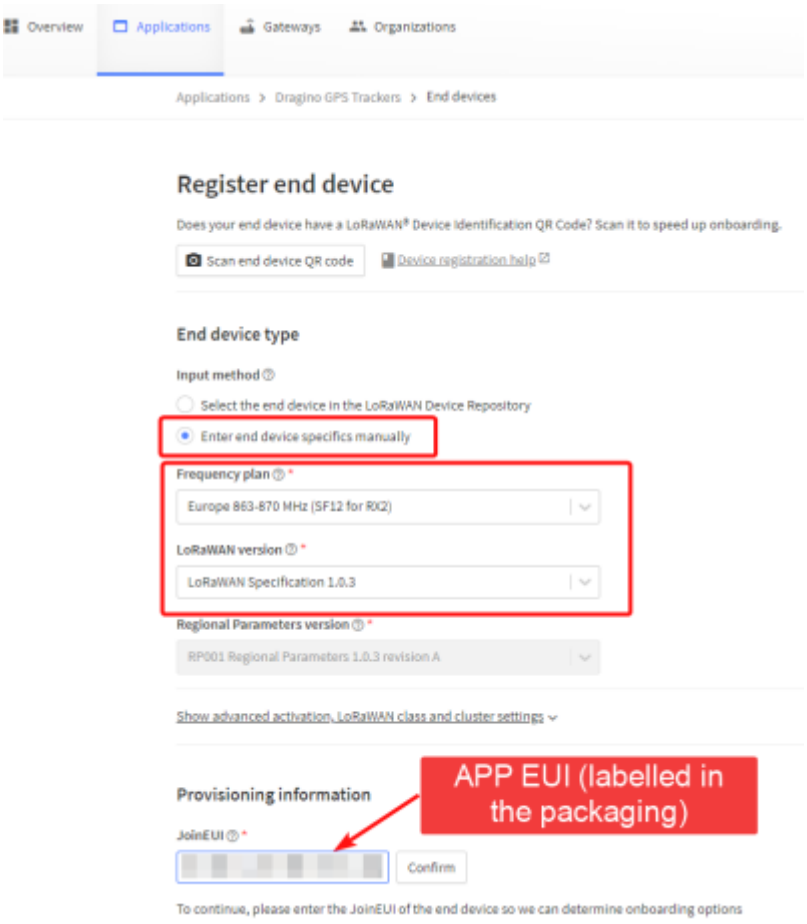
2. Create an Application or select an existing one

1. Navigate to this website: TTN Console, Applications
2. Create an account if you do not have one already
3. Ask your teacher, that he adds you as a collaborator
4. Select an existing application `app-dragino-trackers` (visible because I created this application and invited you as a collaborator).
5. If you create your own application (not necessary), it looks like this:

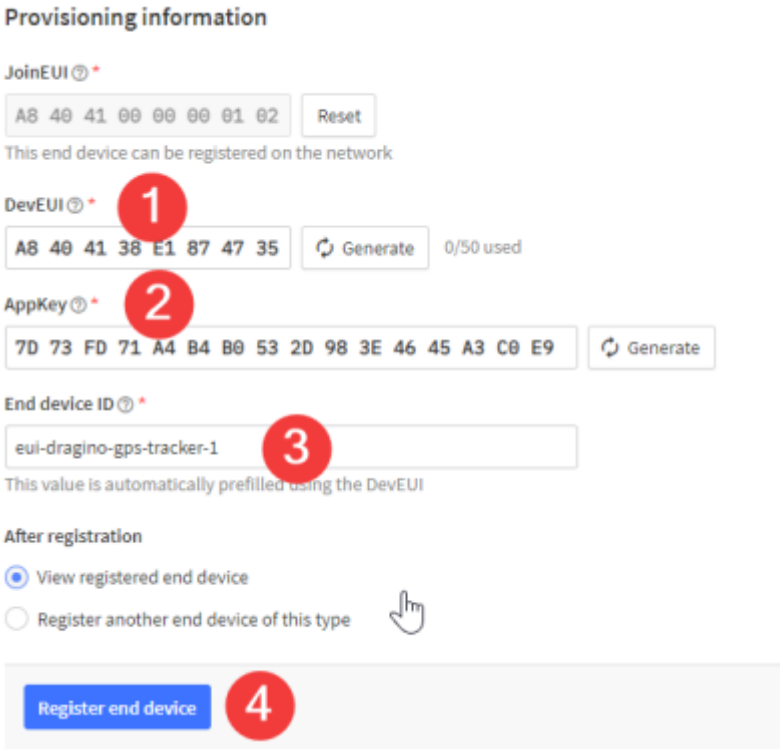
1.
2. give a application ID
3. an an application name
4. a description
5. hit „Create application“

3. Register a new end device to the application

1. Input method: manually
2. Enter this information:

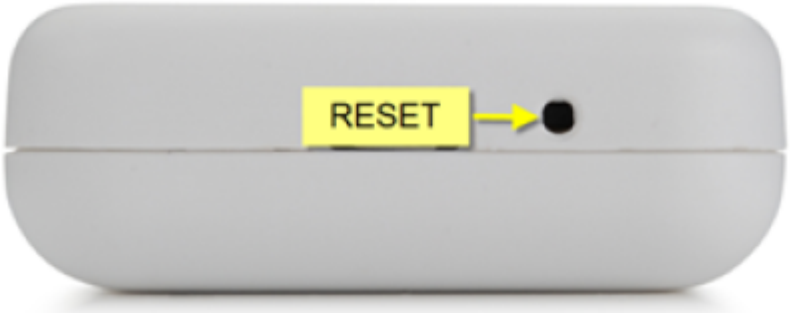


3. Enter DevEUI and AppKey (labelled in the packaging):

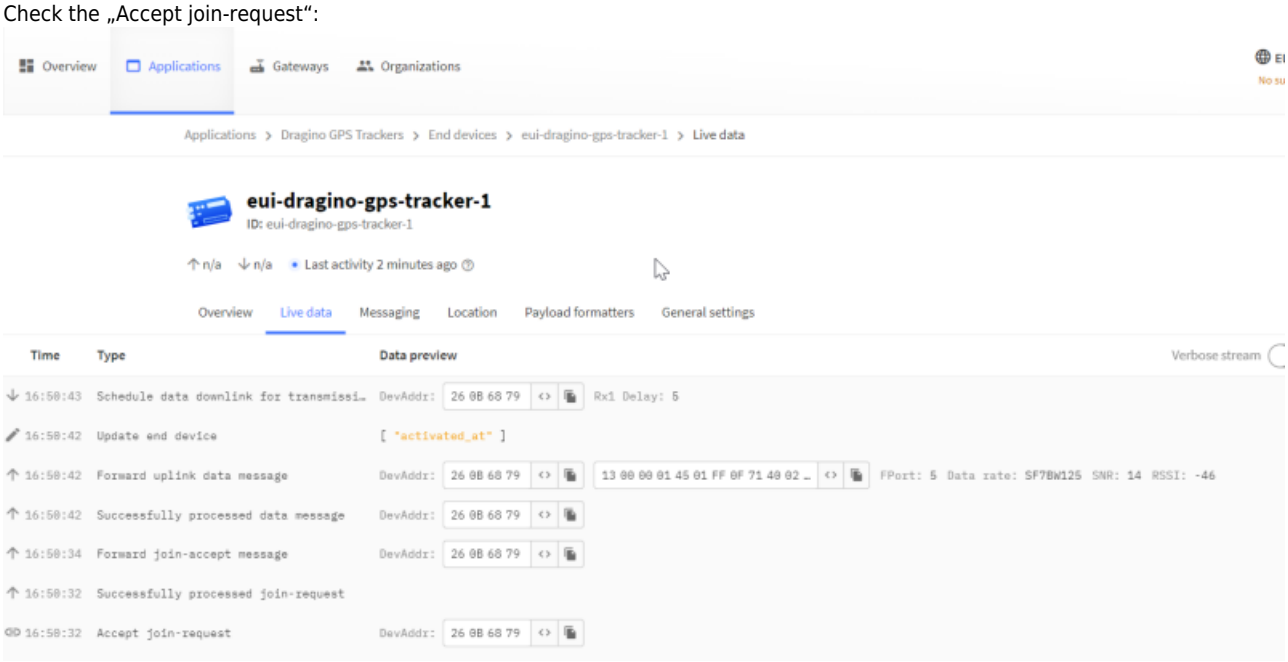


4. Check, if its working

1. Press Reset Button (with a needle)



2. Check the „Accept join-request“:



5. Change Configuration of the Device with Downlink- Messages

The Dragino TrackerD can be reconfigured through one or more downlink messages. I have successfully done this to reduce the transmission intervals and to transmit messages only when movement is detected. Downlinks can be initiated directly from TTN. Please refer to the manual for detailed instructions. Pay particular attention to the following chapters:

- 2.10 Transport Mode
- 3.2.1 Set Transmit Interval
- 3.2.3 Set Transport Mode Packet Transmission Interval
- 3.2.6 Disable/Enable Transport Mode
- 3.2.16 Get or Set Threshold for motion detect

Link to the Manual: TrackerD Official Documentation

6. Your Job

- Register all available TrackerD GPS trackers within your group on TTN in the existing application `app-dragino-trackers`
- Test the trackers rudimentary, if they are joining to the network.
- Modify the configuration as follows (which I believe would be suitable for our purpose):
    - Set MTDC (Transport Mode Packet Interval) to 20 seconds –> The device reports every 10 seconds when in motion. Note: This is a very short interval for LoRa.
    - Set TDC (Uplink Interval) to 900 seconds –> The device reports every 15 minutes, even without movement.
    - Set INTWK to 1 –> Transport Mode is activated
    - Set Threshold for motion detect (to detect walk motion)
- Test the tracker outside of the building

Tags

ue-lora-mapping-and-ttn

LoRaMapping with a drone

We are planning to utilize a drone for LoRa mapping. This involves equipping a drone with a LoRa device, enabling it to collect and transmit data over long distances using the LoRa wireless protocol. The primary objective is to map out LoRa network coverage across different areas. The drone will fly predefined routes.

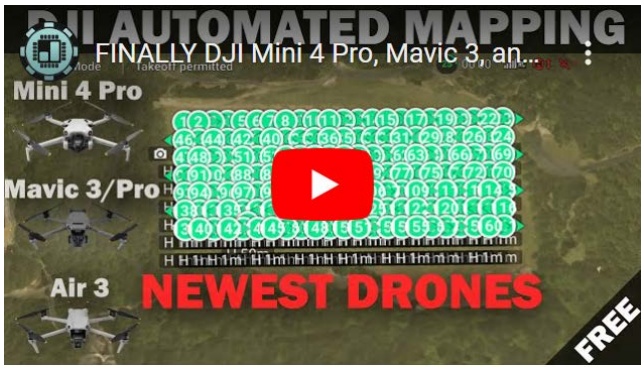
Competency-Oriented Learning Objectives

After studying the content LoRaMapping with a drone, you will be able to...

- equip a drone with a LoRa device
- set way points, to fly a drone automatically
- map out the LoRa network coverage around the antenna

1. How to program waypoints

1.1 Video tutorial



Video

1.2 Generate waypoint files to fly the drone automatically

Use this website to generate the waypoint file

waypointmap.com

Tags

If the page is used for a teaching unit, please leave a tag here. The tags are used for the automatic compilation of learning objectives. With the include_n tag, you can control the order of the contributions in the learning objectives.

ue-lora-mapping-and-ttn

Send GNSS- Data with the IoT Cube, observe RSSI and SNR, create a dashboard for Mapping

In wireless communication, a receiver needs sufficient signal strength and an adequate signal-to-noise ratio (SNR) to separate the useful signal from the modulated carrier. Here, you will become familiar with the two most commonly used indicators of signal strength - RSSI (Received Signal Strength Indicator) and SNR (Signal-to-Noise Ratio).

Competency-Oriented Learning Objectives

After studying the content Send GNSS- Data with the IoT Cube, observe RSSI and SNR, create a dashboard for Mapping, you will be able to...

- to programme the Cube so that it transmits its position via LoRa
- explain the term SNR (Signal-to-Noise Ratio)
- explain the term RSSI (Received Signal Strength Indicator)
- use RSSI and SNR to estimate how good the reception of an IoT device is.

1. Test Program for IoT Cube

- Connect the IoT Cube to the notebook in front of you via USB.
  - Connect the red-marked plug on the side with the antenna (power supply only)
  - Connect the black plug from above to the Micro:Bit
- Open the following website:https://makecode.microbit.org/_b03T9JTw0EP9
- Click on `Edit`
- Study the program
- Download this program to the IoT Cube
- Feel free to edit the program, if you know what you're doing

2. How good is the radio Signal?

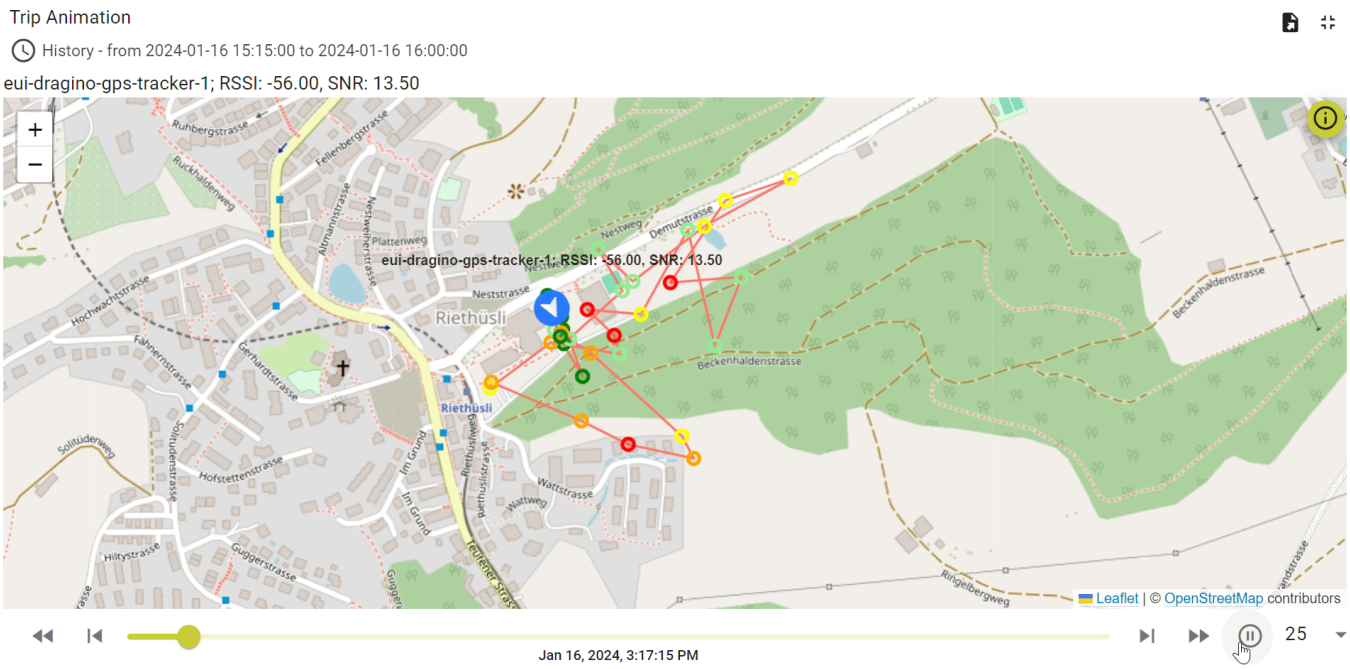
- Research online what the RSSI and SNR values indicate.

3. Go outside with your computer. Observe the data in TTN.

- Log into the TTN console. Maybe you need to set up a hotspot outside.
- See if realistic position data arrives. (Go to your device, e.g. eui-2023-c-09 and click on **Live data**
- Attention: After switching on the power supply, the GNSS module needs approx. 3 minutes until it delivers valid data. The GNSS module will only work outdoors under the open sky.
- Can you find RSSI Values and SNR values in the received **Live data**?
- Can you notice differences when you are further away from the gateway?

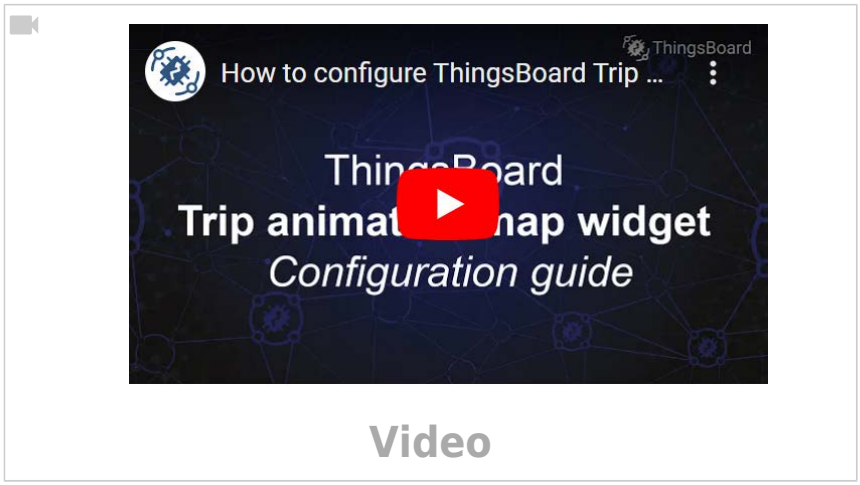
4. Dashboard on ClavisCloud

- Open this website ClavisCloud(Things Board)
- Login with iot@smartfeld.ch (Get the password from Fabian)
- Create a dashboard, that displays the signal strength (RSSI) and the signal-to-noise ratio (SNR) of the live data and historical data of your IoT Cube in combination with its positioning data on a map. Example:



Tipps:

- Use the ClavisCloud to create the Dashboard.
- Select the **Trip Animation** widget.
- Here is a tutorial, that helps you:



- It is possible, to add a **Point color function**. It could look like that:

```
var rssi = data['rssi'];

if (typeof rssi === 'undefined') {
    return 'grey'; // Farbe, wenn RSSI undefiniert ist
} else if (rssi < -120) {
    return 'red'; // Sehr schwaches Signal
} else if (rssi < -100) {
    return 'orange'; // Schwaches Signal
} else if (rssi < -80) {
    return 'yellow'; // Mittleres Signal
} else if (rssi < -60) {
    return 'lightgreen'; // Gutes Signal
} else if (rssi < -40) {
    return 'green'; // Starkes Signal
} else {
    return 'blue'; // Sehr starkes Signal
}
```

Tags

ue-lora-mapping-and-ttn