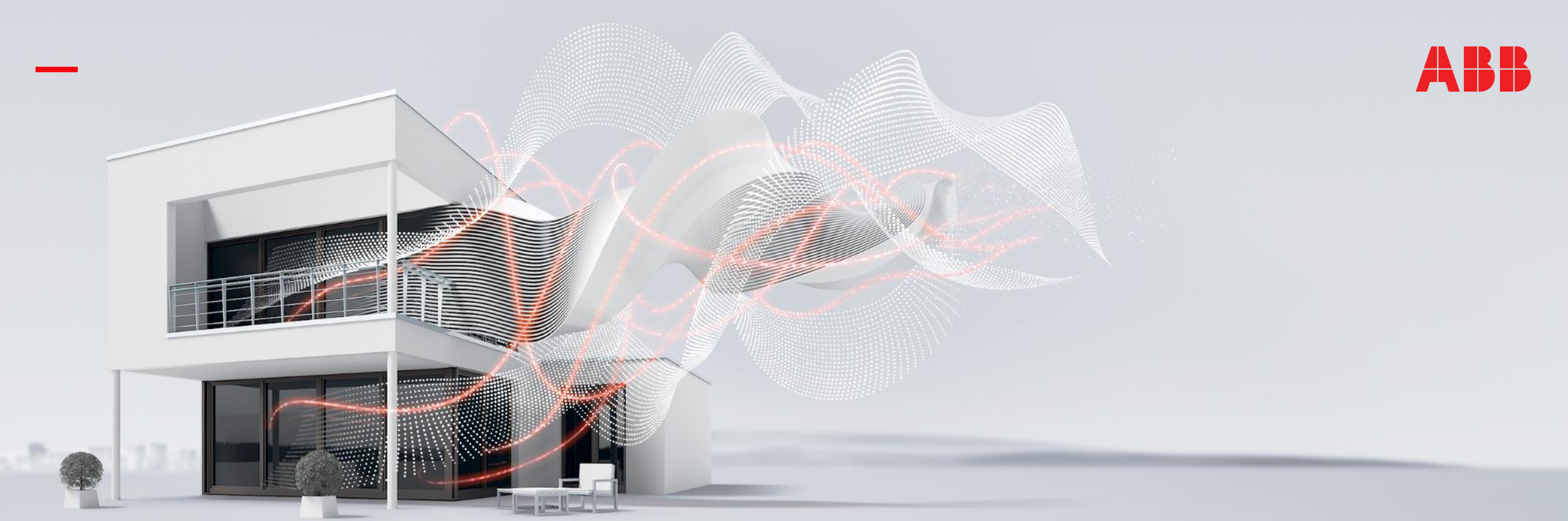


DECEMBER 2020

ABB-free@home® – local REST API

Webinar – Competence Center Europe – Smart Buildings

Thorsten Reibel, Jürgen Schilder, Stefan Grosse, Martin Wichary & Olaf Stutzenberger

 Document ID.: Rev.:

Agenda

Introduction

API – Application Programming Interface

ABB Developer Portal

Installation

Swagger: API Documentation & Design Tool

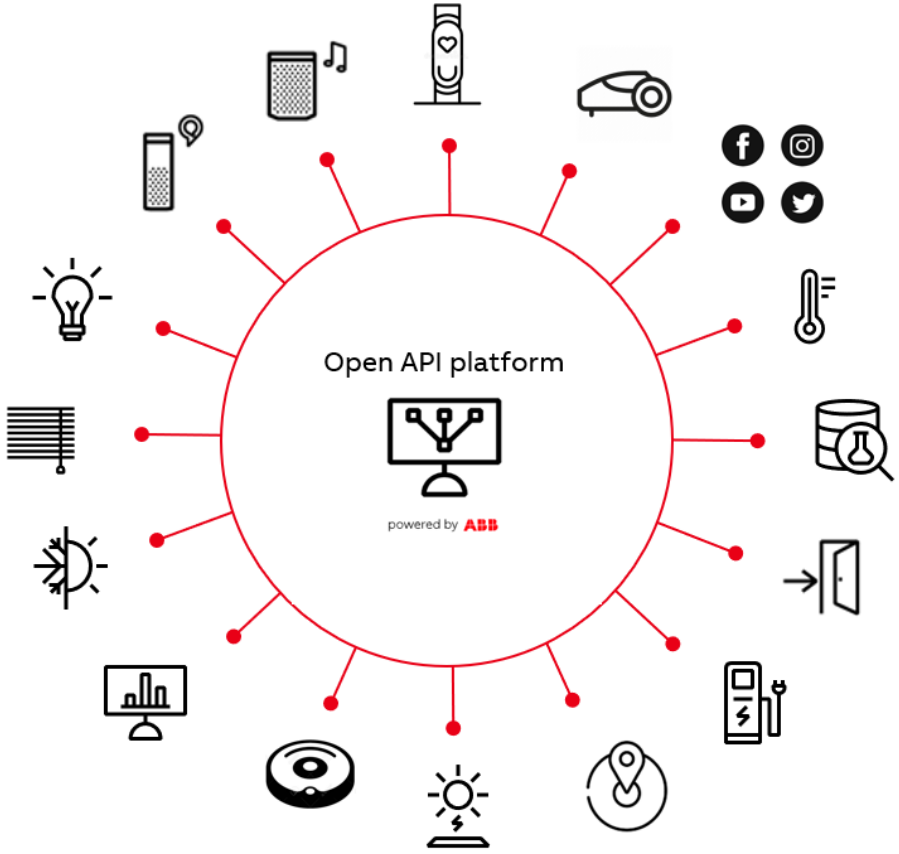


ABB-free@home® – local REST API

Introduction

ABB-free@home® – local REST API

Introduction

Motivation

- ABB-free@home® transforms the house or apartment into an intelligent home. Whether lighting, shading, room temperature, door communication or entertainment – comfort, safety and efficiency can be controlled intuitively
- The Smarter Home Developer Program has been created, so that this promise can be extended to other devices and services in the household
- This will give our customers the possibility to create a holistic smart home solution
- The “free@home local REST API“ will be the next interface extending our open systems approach

Smarter Home Developer Program

New perspectives on networking for external developers. With the "Smarter Home Developer Program" ABB opens its systems for developers and partners.

ABB-free@home® – local REST API

Introduction

Goals

- Customers are able to interconnect additional systems and solutions with free@home and to use them in one ecosystem
  - free@home can be the central system and can be extended with additional devices or services from 3rd party or even ABB offerings
  - free@home can be connected to other central systems inside a building (like BMS)
- The local REST API only works on premise and inside the same building
  - No data leaving the building
  - Limited to be used inside the same building
- Giving our customers the feeling of one overall “smart home” regardless of source and matter

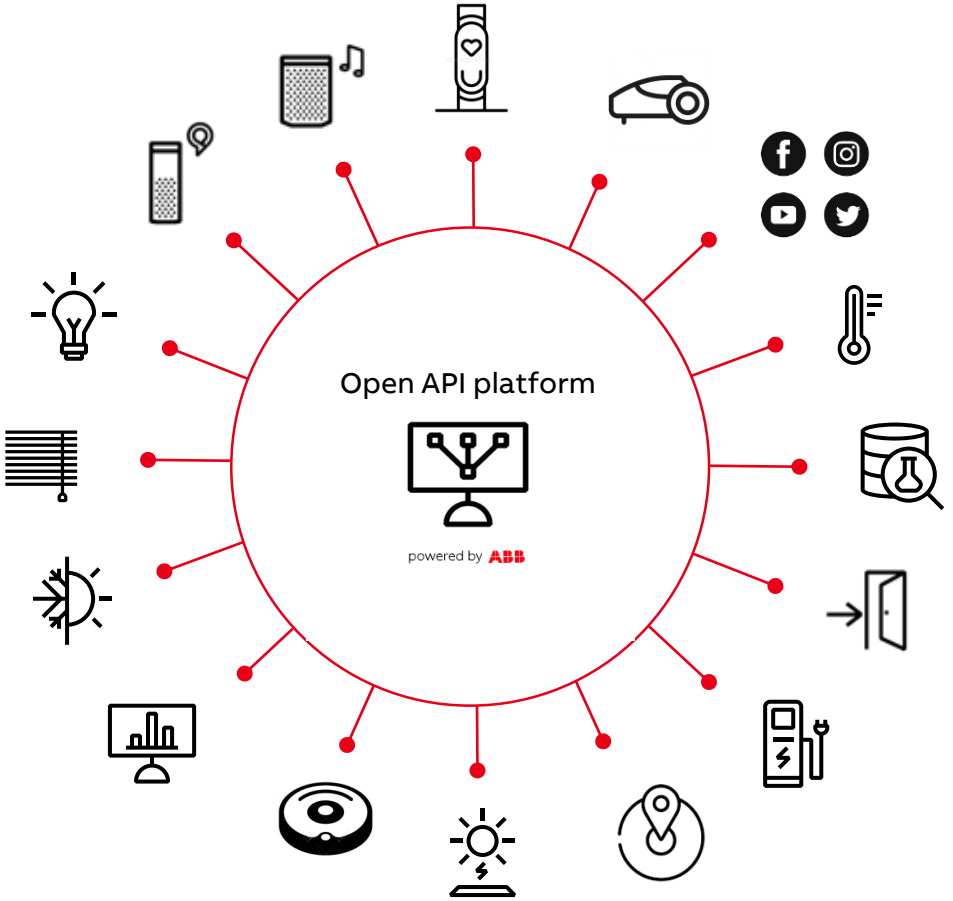


ABB-free@home® – local REST API

Introduction – USPs

Seamless integration into one system

Seamless integration into a professional installed smart home system

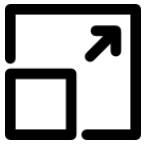
Add 3rd party systems and functions into free@home to sensors, panel displays, scenes and applications like timers or actions

Protecting customers‘ data

Easy to expand with other solutions due to open and standardized interface

Suitable for all markets globally where free@home is available

Targeting different building segments



Suitable for single family house, where tech-savvy end-user will integrate 3rd party solutions into free@home

Suitable for pre-fitted apartments, that come with free@home and other systems

(There must be a validation if the free@home REST API doesn’t fit better to the needs in some cases)

Easy to expand

Easy to expand with other solutions due to open and standardized interface

Suitable for all markets globally where free@home is available

ABB-free@home® – local REST API

Introduction

Example use cases for end customers

Due to the local REST API tech savvy user or developer can develop own applications and can bring different solutions and services together with free@home

The integration of free@home into open Smart Home Hubs offers the possibility to enable many different use cases also to other end customers

Examples:

- Bring vacuum cleaner into free@home
- Control your smart robot mower
- Write and persist data inside a dashboard or google sheet

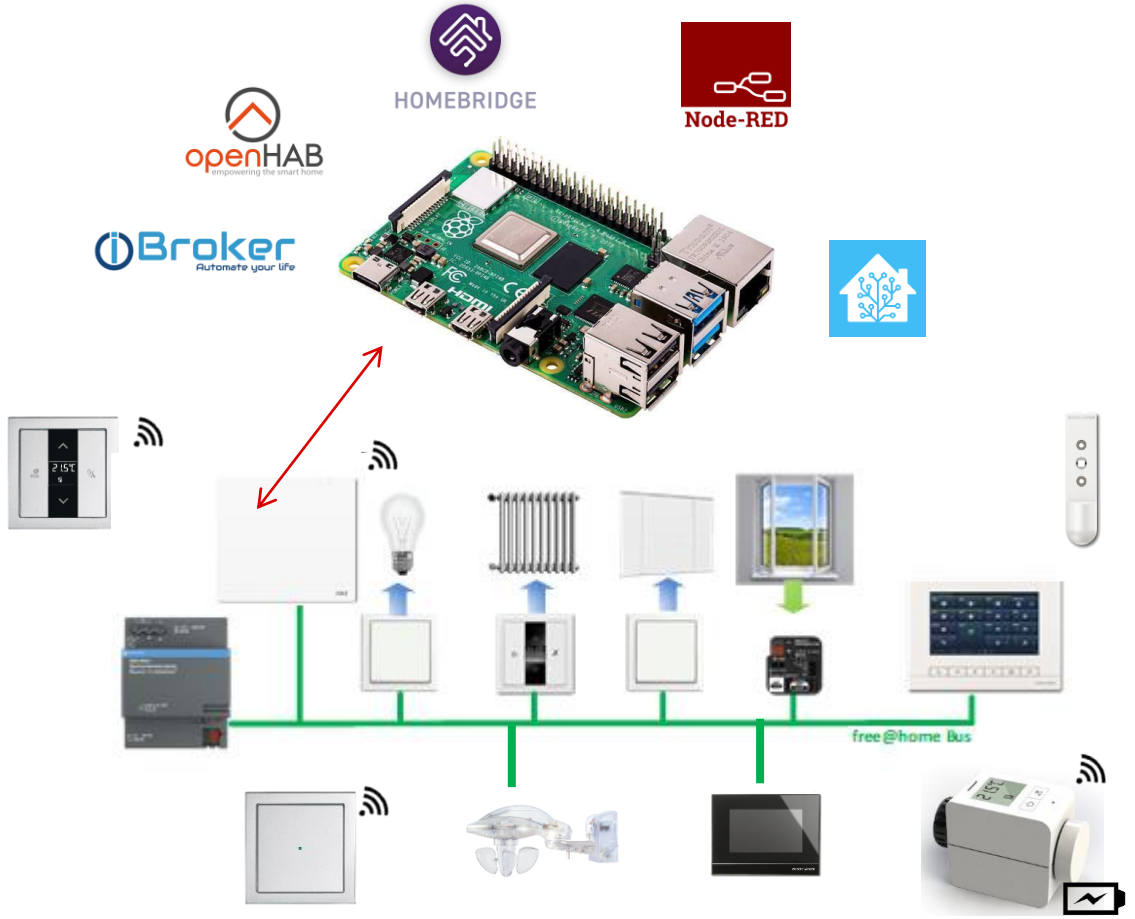


ABB-free@home® – local REST API

Introduction

Example use cases for professional partners

The integration of free@home into BMS offers the possibility to enable many different use cases

In this B2B cases it depends if the BMS manufacturer decides to integrate towards the local or cloud API of free@home because of different pros and contras

Examples:

- Hotels with multiple free@home systems installed
- Multi tenant apartment houses
- free@home and Energy Solutions like EV Charger and Solar

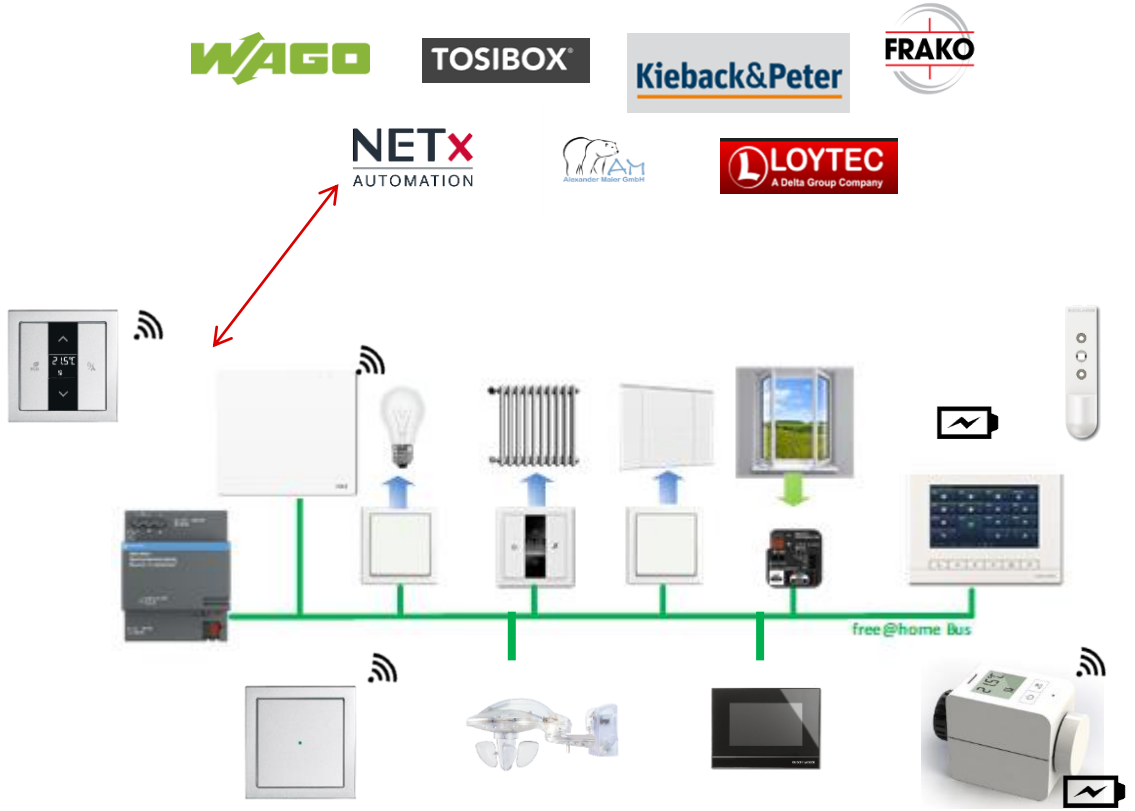


ABB-free@home® – local REST API

API – Application Programming Interface

ABB-free@home® – local REST API

API – Application Programming Interface

Questions

- APIs. What are they?
- What can I do with an API?
- Do I need to understand them?
- What are my customers constantly talking about?
- Do I really need to monitor APIs?
- How to get started?

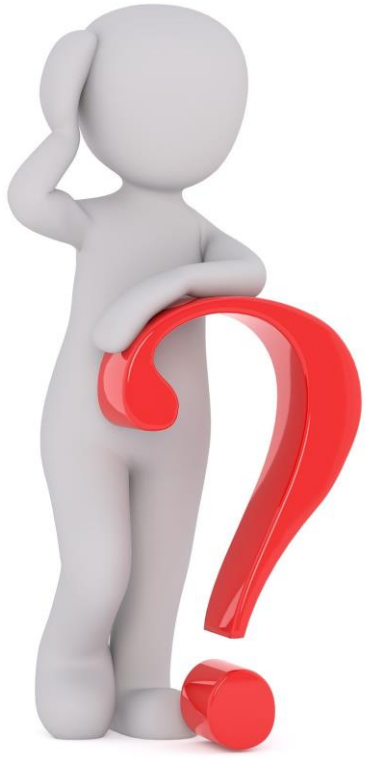


ABB-free@home® – local REST API

API – Application Programming Interface

General Information

- **APIs** allow different systems to communicate with each other. It lets a mobile app talk to a social network, or a payments service work the same way across the web, TV, mobile and other platforms
- **REST** is a Representational State Transfer. It's a stateless, client-server, cacheable **communication protocol** that usually uses HTTP

Source: https://www.redhat.com/de/topics/api/what-are-application-programming-interfaces

ABB-free@home® – local REST API

API – Application Programming Interface

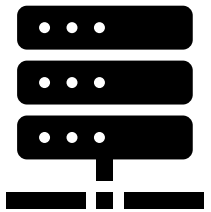
Cloud vs. Local API

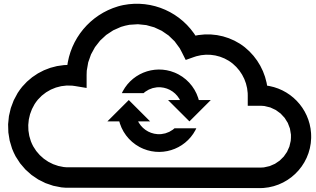
Cloud API / Web API

- Cloud-based API integrated in MyBuilding
- Controlled data-exchange with free@home to extend the Smart Home functionality
- Ideal for making Smart Home services available for free@home

Local API

- New interface for application programming (API) for ABB-free@home®





Source: https://www.redhat.com/de/topics/api/what-are-application-programming-interfaces

ABB-free@home® – local REST API

API – Application Programming Interface

Internal vs. Public vs. External API

Internal API / Private API

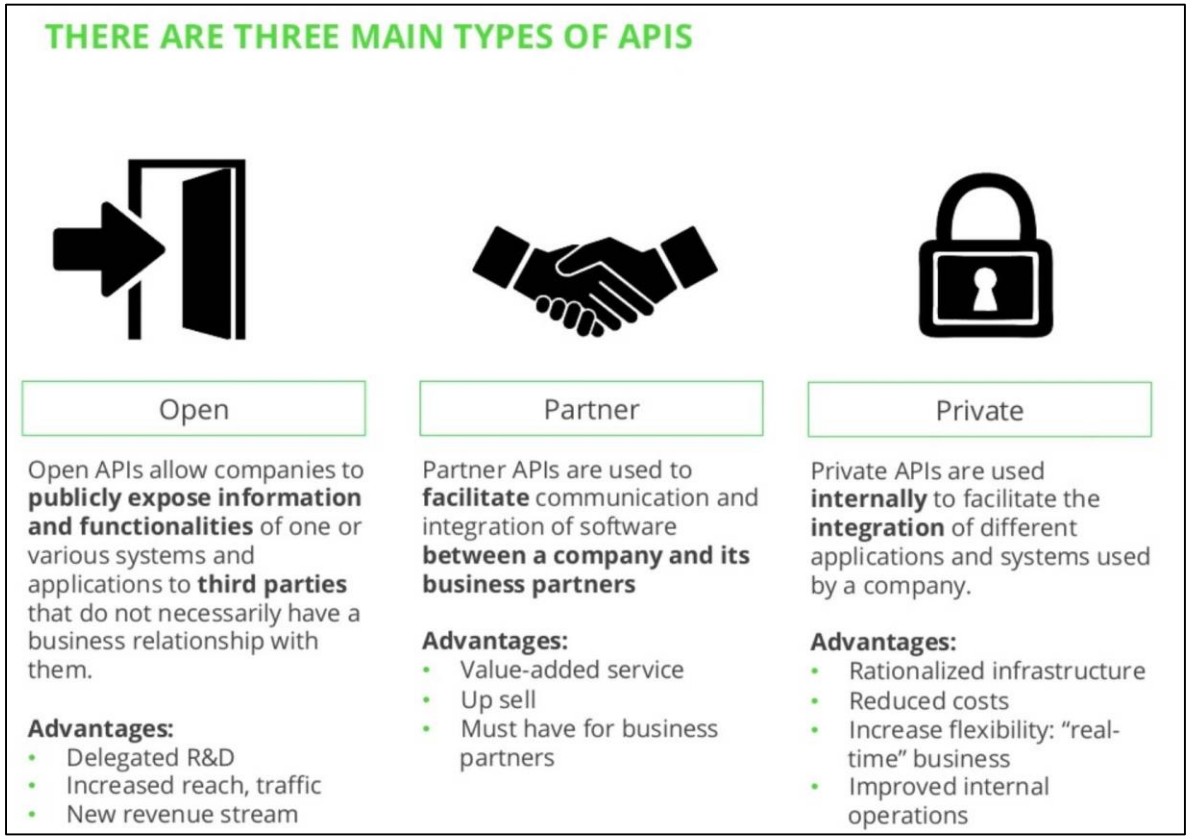
- A private API is intended for internal use only
- Example: free@home connection to MyBuildings

Public API / Partner API

- The API is shared with specific business partners
- This allows you to tap additional sources of income without compromising quality
- Example: free@home connection to HomeConnect, Miele, Sonos, HUE, etc.

External API / Open API

- This API is available to everyone
- This allows third parties to develop apps to interact with your API and opens up opportunities for innovation
- free@home open for end customers



Source: https://www.redhat.com/de/topics/api/what-are-application-programming-interfaces
Source 2: https://nordicapis.com/how-nordic-businesses-are-choosing-between-private-partner-and-public-api-strategies/

ABB-free@home® – local REST API

API – Application Programming Interface

Situation today

- free@home is already working with API solutions
- The usage of API solutions is not new
- But: the available solutions were limited to the integrated solutions of ABB and their partners
- Now the end customer can generate new use cases and integrations on his own

ABB-free@home® – local REST API

ABB Developer Portal

ABB-free@home® – local REST API

ABB Developer Portal

Introduction

- Whether you are looking to integrate our APIs into your application, website or service, you can find out how to do it at our ABB Developer Portal
- Our APIs enable you to create value added experiences with Smart Home and Smart Building solutions
  - Documentations
  - Tutorials
  - APIs
  - Products
  - Support
  - FAQ
- https://developer.eu.mybuildings.abb.com/

ABB-free@home® – local REST API

ABB Developer Portal

First steps

- To use the Cloud API an ABB MyBuildings account is needed
- Register on MyBuildings and send a request on the website to get access to the API

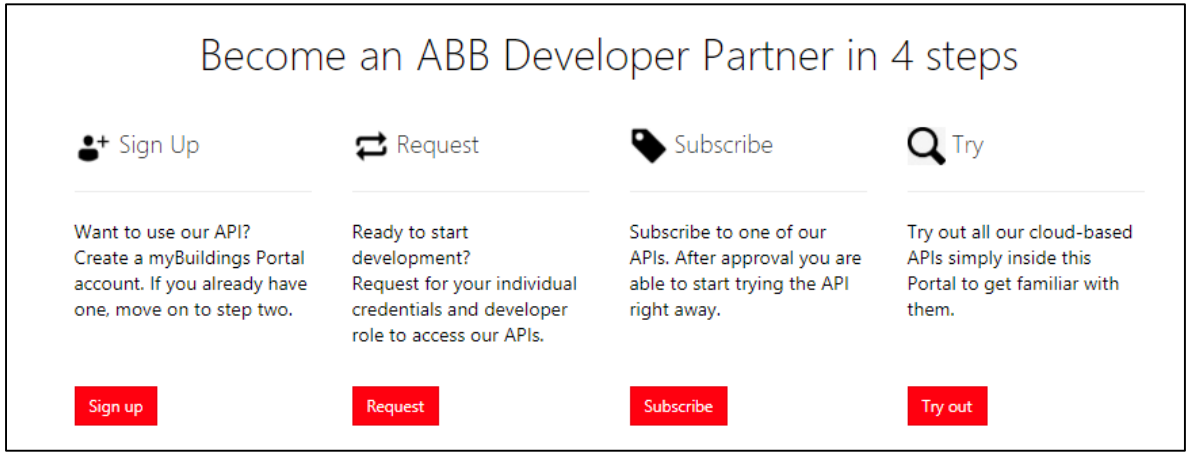
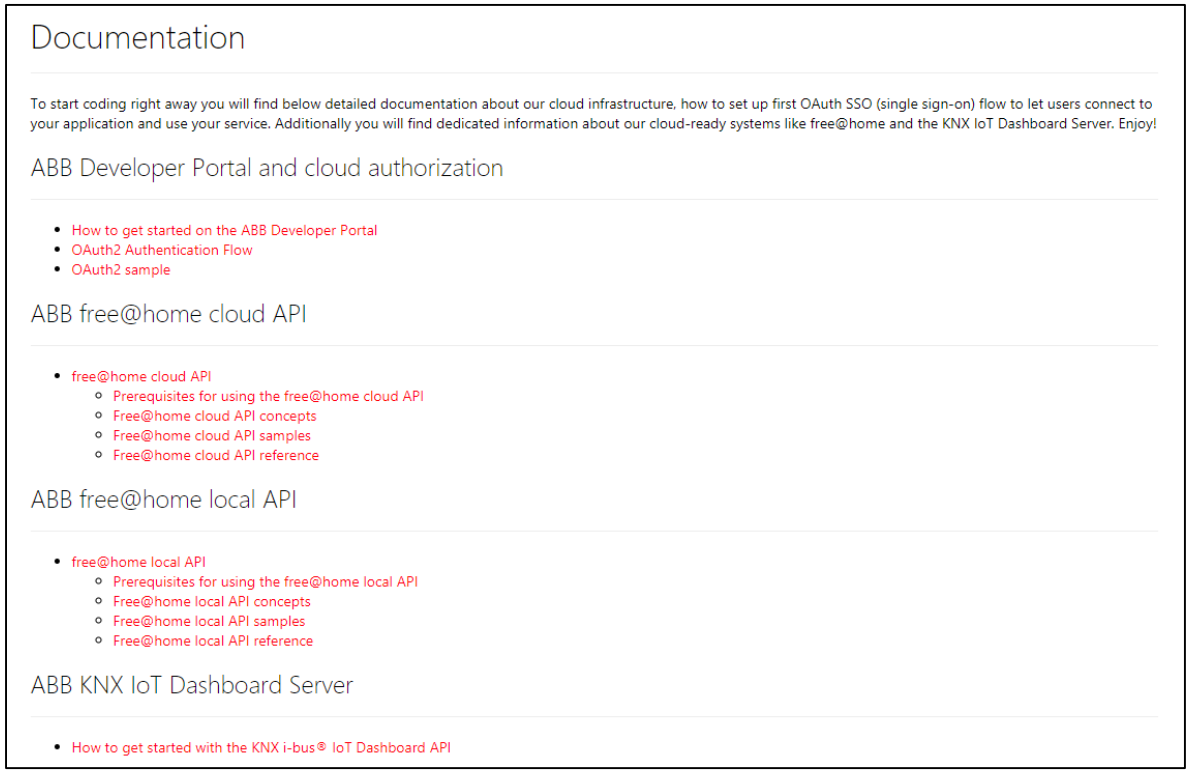


ABB-free@home® – local REST API

ABB Developer Portal

Documentations

- Interested users will find many documentations, tutorials and examples under the tab “Documentation”
- Details step-by-step guides for all available APIs



Documentation

To start coding right away you will find below detailed documentation about our cloud infrastructure, how to set up first OAuth SSO (single sign-on) flow to let users connect to your application and use your service. Additionally you will find dedicated information about our cloud-ready systems like free@home and the KNX IoT Dashboard Server. Enjoy!

ABB Developer Portal and cloud authorization

- How to get started on the ABB Developer Portal
- OAuth2 Authentication Flow
- OAuth2 sample

ABB free@home cloud API

- free@home cloud API
  - Prerequisites for using the free@home cloud API
  - Free@home cloud API concepts
  - Free@home cloud API samples
  - Free@home cloud API reference

ABB free@home local API

- free@home local API
  - Prerequisites for using the free@home local API
  - Free@home local API concepts
  - Free@home local API samples
  - Free@home local API reference

ABB KNX IoT Dashboard Server

- How to get started with the KNX i-bus® IoT Dashboard API

ABB-free@home® – local REST API

Installation

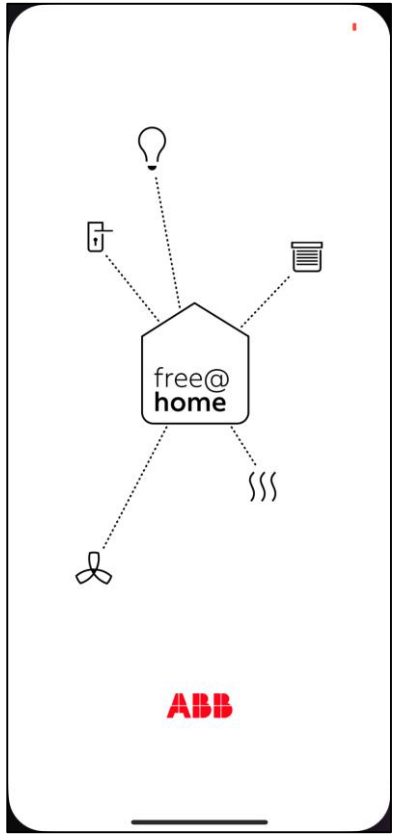
ABB-free@home® – local REST API

Installation

First steps

- The free@home local API is capable accessing the free@home system access points of end users when they are connected to the local network. For this, a few prerequisites must be met by the end user
- This API is capable of controlling a SysAP in the local network
- The end user needs an ABB-free@home smart access point with version 2.6.0 or later
- The free@home next App for a mobile device
- The end user must enable the local API in the SysAP settings of the free@home next app, found under:

  More -> Settings -> Local API



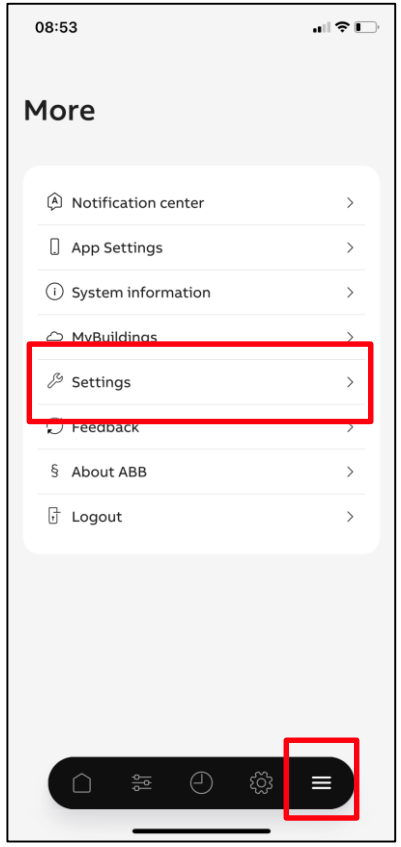


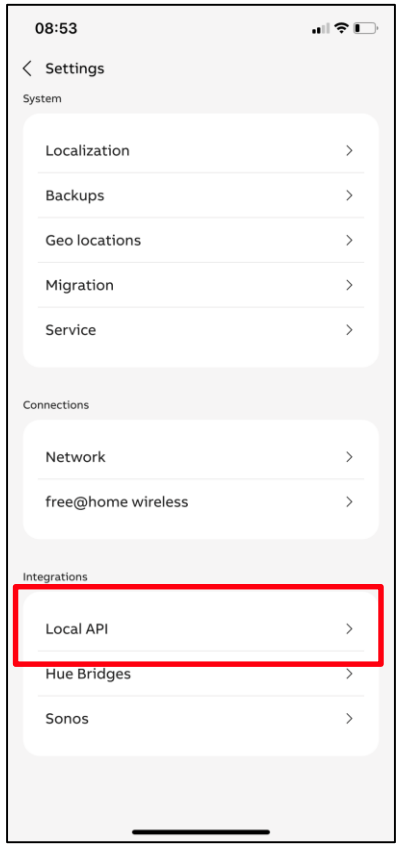
ABB-free@home® – local REST API

Installation

First steps

- The free@home local API is capable accessing the free@home system access points of end users when they are connected to the local network. For this, a few prerequisites must be met by the end user
- This API is capable of controlling a SysAP in the local network
- The end user needs an ABB-free@home smart access point with version 2.6.0 or later
- The free@home next App for a mobile device
- The end user must enable the local API in the SysAP settings of the free@home next app, found under:

  More -> Settings -> Local API



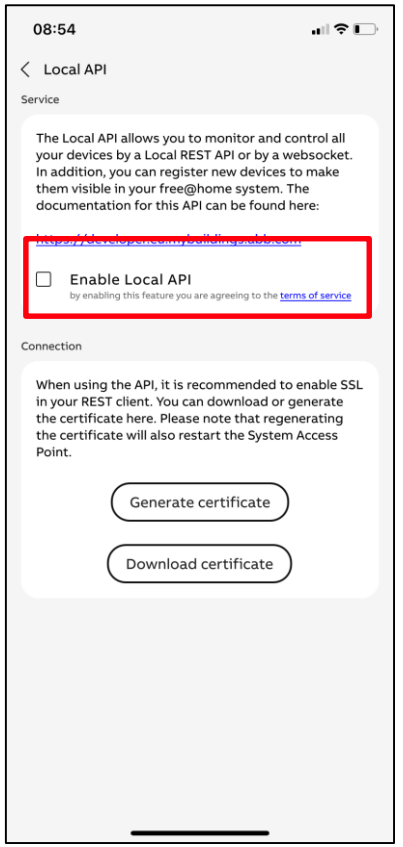


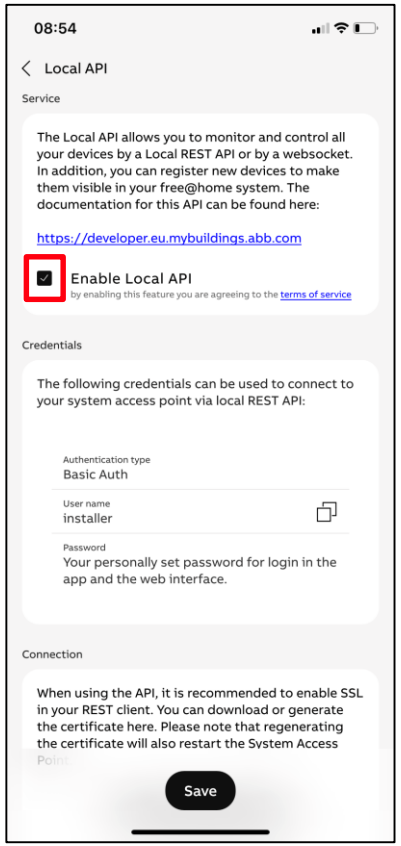
ABB-free@home® – local REST API

Installation

First steps

- The free@home local API is capable accessing the free@home system access points of end users when they are connected to the local network. For this, a few prerequisites must be met by the end user
- This API is capable of controlling a SysAP in the local network
- The end user needs an ABB-free@home smart access point with version 2.6.0 or later
- The free@home next App for a mobile device
- The end user must enable the local API in the SysAP settings of the free@home next app, found under:

  More -> Settings -> Local API



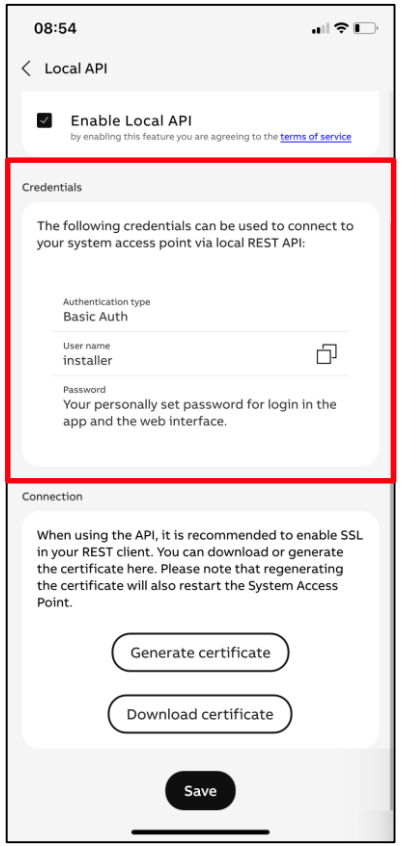


ABB-free@home® – local REST API

Installation

First steps

- SSL certificate to generate an encrypted communication between free@home and your Raspberry Pi
- The generated certificate must be integrated into Node-RED
- Website will be displayed as HTTPS (Hyper Text Transfer Protocol Secure)

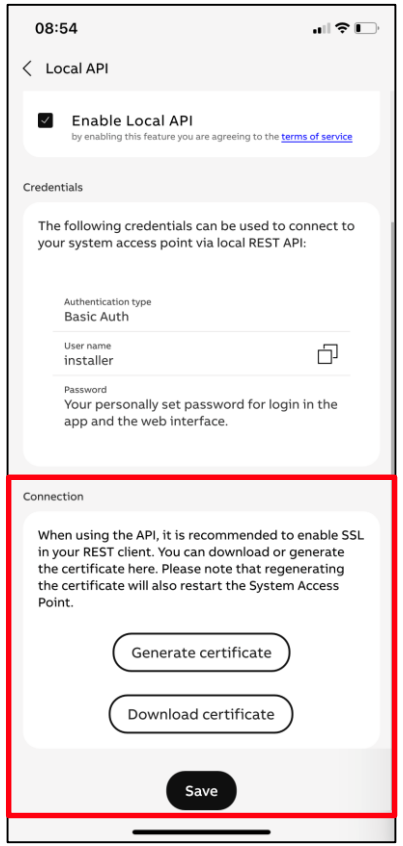


ABB-free@home® – local REST API

Testing the API

ABB-free@home® – local REST API

Testing the API

Web interface

– To test the API, the calls can be triggered via a web interface. This is available at

http://{IP-Address of the System Access Point}/swagger

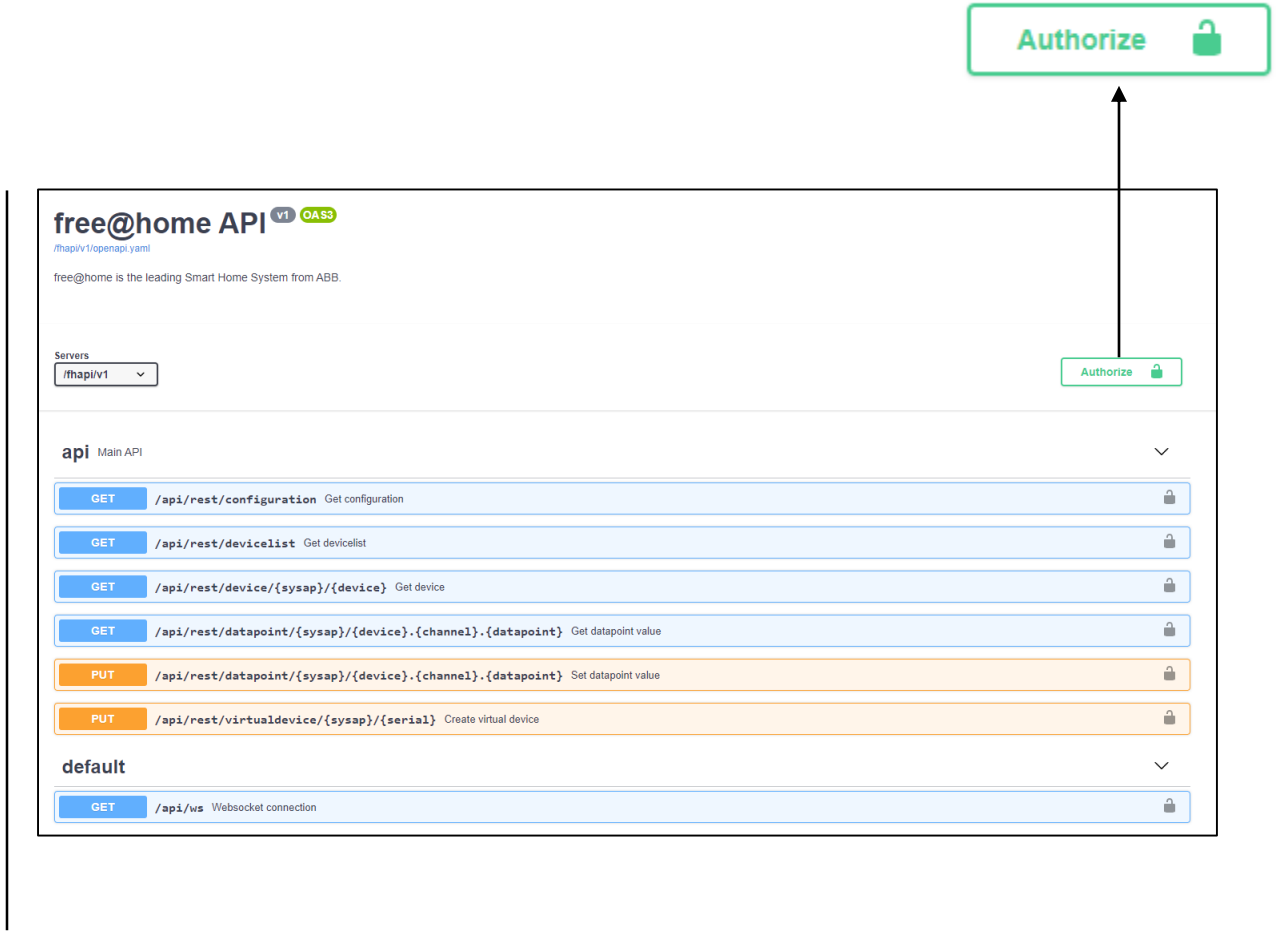


ABB-free@home® – local REST API

Testing the API

Web interface

– Enter the username and the password and select “Authorize” to authorize

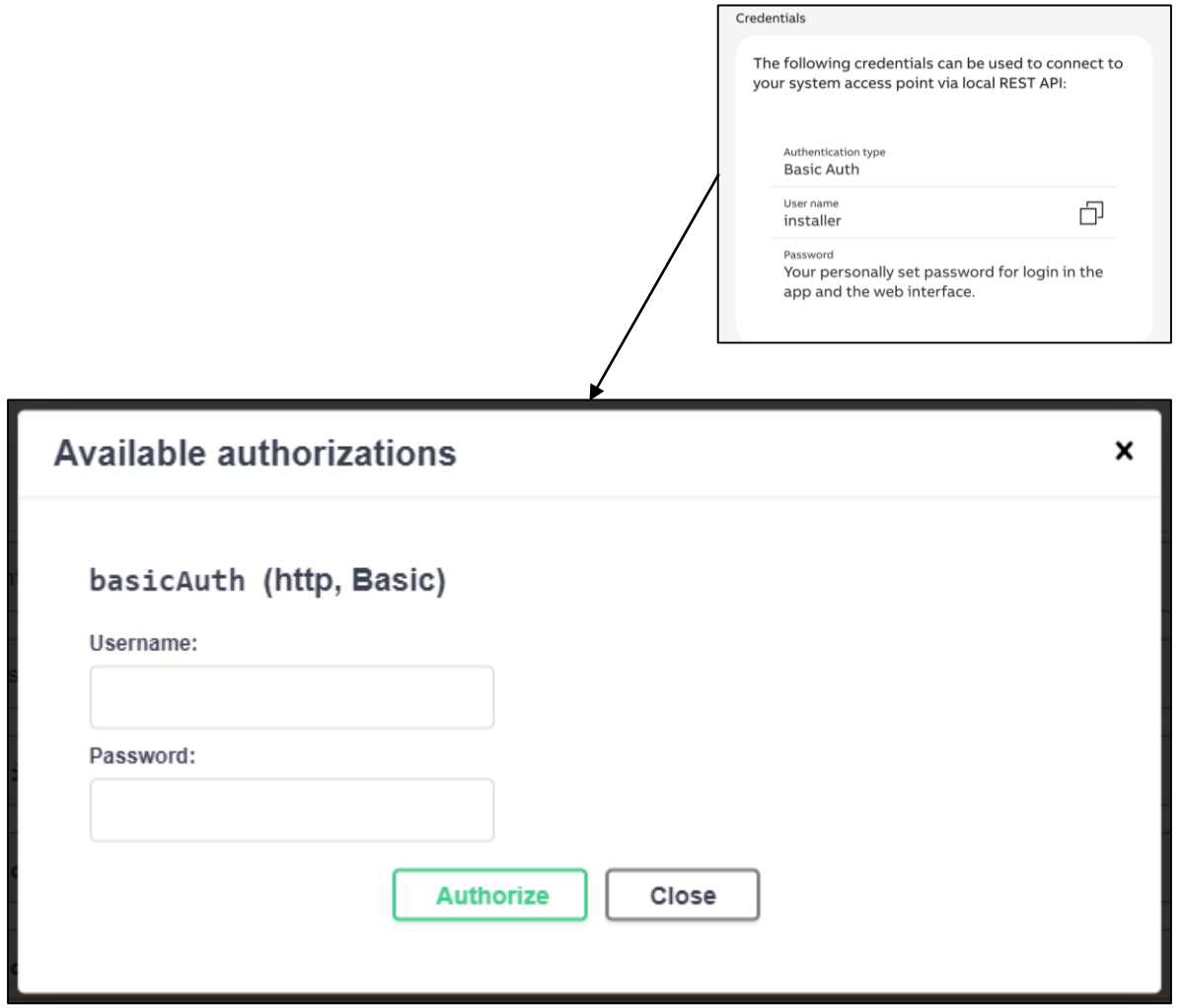


ABB-free@home® – local REST API

Testing the API

Web interface

- The status will be indicated as “Authorized” now
- You can use the available “get” and “put” commands in the web interface now

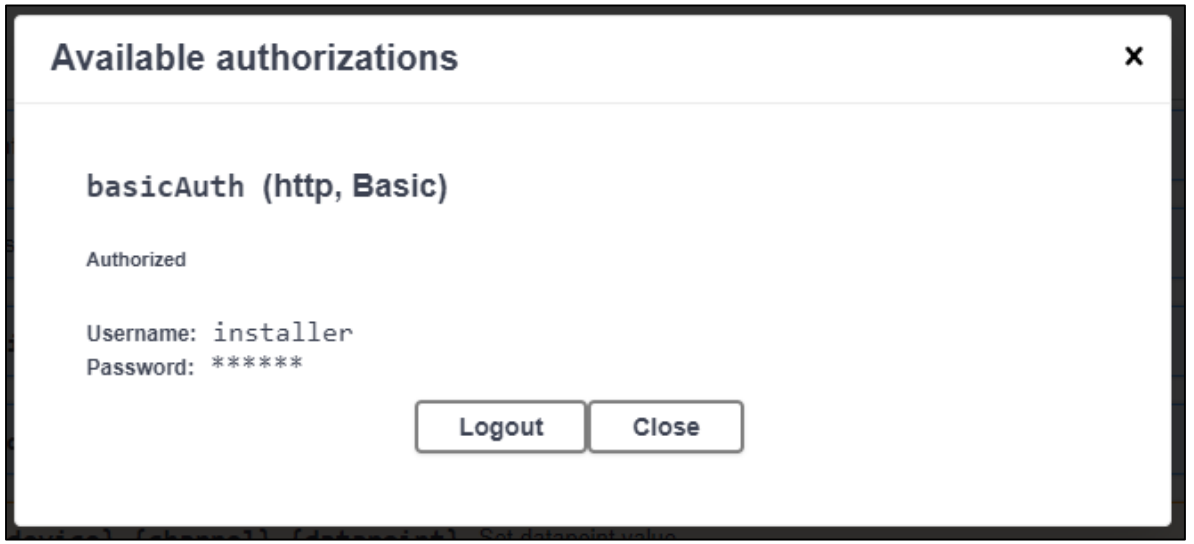


ABB-free@home® – local REST API

Testing the API

Web interface

- The status will be indicated as “Authorized” now
- You can use the available “get” and “put” commands in the web interface now
- To disconnect click again on “Authorize” and “Logout”

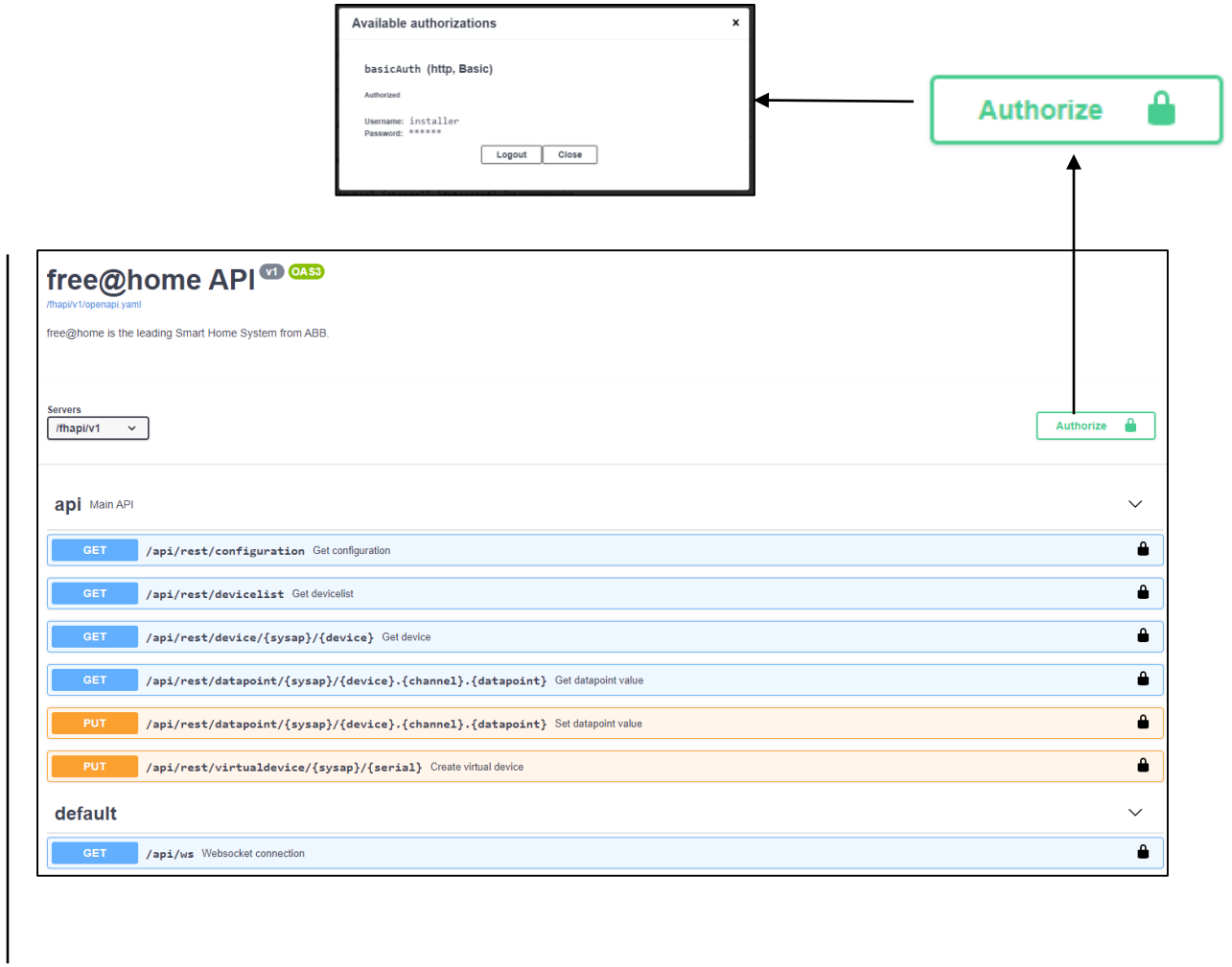


ABB-free@home® – local REST API

Testing the API

Main API – Commands

- "Get configuration": Get configuration for all user registered System Access Points, this includes the schema for all devices, channels and data points, the floorplan and current user information
- "Get devicelist": Get list of devices for all System Access Points
- "Get device": Get configuration information for given device
- "Get datapoint value": Get the current value of a given datapoint
- "Set datapoint value": Set a new value for a given datapoint
- "Create virtual device": Create a virtual device inside free@home

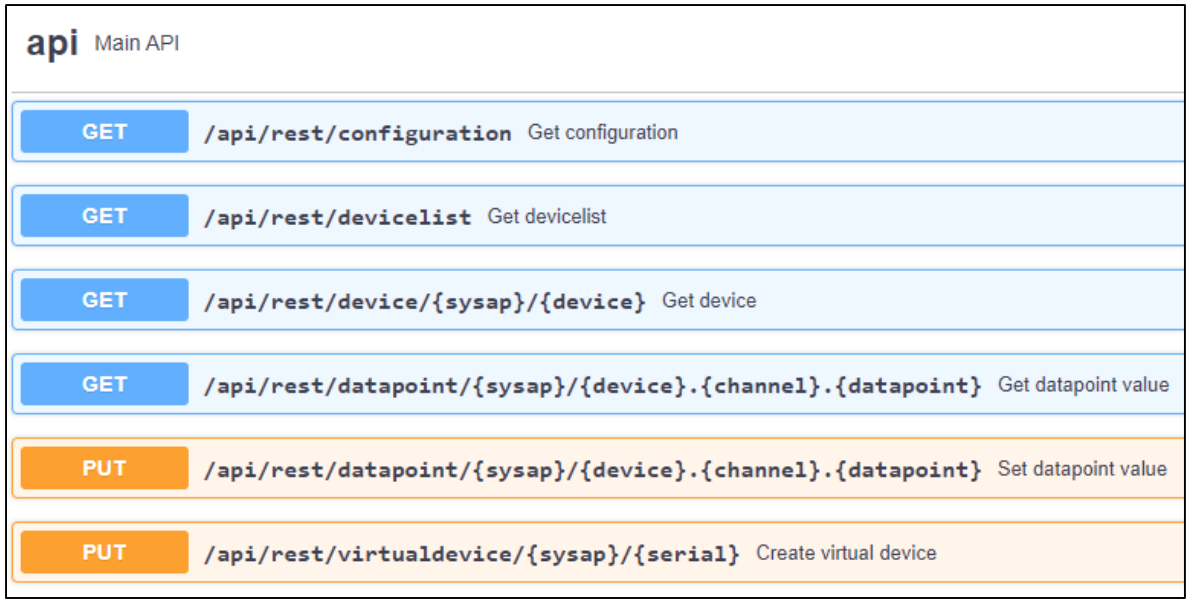


ABB-free@home® – local REST API

Testing the API

Main API – Get configuration

- All functions can be opened by clicking on the tab
- A complete description of the function with all linked parameters and possible responses will be displayed
- Function can be tested by clicking on “Try it out”

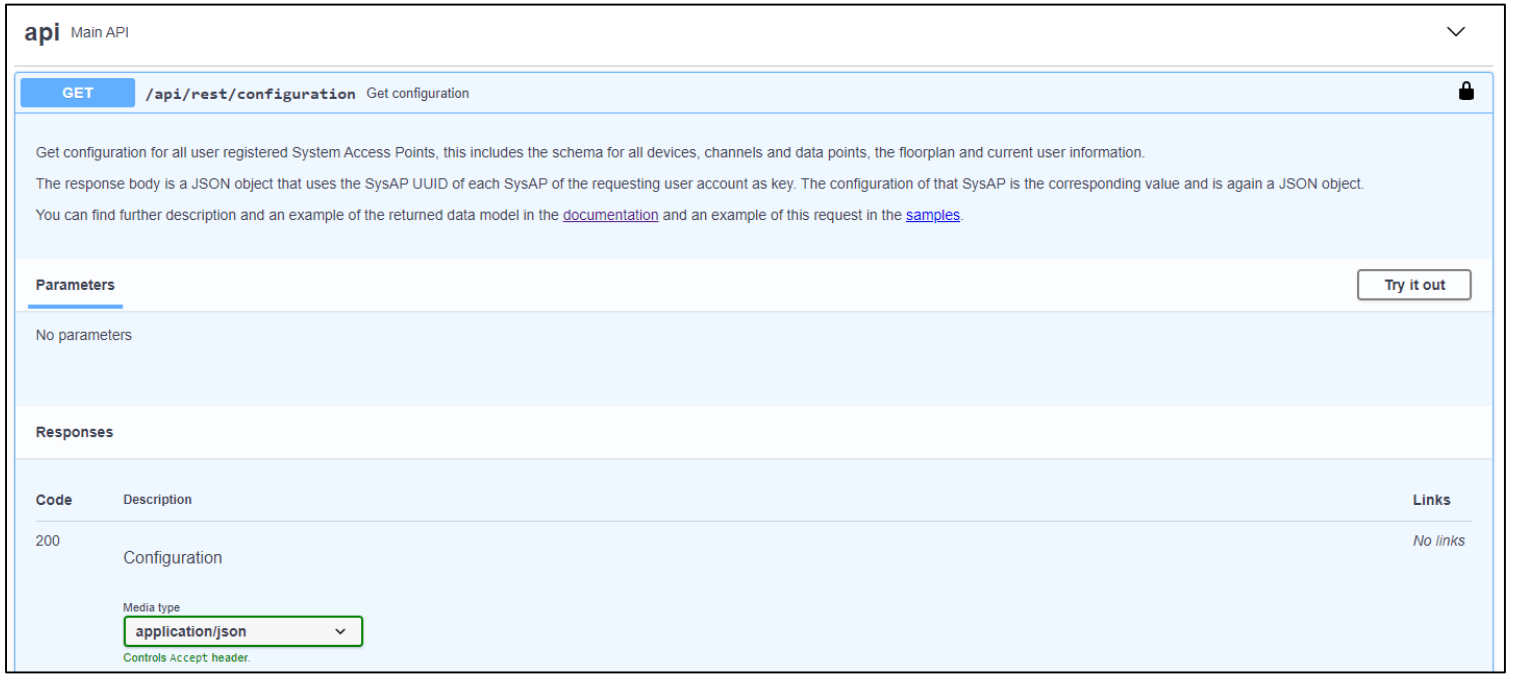


ABB-free@home® – local REST API

Testing the API

Main API – Get configuration

- All functions can be opened by clicking on the tab
- A complete description of the function with all linked parameters and possible responses will be displayed
- Function can be tested by clicking on “Try it out”
- To start the application, click on “Execute”

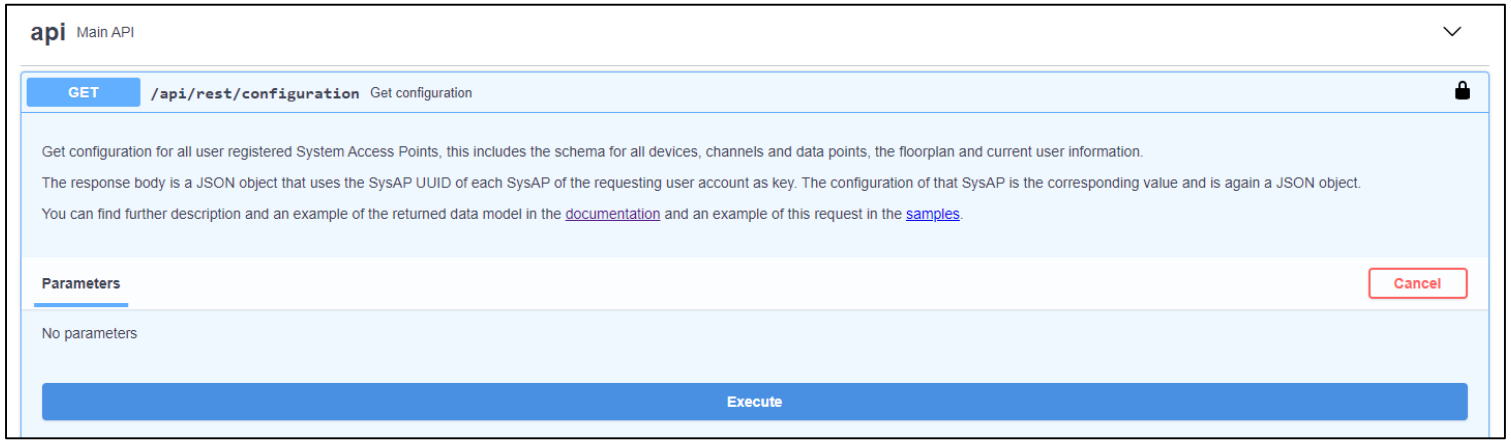


ABB-free@home® – local REST API

Testing the API

Main API – Get configuration

- All functions can be opened by clicking on the tab
- A complete description of the function with all linked parameters and possible responses will be displayed
- Function can be tested by clicking on “Try it out”
- To start the application, click on “Execute”
- The response will be displayed below

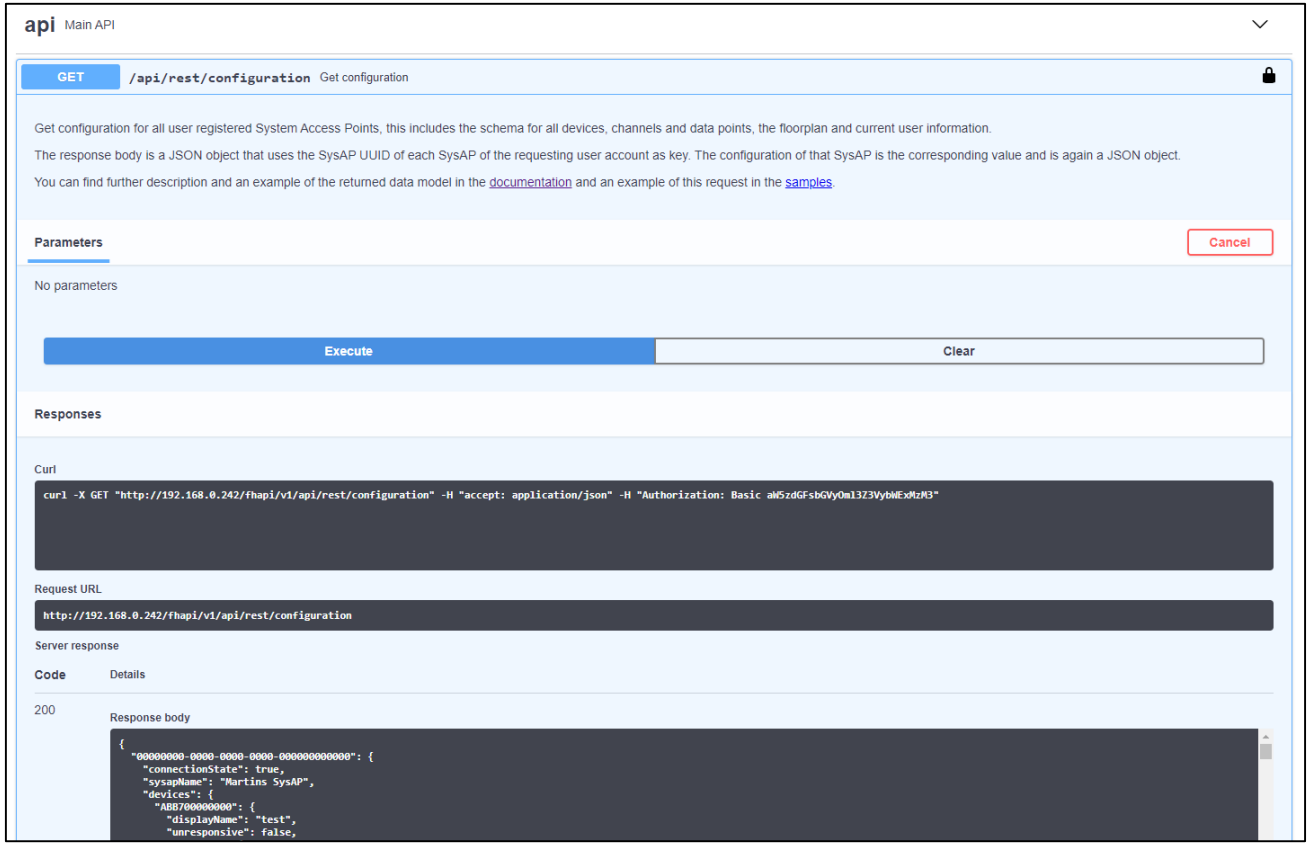


ABB-free@home® – local REST API

Testing the API

Main API – Get configuration

- All functions can be opened by clicking on the tab
- A complete description of the function with all linked parameters and possible responses will be displayed
- Function can be tested by clicking on “Try it out”
- To start the application, click on “Execute”
- The response will be displayed below
- The “Get configuration” function will generate the complete configuration of your free@home system

ABB-free@home® – local REST API

Testing the API

Main API – Get devicelist

- Get list of devices for all System Access Points
- This endpoint is similar to the /api/rest/configuration endpoint, but only provides access to the list of Device IDs that are known by each SysAP, not their corresponding configuration

ABB-free@home® – local REST API

Testing the API

Main API – Get device

- Get configuration information for given device
- This endpoint is similar to the /api/rest/configuration endpoint, but except for returning the full configuration of all SysAPs, it returns the configuration of a single device in a single SysAP only and therefore is much faster and requires less bandwidth
- The response body is a JSON object that maps the (single) SysAP UUID to an object that contains "devices" object (only) which in turn holds the specified device object (only)

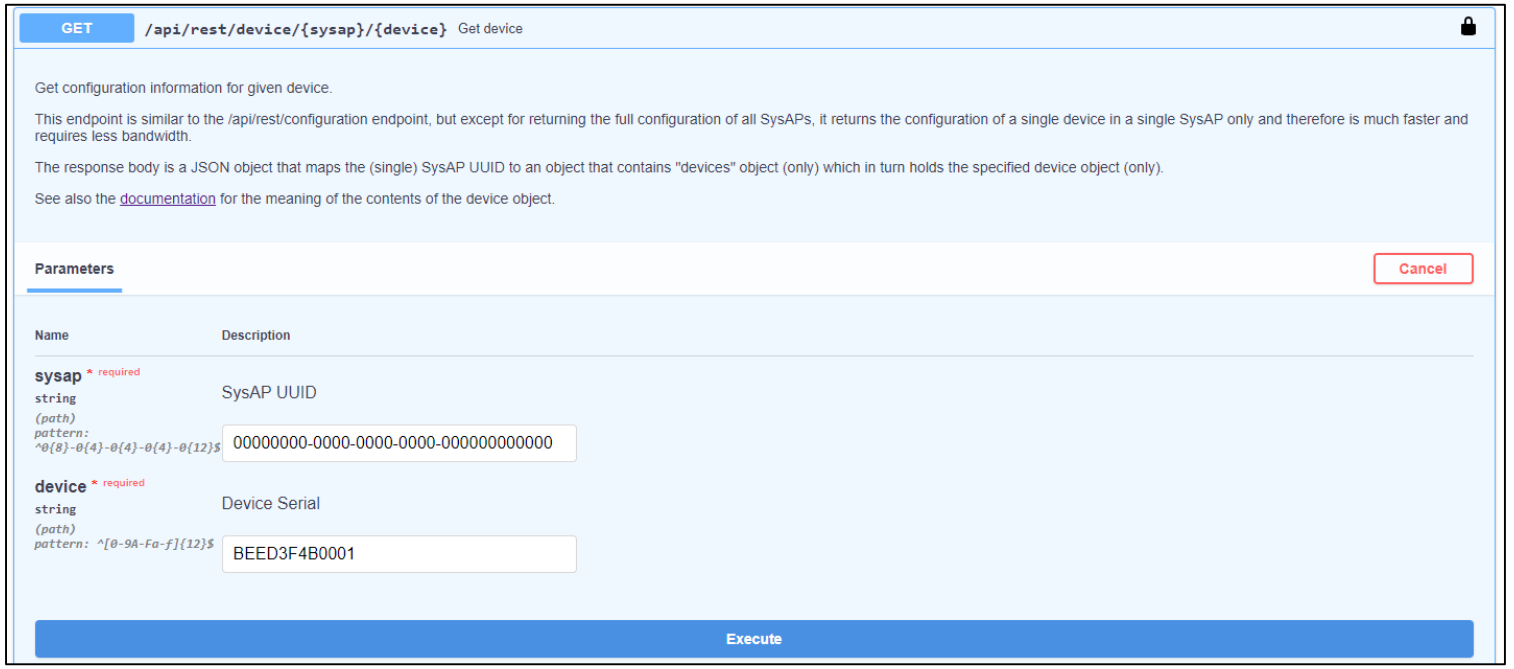
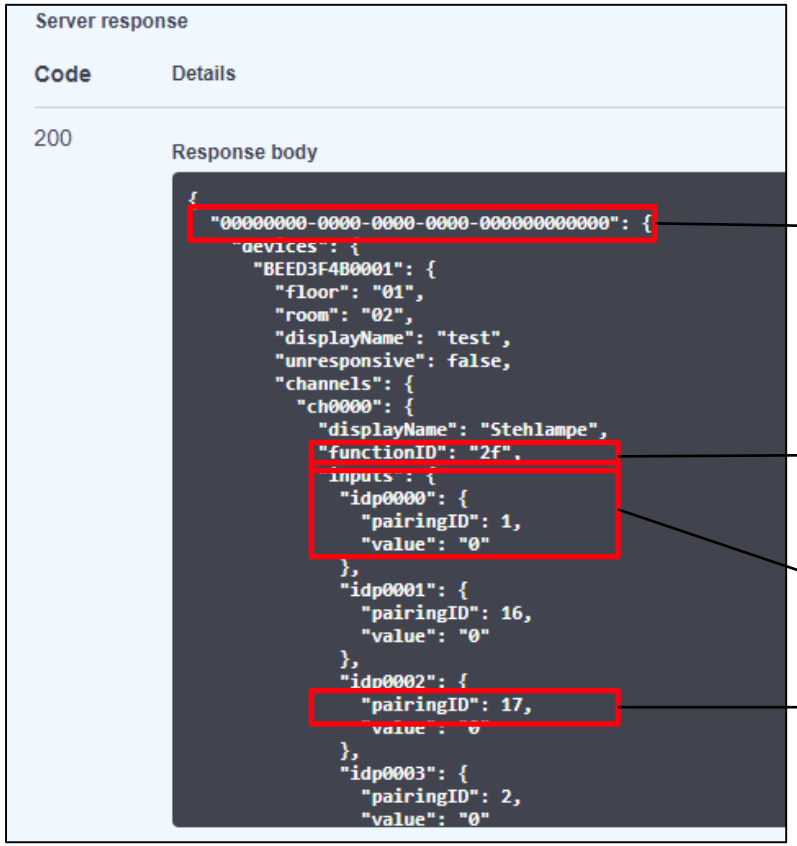


ABB-free@home® – local REST API

Testing the API

Main API – Get device

- Get configuration information for given device
- This endpoint is similar to the /api/rest/configuration endpoint, but except for returning the full configuration of all SysAPs, it returns the configuration of a single device in a single SysAP only and therefore is much faster and requires less bandwidth
- The response body is a JSON object that maps the (single) SysAP UUID to an object that contains "devices" object (only) which in turn holds the specified device object (only)



SSID — Displayed as 000…000, when the local API is used

2f — FID_RGB_ACTUATOR — Dim actuator

1 — in — Switch On/Off — 0 - Off, 1 - On

17 — in — Absolute control of the set value — percent from 0 to 100

Documentation:
https://developer.eu.mybuildings.abb.com/fah_local/concepts/#data-model

ABB-free@home® – local REST API

Testing the API

Comparison API vs. KNX – Where is the device located?

Server response

Code | Details

200

Response body


```
{
  "00000000-0000-0000-0000-000000000000": {
    "devices": {
      "REDACTED": {
        "floor": "01",
        "room": "02",
        "displayName": "test",
        "unresponsive": false,
        "channels": {
          "ch0000": {
            "displayName": "Stehlampe",
            "functionID": "2f",
            "inputs": {
              "idp0000": {
                "pairingID": 1,
                "value": "0"
              },
              "idp0001": {
                "pairingID": 16,
                "value": "0"
              },
              "idp0002": {
                "pairingID": 17,
                "value": "0"
              },
              "idp0003": {
                "pairingID": 2,
                "value": "0"
```

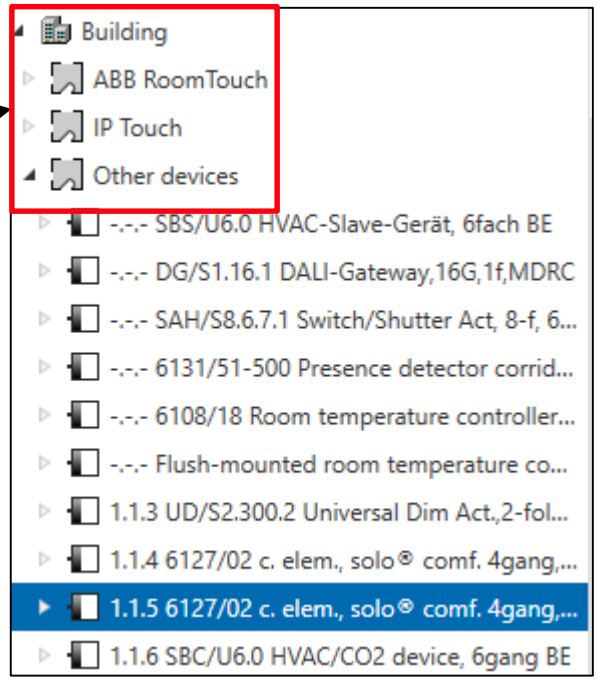



ABB-free@home® – local REST API

Testing the API

Comparison API vs. KNX – How is the device called and what is it?

Server response

Code | Details

200

Response body


```
{
  "00000000-0000-0000-0000-000000000000": {
    "devices": {
      "BEED3F4B0001": {
        "floor": "01",
        "room": "02",
        "displayName": "test",
        "unresponsive": false,
        "channels": {
          "ch0000": {
            "displayName": "Stehlampe",
            "functionID": "2f",
            "inputs": {
              "idp0000": {
                "pairingID": 1,
                "value": "0"
              },
              "idp0001": {
                "pairingID": 16,
                "value": "0"
              },
              "idp0002": {
                "pairingID": 17,
                "value": "0"
              },
              "idp0003": {
                "pairingID": 2,
                "value": "0"
```

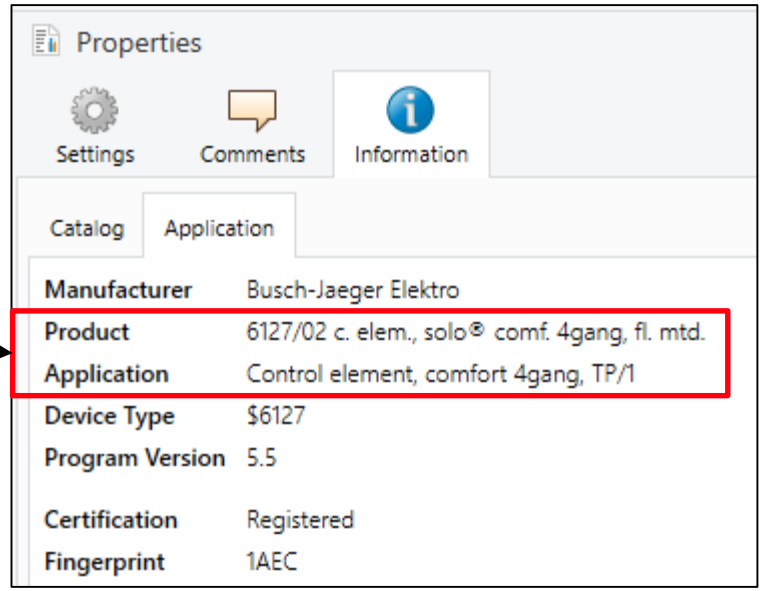



ABB-free@home® – local REST API

Testing the API

Comparison API vs. KNX – Which functions/channels are available?

Server response

Code | Details

200

Response body


```
{
  "00000000-0000-0000-0000-000000000000": {
    "devices": {
      "BEED3F4B0001": {
        "floor": "01",
        "room": "02",
        "displayName": "test",
        "unresponsive": false,
        "channels": {
          "ch0000": {
            "displayName": "Stehlampe",
            "inputs": {
              "idp0000": {
                "pairingID": 1,
                "value": "0"
              },
              "idp0001": {
                "pairingID": 16,
                "value": "0"
              },
              "idp0002": {
                "pairingID": 17,
                "value": "0"
              },
              "idp0003": {
                "pairingID": 2,
                "value": "0"
```


Number	Name	Object Function	Description	Group Address	Length	C	R	W	T	U	Data Type	Priority
1	S1.1: Switching	Input / output			1 bit	C	-	W	T	U	switch	Low
1	S1.1: Switching	Input / output			1 bit	C	-	W	T	U	switch	Low
2	S1.1: Relative dimming	Output			4 bit	C	-	-	T	-	dimming c...	Low
21	S2.1: Switching	Input / output			1 bit	C	-	W	T	U	switch	Low
41	S3.1: Switching	Input / output			1 bit	C	-	W	T	U	switch	Low
61	S4.1: Switching	Input / output			1 bit	C	-	W	T	U	switch	Low
61	S4.1: Value switching	Input / output			1 byte	C	-	W	T	U	percentag...	Low

Data Type

1.* 1-bit

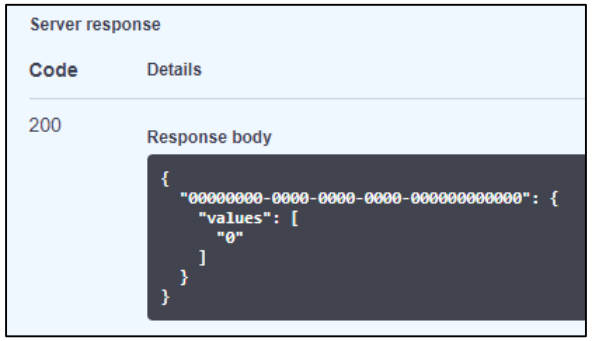
1.001 switch

ABB-free@home® – local REST API

Testing the API

Main API – Get datapoint

– Get the current value of a given datapoint



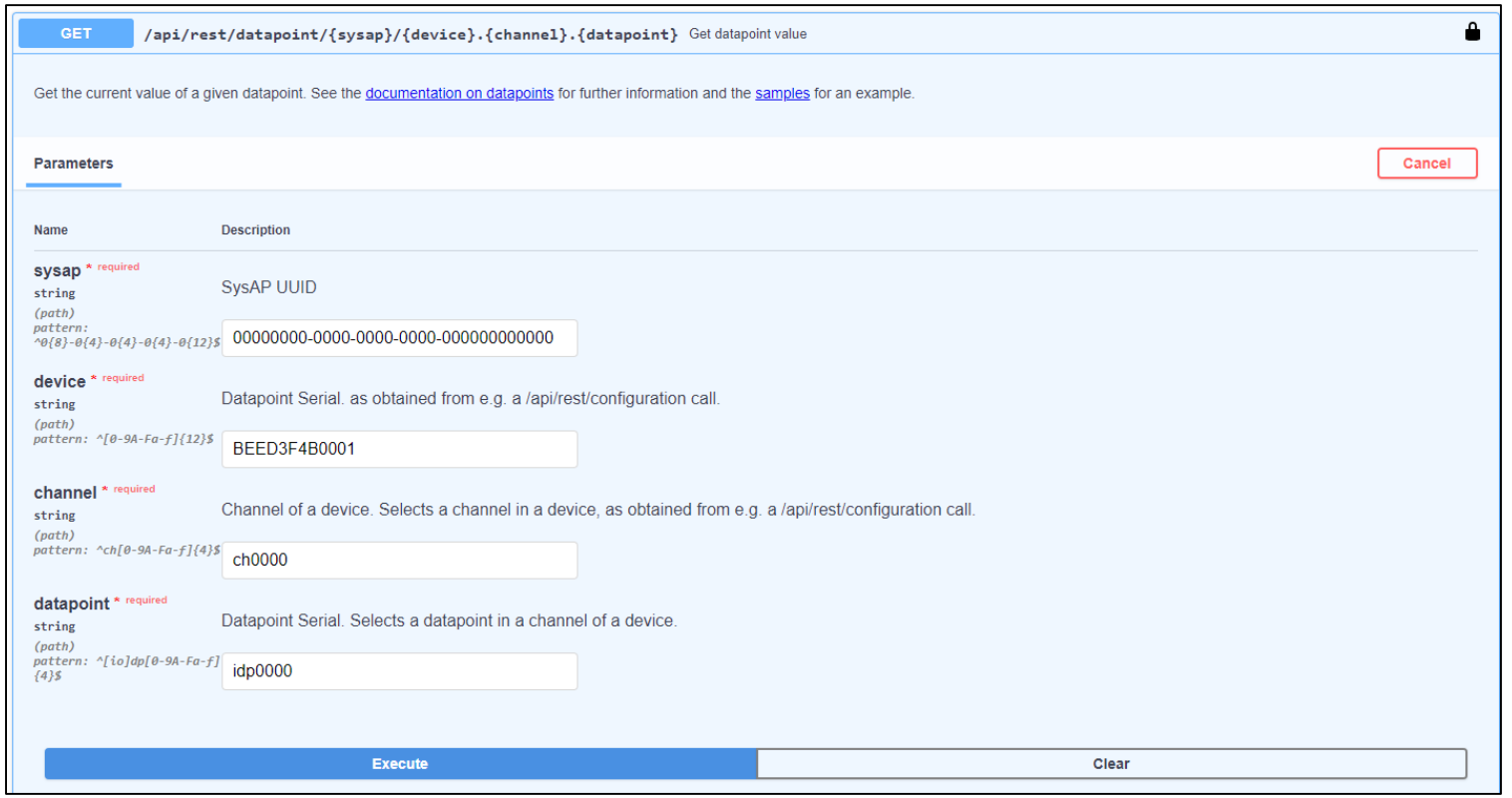


ABB-free@home® – local REST API

Testing the API

Comparison API vs. KNX – How is the device called and what is it?

Curl

```
curl -X GET "http://192.168.0.129/fhapi/v1/api/rest/datapoint/00000000-0000-0000-0000-000000000000/6000D2CB27B2.ch0000.odp0000" -H "accept: application/json"
```

Server response

Code	Details
200	Response body

```
{
  "00000000-0000-0000-0000-000000000000": {
    "values": [
      "0"
    ]
  }
}
```

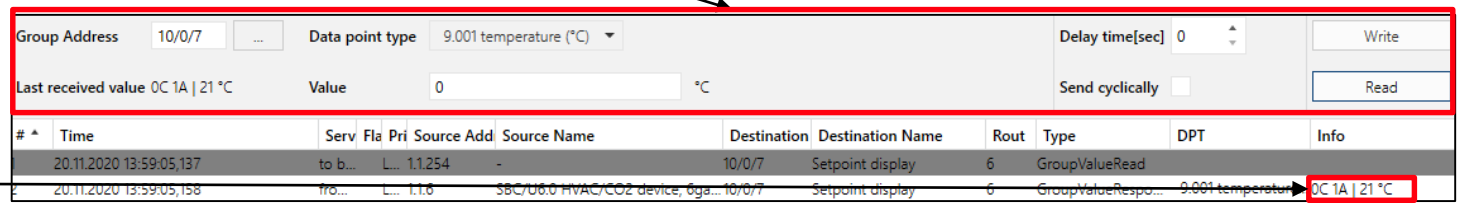


ABB-free@home® – local REST API

Testing the API

Main API – Set datapoint value

– Set a new value for a given datapoint

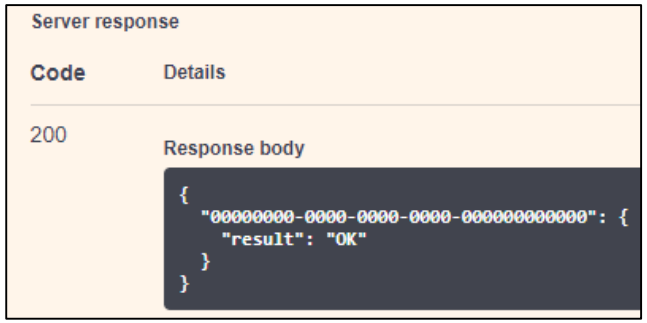


ABB-free@home® – local REST API

Testing the API

Main API – Virtual devices

- Create a virtual device inside free@home
- You can choose a serial number freely

- BinarySensor
- SwitchingActuator
- CeilingFanActuator
- RTC
- DimActuator
- WindowSensor
- ShutterActuator
- WeatherStation
- Weather-TemperatureSensor
- Weather-WindSensor
- Weather-BrightnessSensor
- Weather-RainSensor
- CODetector
- FireDetector

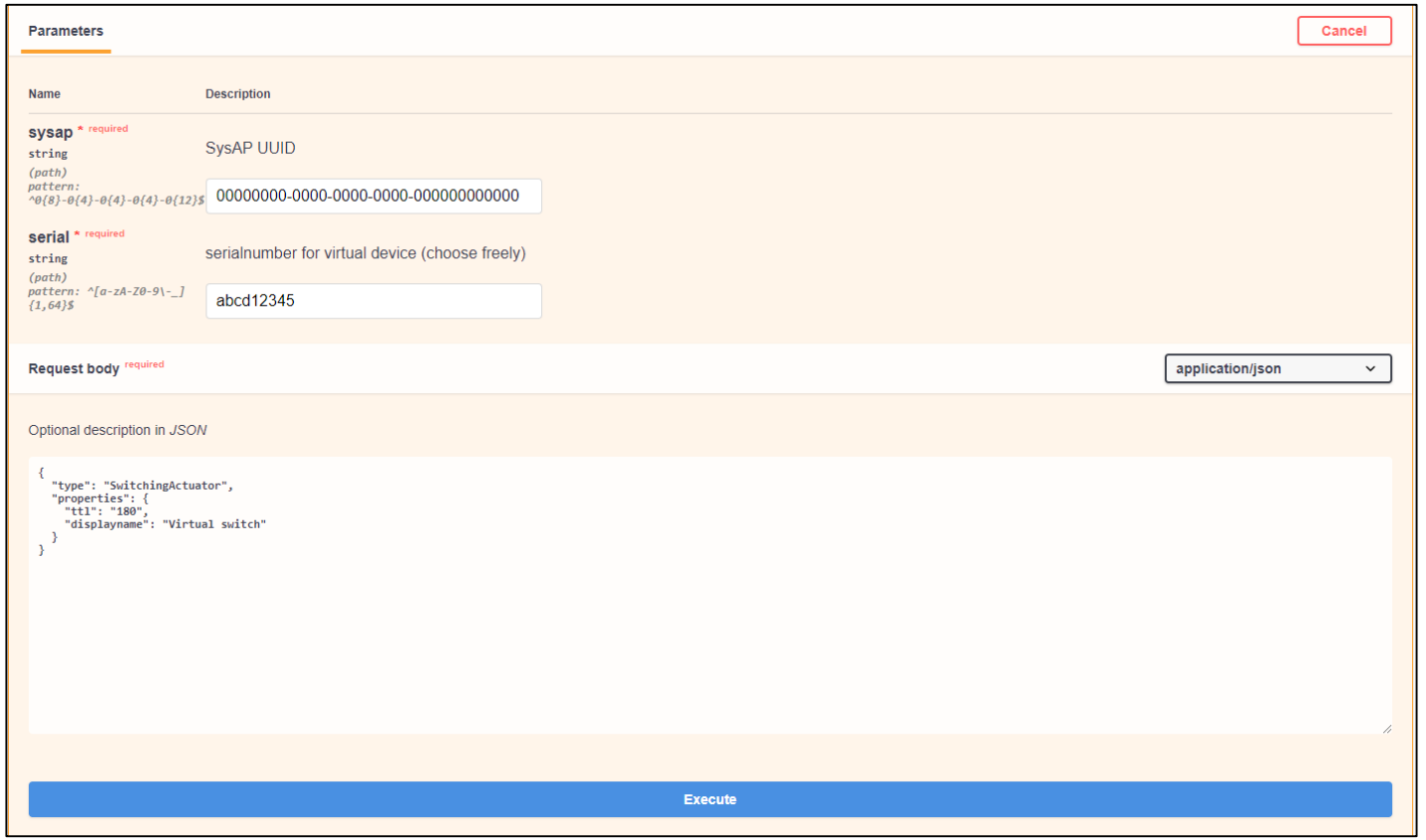


ABB-free@home® – local REST API

Testing the API

Main API – Virtual devices – TTL

- TTL = Time to Life
- Timelife of the virtual device in seconds
- To mark a virtual device as unresponsive (or ready for removal), reregister the device with a TTL value of zero
- To avoid re-registering of the virtual device at regular intervals, the API user can specify a TTL value of -1

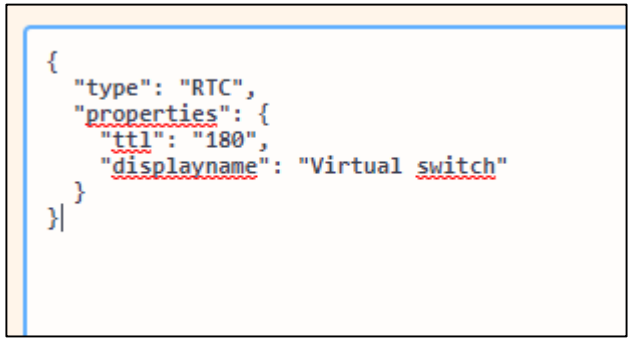


ABB-free@home® – local REST API

Testing the API

Use cases with virtual devices

- You can register a garage door opener in your home automation system as virtual device to control and monitor it from there
- Use virtual switch actuators for internal actions (logic functions)
- Use virtual switch actuators to control 3rd party systems like Gardena or Velux via free@home
- Integrate weather data from Netatmo
- Integrate a nest thermostat into free@home
- Etc.

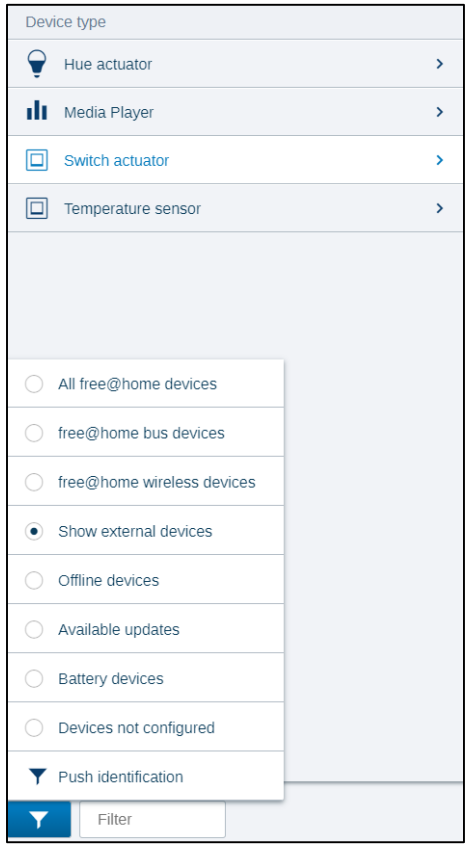


ABB-free@home® – local REST API

Testing the API

Deleting virtual devices via free@home

- Virtual devices can be deleted via the free@home menu
- Device configuration → select the virtual device, which should be deleted → Maintenance → Reset and “Delete external device”

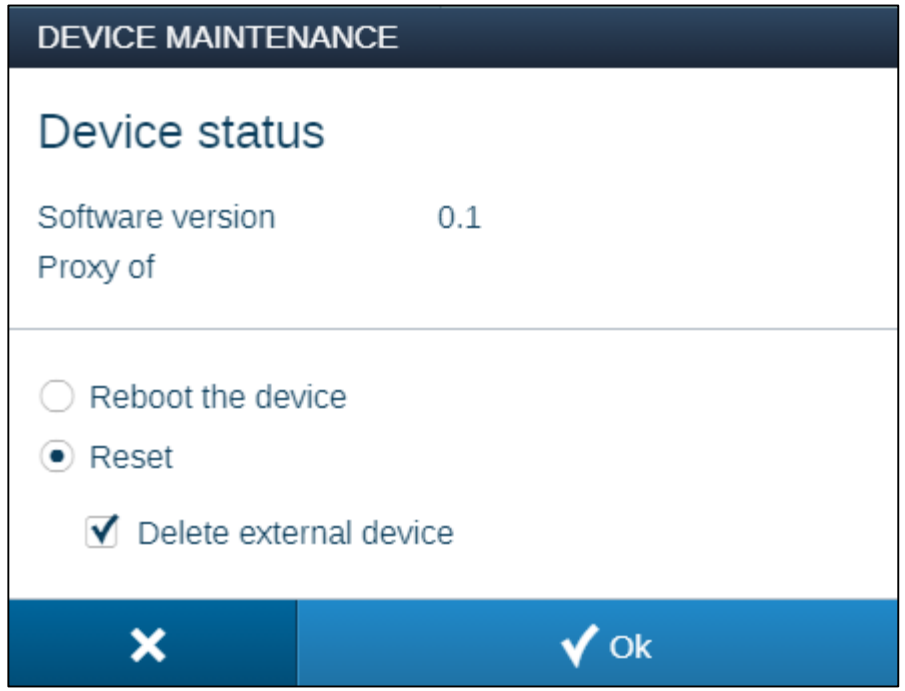


ABB-free@home® – local REST API

Testing the API

WS - Websockets

- WebSocket is a computer communication protocol, providing full-duplex communication channels over a single TCP connection
- Used to push, send and receive data to and from browsers (example: chat applications like Telegram, Whatsapp, etc.)
- Here we use websockets to send and receive data to and from the System Access Point

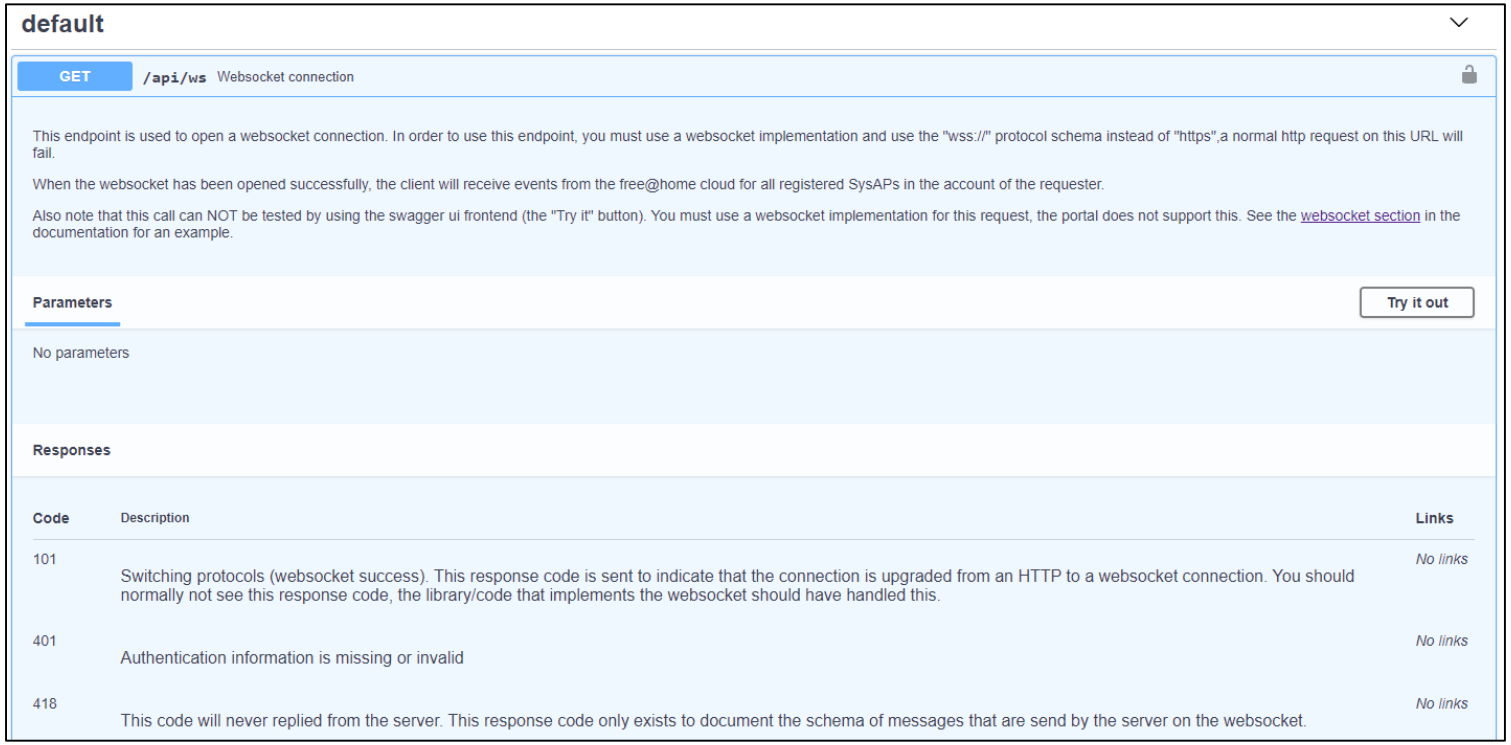


ABB-free@home® – local REST API

Practical Examples

ABB-free@home® – local REST API

Installation

Preparation

– The following devices are needed to work with the free@home API:
  - System Access Point 2.0 SAP/S.3 (FW: 2.6.0. or higher)
  - Raspberry Pi + Power Supply + Housing
  - SD card with 4GB or more
  - Windows/MAC/Linux PC with SD card reader

ABB-free@home® – local REST API

Practical Examples

Integration free@home in Apple Home(Kit)

– free@home can be integrated into Apple Home Kit with the local REST API

ABB-free@home® – local REST API

Practical Examples

Integration free@home in Apple Home(Kit)

- Setup your Raspberry Pi and install the Homebridge application
- Open http://homebridge.local or http://<ip address of your server>

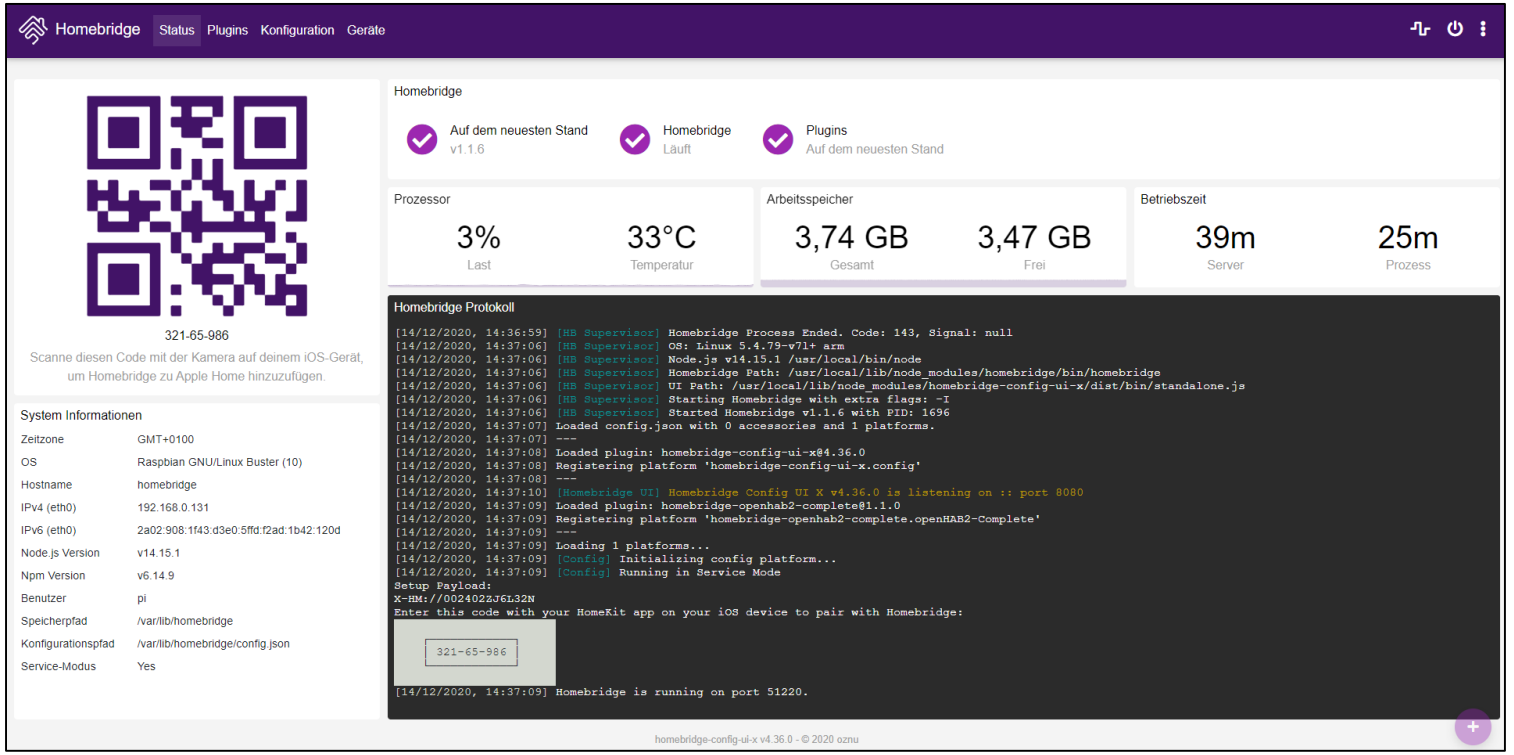
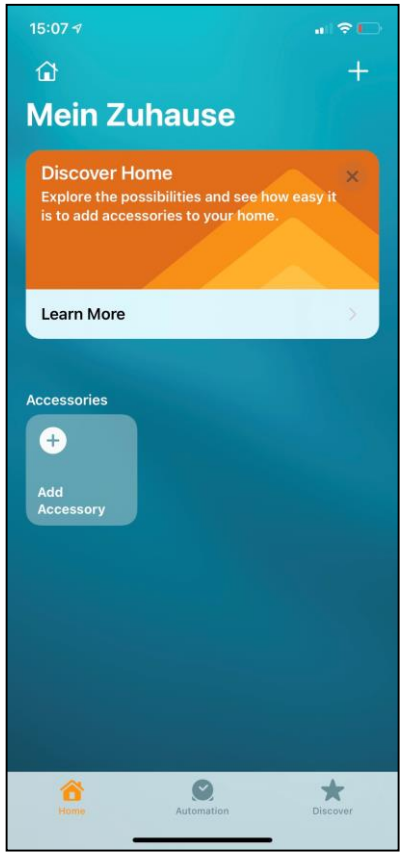


ABB-free@home® – local REST API

Practical Examples

Integration free@home in Apple Home(Kit)

- Open the Home app on your mobile device
- Tap the Home tab, then tap ⊕
- Add Accessories and scan the QR code displayed in the Homebridge user interface
- Follow the installation wizard to finish the integration



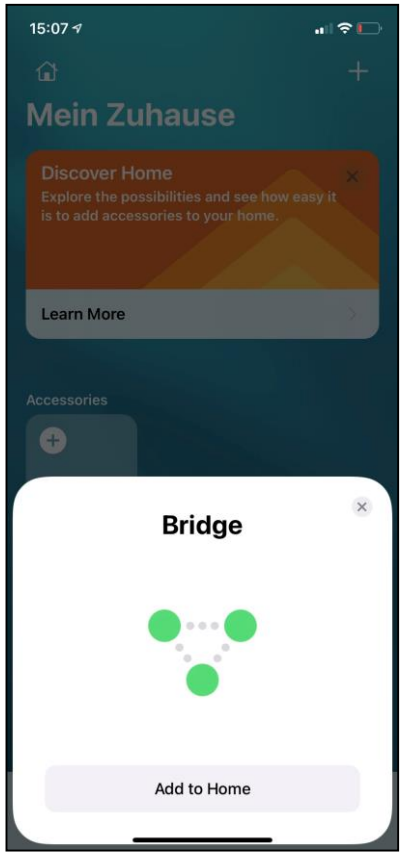


ABB-free@home® – local REST API

Practical Examples

Integration free@home in Apple Home(Kit)

- Open the Homebridge interface again and navigate to “Plugins”
- Search for “freeathome” and install the plugin
- Navigate to “configuration”
- Edit the config.json with your personal credentials

```
{
    "platform": "free@home",
    "sysIP": "<IP>",
    "username": "<USERNAME>",
    "password": "<PASSWORD>",
    "mappings": {}
}
```

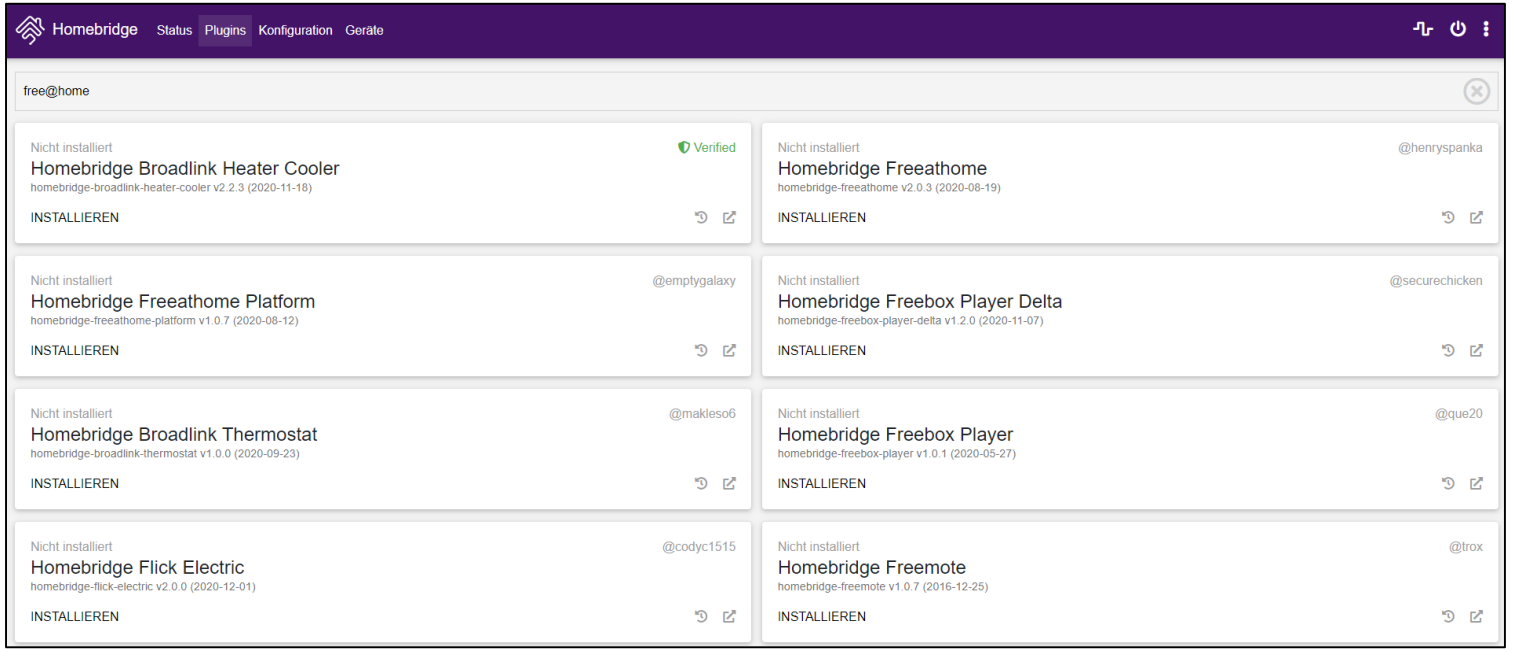
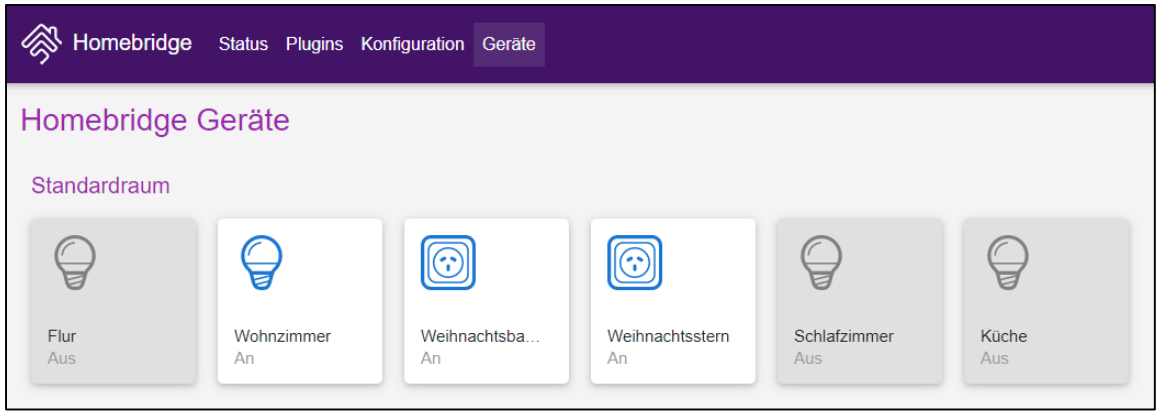


ABB-free@home® – local REST API

Practical Examples

Integration free@home in Apple Home(Kit)

– The free@home devices should appear in your Home app now (this may take some minutes after the restart)



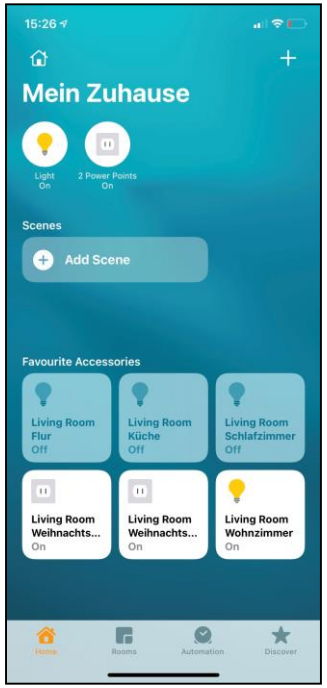


ABB-free@home® – local REST API

Practical Examples

Netatmo integration

- In this example an open hardware platform like a raspberry pi is taken and equipped with open-source software like „Node-RED“
- Inside free@home a virtual weather station is created which appear in free@home, but is not belonging to any physical auxiliary device
- Now the data from a 3rd party weather station is injected to the virtual weather station inside free@home
- On the righthand side you can see the data inside free@home next app. It appears as native weather station data
- Precondition: IoT Hub set up and running and a SysAP 2.0 with activated API

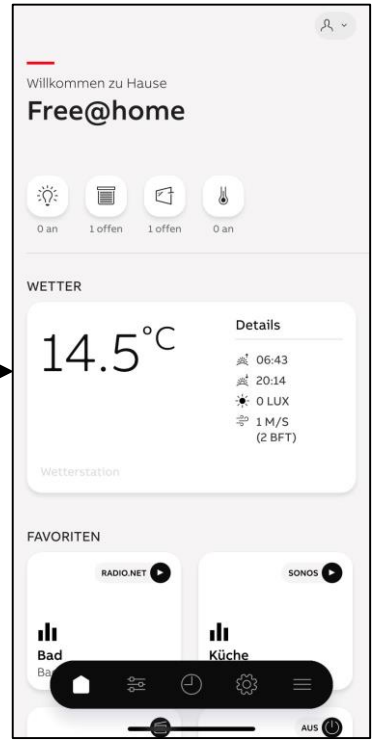


ABB-free@home® – local REST API

Practical Examples

Setup of the IoT Hub

- To integrate a 3rd party system to free@home an IoT Hub must be installed on your Raspberry Pi
- Possible IoT Hubs:
  - OpenHAB2: open IoT hub to network several systems and services with each other (free@home binding already included)
  - Node-RED: visual IoT logic editor for connecting several systems with the help of "flows“
  - Grafana: open web visualization to be able to easily build dashboards
  - influxDB: Database in the background to persist all values in the smart home

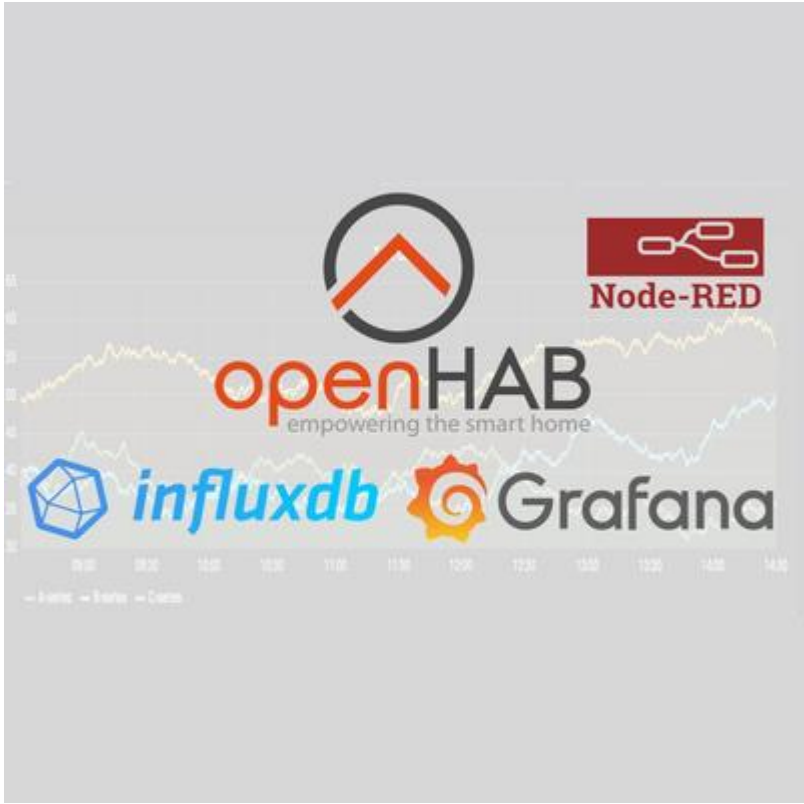


ABB-free@home® – local REST API

Practical Examples

Linking free@home and Netatmo

- 1. Integrate free@home in openHAB
- 2. Integrate Netatmo in openHAB
- 3. Create a virtual weather station in free@home (swagger interface)
- 4. Linking Netatmo and free@home via Node-RED

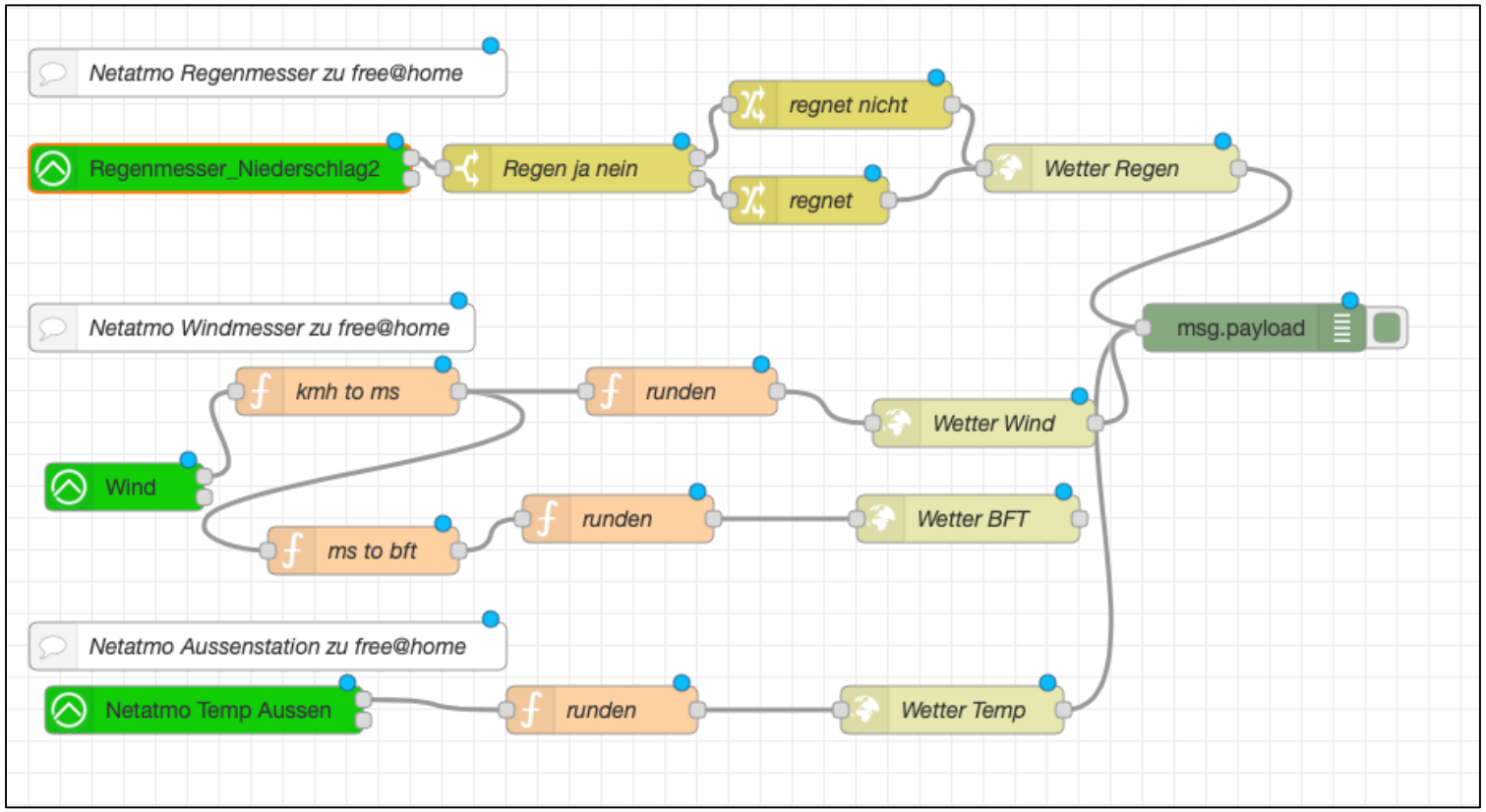


ABB-free@home® – local REST API

Practical Examples

Visualize data from free@home

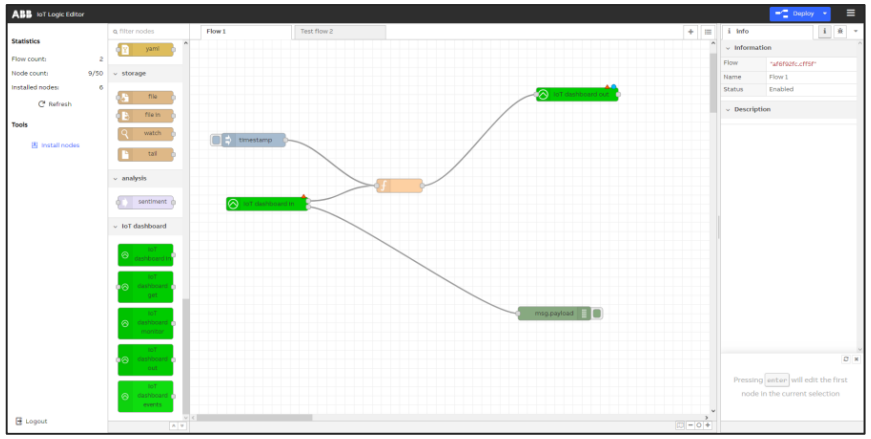


ABB-free@home® – local REST API

Practical Examples

Visualize data from free@home

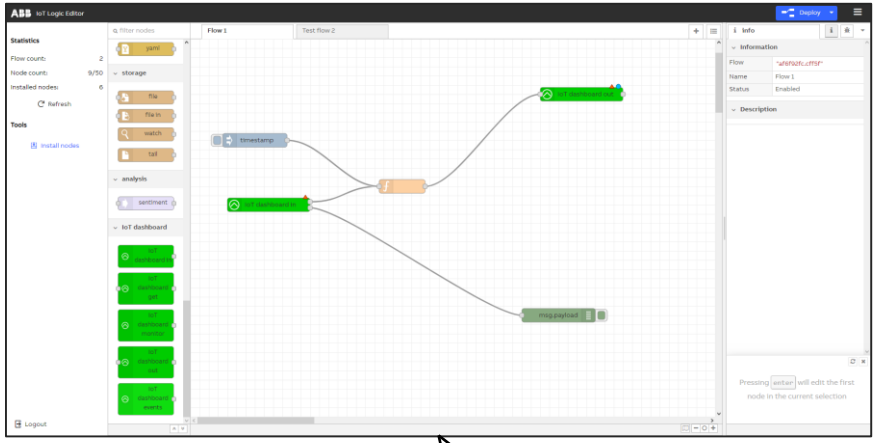
- In this example an open HW platform like a raspberry pi is taken an equipped with open-source software like „Node-RED, InfluxDB and Grafana
- This gives the possibility to easily gather and persist all data coming from the free@home system and to visualize it

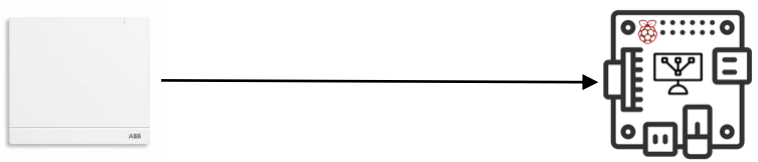
ABB-free@home® – local REST API

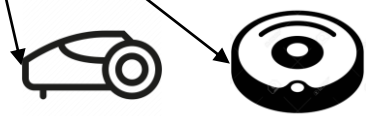
Practical Examples

Smart robot mower integration

– In this example a smart robot mower can be started with virtual devices in the free@home interface







Disclaimer

Technical data in this presentation are only approximate figures. The information in this presentation is subject to change without notice and should not be construed as a commitment by ABB. ABB assumes no responsibility for any errors that may appear in this presentation.

ABB shall in no case be liable under, or in connection with the presentation towards any person or entity, to which the presentation has been made available, in view of any damages or losses – irrespective of the legal grounds. In particular ABB shall in no event be liable for any indirect, consequential or special damages, such as - but not limited to – loss of profit, loss of revenue, loss of earnings, cost of capital or cost connected with an interruption of business.


ABB