

ROBERT W.B. LINN

# Workbook Home Automation

---

Exploring openHABian & openHAB2

By Robert W.B. Linn, Pinneberg, Germany

24.08.2018

THIS DOCUMENT IS PROVIDED BY THE AUTHOR ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

# Table of Contents

Table of Contents .....	1
Objectives .....	6
Screen shots Basic UI and Setup .....	7
Getting Started .....	8
Hardware .....	8
Home Automation .....	8
volkszaehler .....	8
Service .....	8
Software .....	8
Home Automation .....	8
volkszaehler .....	8
Prepare .....	9
RFXCOM RFXtrx433E Transceiver .....	9
Raspberry Pi .....	10
USB Symlink .....	10
WLAN Static IP .....	12
WLAN Power Save Mode .....	13
Ethernet Link volkszaehler Raspberry Pi .....	14
Firmware Update .....	14
openHAB Bindings .....	16
RFXCOM Binding .....	17
Notes .....	17
Binding Installation .....	18
Thing USB Transceiver Add & Configure .....	18
Somfy Blinds RTS Motor Manual Configuration .....	19
Thing Rfy Actuator Configuration .....	20
Thing Temperature_Humidity (TFA Dostmann TS34C) .....	22
Thing Wind (TFA Dostmann 30.3168) .....	24
Thing FrontDoor-Security (PB62R) .....	25
Weather Binding (OpenWeatherMap Service) .....	29
Binding Installation .....	29
Binding Configuration .....	29
Item Configuration .....	30
Weather Binding (Weather Underground Service) .....	32
Binding Configuration .....	32

Thing Configuration .....	32
Item Configuration .....	32
Astro Binding .....	33
Thing Astro moon data & Sun data.....	33
Item Configuration .....	33
MQTT Binding.....	34
Binding Installation .....	34
Install Mosquitto Broker .....	34
Install openHAB MQTT Action .....	34
Services Configuration .....	34
Hue Binding .....	35
Hue Bridge Installation .....	35
Binding Installation .....	35
Thing Hue Bridge.....	35
Thing Hue White Lamp.....	36
Thing Hue Adding More Lamps .....	36
Thing Log Examples .....	37
Hue Light Control Examples.....	38
Basic UI - Home Automation.....	44
Objectives.....	44
Configuration Setup .....	44
Development .....	44
Configuration Files.....	46
Naming Conventions.....	47
Custom Icons .....	48
Layout Concept .....	49
Sitemap Access .....	49
Parameter.....	50
Persistence Configuration .....	51
Charts rrd4j.....	51
Transform Configuration.....	53
Scale Winddirection .....	53
Scale Windspeedmsbft .....	54
Node-RED as Rules Engine .....	55
Overview .....	55
Update Node-RED .....	56
Node-RED openHAB2 Nodes .....	57

Define the openHAB2 Controller Node .....	57
Logging State Change MakeLab HueLight1 .....	58
Node-RED as MQTT Publisher .....	59
Dashboard UI .....	60
Functions Introduction .....	66
Function Weather .....	67
Goal .....	67
Items Configuration.....	68
Sitemap Configuration .....	68
Function Astro Information.....	69
Goal .....	69
Item Configuration .....	69
Transformation Configuration.....	70
Sitemap Configuration .....	71
Chart Configuration.....	72
Function Security Front Door .....	73
Goal .....	73
Function Power Consumption.....	74
Goal .....	74
Items Configuration.....	75
Sitemap Configuration .....	75
Persistence Configuration .....	75
Rules Configuration .....	75
Notes.....	77
Function Charts rrd4j .....	78
Goal .....	78
Persistence Configuration .....	78
Items Configuration.....	79
Sitemap Configuration .....	79
Screen Shots .....	79
Delete Chart Data .....	80
Selective Charts.....	80
Period Values .....	81
Refresh Period .....	81
Display Chart in Web Browser .....	81
Function MQTT .....	82
Goal .....	82

Overview Topics Published .....	82
Message Energy Power Consumption.....	83
Message Security Front Door State.....	84
Message Security Front Door Reset.....	86
Message Security Front Door Notify .....	88
Message HueLight1 Brightness .....	89
Function Homepage Counter.....	90
Goal .....	90
Concept.....	90
Items Configuration.....	90
Sitemap Configuration .....	90
Rules Configuration .....	91
Function Raspberry Pi System Information .....	92
Goal .....	92
Concept.....	92
Measurement & Action .....	93
Items Configuration.....	94
Sitemap Configuration .....	94
Rules Configuration .....	94
Bash Script for Tests .....	96
Function Hue .....	97
Goal .....	97
Items Configuration.....	98
Sitemap Configuration .....	99
Rules Configuration .....	100
Function Anemometer.....	101
Goal .....	101
Items Configuration.....	101
Sitemap Configuration .....	102
Rules Configuration .....	102
Transformation Configuration.....	102
Function Waste Calendar .....	104
Goal .....	104
Items Configuration.....	105
Sitemap Configuration .....	105
Rules Configuration .....	106
Concept.....	107

Python Script .....	108
Bash Script .....	110
Appendix: Setup volkszaehler.....	111
Hardware .....	111
Software .....	111
Setup Raspberry Pi .....	111
Option 1 Raspberry Pi WLAN Network fixed IP address .....	111
Option 2 Raspberry Pi ETH Network fixed IP address .....	112
volkszaehler.....	112
PowerMeter Setup .....	112
USB Configuration.....	113
Define Channels .....	114
vzlogger Setup.....	114
vzlogger Service .....	115
Appendix: Test Basic UI .....	118
Objectives.....	118
Items Configuration.....	118
Sitemap Configuration .....	119
Rules Configuration .....	119
Check Configuration .....	120
Appendix: openHAB Hints .....	121
Restart.....	121
Clean-up Cache and Temp files .....	121
Create a Backup .....	121
Clear the Paper UI Inbox.....	122
Appendix: To-Do-List.....	123

# Objectives

To build a Home Automation solution, running on a Raspberry Pi with openHABian and openHAB2.

## *As an openHAB Beginner*

- ✓ The main goal is to learn openHABian & openHAB2.
- ✓ Get a Home Automation solution up & running and extend.
- ✓ There might be better solutions for things shared – but the solution works fine so far.
- ✓ Getting motivated to develop more with openHAB.

## *Functions*

- Display temperature & humidity measured in living room, basement & garage.
- Display weather information obtained from the OpenWeatherMap service.
- Charts for selective weather items, room temperature & humidity.
- Display Astro data sun rise, sun set, daylight duration, moon phase and other info.
- Control Somfy roller shutters with RTS motors in living room and bed room.
- Philips Hue Lighting System control via Hue Bridge for ZigBee devices.
- Security door & window wireless contact detectors.
- Energy Power Consumption metering from “volkszaehler” with charts.
- Homepage counter with chart.
- Raspberry Pi system information with charts and threshold email notification.
- MQTT subscribe & publish messages to trigger actions or information.
- *Add more features whilst developing ...*

## *Explore How To*

- Setup & configure openHABian & openHAB2.
- Use the RFXCOM RFXtrx USB RF Transceiver (RFXtrx433E) for
  - Temperature & Humidity devices.
  - External Wind device (only for RFXCOM tests).
  - Other 433Mhz devices, i.e. door contact
- Use external services, i.e. OpenWeatherMap.
- Use actions, i.e. MQTT messaging.
- Use bindings like MQTT, Astro, Philips Hue.
- Use Node-RED as an alternative rules engine.
- Use the openHAB Android App (native client).
- Create advanced User Interfaces, i.e. HABPanel & Node-RED.

## *Approach*

- Setting up the Raspberry Pi, RFXCOM with connected devices, Somfy roller shutters and other devices. Use the Paper UI for Binding, Things, Channel configuration.
- Create the Basic UI textual configuration files including rrd4j charts & MQTT.
- Planned UI's HABPanel & Node-RED (later stage, not included in this document)

# Screen shots Basic UI and Setup

*Notes:* Pictures from beginning of the project – in the meantime the setup has been evolved.

**Weather**

Temperature	5.0 °C	Humidity	80 %
Pressure	1010 hPa	Wind Direction	ESE
Windspeed Beaufort	1	Home	>
Lanzarote	>	Charts	>

**Home Details**

Observation time	10.01.2018 14:20	Temperature	5.0 °C
Temperature feel	3.3 °C	Humidity	80 %
Pressure	1010 hPa	Wind Direction	ESE
Wind Speed	5.40 km/h	Windspeed Beaufort	1
Clouds	90 %	UV Index	
Ozone	- ppm	Condition	overcast clouds
Delivered by OpenWeatherMap			

**Charts Home**

Period: 4H **DAY** WEEK MONTH YEAR

**Rooms, Security, Info**

**Living Room**

Temperature: 18.0 °C Humidity: 47 %

Blind: ^ X v Charts >

**Bed Room**

Temperature: 10.4 °C Humidity: 66 %

Shutter: ^ X v

**Basement**

Temperature: 13.6 °C Humidity: 57 %

**Security**

Front Door Security >

**Energy**

Power Consumption: 428 kWh Power Consumption Charts >

Info >

**Prototype - Setup**

**openHAB User Interfaces**

- openHAB Paper UI HTML5 web application
- openHAB Basic UI web interface
- openHAB Android Application

**Prototype – Setup Raspberry Pi Home Automation & volkszaehler**



# Getting Started

Hardware, service and software used for this openHAB project.

## Hardware

### Home Automation

- 1x Raspberry Pi 3 Model B v1.2
- 1x RFXCOM RFXtrx433E USB 433.92MHz Transceiver
- 3x TFA Dostmann TS34C Temperature & Humidity device
- 1x TFA Dostmann 30.3168 Wind Meter (speed, direction, temperature) [Test only]
- 2x Somfy Blinds RTS Pure
- 1x PB-62R Door & window wireless contact detector

### volkszaehler

- 1x Raspberry Pi 2 Model B (volkszaehler server)
- 1x Power Meter EMH ED300L.
- 1x volkszaehler IR-write-read-device

## Service

- OpenWeatherMap

## Software

### Home Automation

- openHABian 1.4 release Raspbian GNU/Linux 8 (jessie) with openHAB 2.2
- RFXCOM RFXflash Programmer 8.0.0.0 - to update the firmware on the RFXtrx433E
- RFXCOM RFXmng 18.0.0.18 - to test and manage RFXtrx433E connected devices
- VSCode - for developing the openHAB textual configuration files
- WinSCP - to exchange files between development device and the Raspberry Pi
- PuTTY - to run commands from a terminal session

### volkszaehler

- Raspbian GNU/Linux 8 (jessie)
- Volkszähler image (2015-11-11)

# Prepare

Various preparation steps prior developing the solution in openHAB.

## RFXCOM RFXtrx433E Transceiver

Prepare the RFXtrx433E device:

- Install the utility programs RFXflash and RFXmngr on a Windows PC.
- Connect the RFXtrx433E to an USB port.
- Flash the latest firmware (RFXtrx433\_Ext2\_Firmware) as described in the [RFXtrx User Guide](#).
- Open the RFXmngr, connect to the RFXtrx433E, obtain status information and log incoming messages for devices found.

### Example RFXmngr Log

```
Get Status
-----
Packettype      = Interface Message
subtype         = Interface Response
Sequence nbr    = 1
response on cmd = Get Status
Transceiver type = 433.92MHz
Firmware version = 1022
Firmware Type   = Ext2
Transmit power  = 10dBm
Hardware version = 1.2
...
```

Depending devices found, incoming **messages** are logged, i.e. for a TFA TS34C (Temperature & Humidity sensor)

```
30.12.2017 13:46:01
Packettype      = TEMP_HUM
subtype         = TH7 - Cresta, TFA TS34C
                  channel 1
Sequence nbr    = 102
ID              = 280E decimal:10254
Temperature     = 17,3 °C
Humidity        = 54
Status          = Comfortable
Signal level    = 5 -80dBm
Battery         = OK
```

### Notes

- The RFXmngr is not only used to log connected devices but also to manually configure connected devices, like the Somfy RTS Devices. It is not intended to do so via openHAB.
- The Device **ID decimal value** is important as required in openHAB, i.e. the ID 10254 for the packet type TEMP\_HUM as used for the Thing TEMPERATURE\_HUMIDITY-10254 (see more below).

# Raspberry Pi

## USB Symlink

On the Raspberry Pi, the RFXtrx433E is connected to an USB port.

To ensure the right USB Port is used, a symbolic link (**symlink**) is assigned for an USB Port. Steps to create a symbolic link for the RFXCOM RFXtrx433E device connected to one of the USB ports:

1. Do not plug in the RFXCOM device.
2. List the devices: `ls /dev/tty*`
3. Plug in the RFXCOM device to an USB port.
4. List the devices again to check out the new USB device: `ls /dev/tty*`
5. A new device should be listed, i.e: **/dev/ttyUSB0**

To define the symlink, certain USB device information are required; `idVendor`, `idProduct`.

### Get detailed USB devices information

```
sudo lsusb -v
```

### Seek for the RFXCOM entry listed under `iManufacturer`

```
Bus 001 Device 008: ID 0403:6001 Future Technology Devices International, Ltd
FT232 USB-Serial (UART) IC
Device Descriptor:
...
  idVendor           0x0403 Future Technology Devices International, Ltd
  idProduct          0x6001 FT232 USB-Serial (UART) IC
  bcdDevice           6.00
  iManufacturer      1 RFXCOM
  iProduct            2 RFXtrx433
  iSerial             3 A1YQCEQY
...
```

The data required to define the symlink are the `idVendor`, `idProduct`.

### Create the symlink

Raspberry Pi folder `/etc/udev/rules.d/` file `99-usb-serial.rules`:

```
sudo nano /etc/udev/rules.d/99-usb-serial.rules
Add the line:
SUBSYSTEM=="tty", ATTRS{idVendor}=="0403", ATTRS{idProduct}=="6001",
SYMLINK+="ttyUSBBrfxtrx1", GROUP="dialout", MODE="0666"
```

### Add the openHAB user `openhab` to the `dialout` group

```
sudo usermod -a -G dialout openhab
```

**Add the symlink to JAVA\_OPTS**

To the JAVA\_OPTS environment variable in folder /etc/default , file openhab2

```
cd /etc/default
sudo nano openhab2
Add line:
#####
## JAVA OPTIONS
## Additional options for the JAVA_OPTS environment variable.
EXTRA_JAVA_OPTS="-Dgnu.io.rxtx.SerialPorts=/dev/ttyUSBBrfxtrx1"
```

Reboot the Raspberry Pi.

**Check the USB Port**

After boot, check the USB port

```
ls -l /dev/ttyUSBBrfxtrx1
```

The USB symlink is listed as

```
lrwxrwxrwx 1 root root 7 Dec 30 14:31 /dev/ttyUSBBrfxtrx1 -> ttyUSB0
```

Preparation of the RFXtrx433E is done, next install & configure in openHAB.

## WLAN Static IP

For openHABian running release Raspbian GNU/Linux 8 (jessie), set a fixed IP address.

### Edit dhcpcd.conf

```
sudo nano /etc/dhcpcd.conf
```

### Add the lines (change the IP addresses accordingly)

```
interface wlan0
static ip_address=192.168.N.NN/24
static routers=192.168.0.1
static domain_name_servers=192.168.0.1
```

### Reboot the Raspberry Pi and check the WLAN address

```
ifconfig
Output:
wlan0      Link encap:Ethernet  HWaddr b8:27:eb:fa:44:fc
            inet addr:192.168.N.NN  Bcast:192.168.0.255  Mask:255.255.255.0
...
```

## WLAN Power Save Mode

If the openHAB Basic UI is not refreshing, the cause could be that the Raspberry Pi WLAN Power Save Mode is ON. After certain time, the network connection drops.

### Check the Raspberry Pi WLAN power save mode

```
iw wlan0 get power_save
```

Example Output: Power save: on

#### *Notes*

iwconfig provides this information also, i.e. shows Power Management: off

To turn the Raspberry Pi WLAN Power Save Mode OFF, run

```
sudo iw wlan0 set power_save off
```

#### *Notes*

Run as sudo else error message: command failed: Operation not permitted (-1)

Check again the Power Save Mode (iwconfig or iw wlan0 get power\_save).

The Power Save Mode is set back to default ON when the Raspberry Pi reboots.

To turn the Raspberry Pi WLAN Power Save Mode OFF during reboot, add to the crontab

```
# Disable wlan power save
@reboot sudo iw wlan0 set power_save off
```

## Ethernet Link volkszaehler Raspberry Pi

The volkszaehler Raspberry Pi and the Home Automation Raspberry Pi are direct connected via an Ethernet cable. A standard cable is used, no twisted wires required.

The communication uses ETH0 with a static network addresses on both sides.

### IMPORTANT:

The ETH0 network address must be different then the WLAN network address.

### Configuration

<b>volkszaehler Raspberry Pi</b> [Raspbian GNU/Linux 8 (jessie)] Linux 39 4.1.19-v7+ #858	<b>Home Automation Raspberry Pi</b> [Raspbian GNU/Linux 8 (jessie)] Linux openHABianPi 4.9.35-v7+ #1014
eth0 inet addr:169.254.87.85 Bcast:169.254.255.255 Mask:255.255.0.0	eth0 inet addr:169.254.87.84 Bcast:169.254.255.255 Mask:255.255.0.0
Set the static Ethernet network Address sudo nano /etc/dhcpd.conf	sudo nano /etc/dhcpd.conf
Add the lines	
interface eth0 static ip_address=169.254.87.85 static routers=169.254.0.1	interface eth0 static ip_address=169.254.87.84 static routers=169.254.0.1

### Notes

\$ uname -a to determine the Linux version

\$ cat /etc/os-release to determine the Raspberry Pi release

Steps to login from Raspberry Pi openHAB to the Raspberry Pi Domoticz:

- Login to Raspberry Pi openHAB as user openhabian.
- Check if the Raspberry Pi Domoticz is reachable: \$ping 169.254.87.85
- Check if the sshd service is running on the Raspberry Pi openHAB: \$ps ax | grep sshd
- Connect to the Raspberry Pi Domoticz via ssh: \$ssh [pi@169.254.87.85](mailto:pi@169.254.87.85)  
pi@169.254.87.85's password: \*\*\*\*\*
- The pi@139 prompt is shown: pi@139 ~ \$

## Firmware Update

It is recommended to update the firmware – **BUT USE AT OWN RISK.**

### Why?

Found on the Raspberry Pi forum an issue with the Firmware related to WiFi network driver, which caused a steady increase of Memory used.

Action taken:

Installed & run tool [rpi-update](#) to upgrade the Raspberry Pi firmware from 4.9.35-v7+ to 4.9.76-v7+.





# openHAB Bindings

Additional information gathered, for some of the bindings used for this solution.

## *Notes*

To check which openHAB Add-ons have been installed, check the add-ons configuration file.

## **Configuration File**

```
/var/lib/openhab2/config/org/openHAB/addons.config
```

## **Configuration Content**

```
action="mail,mqtt"  
binding="gpio,mqtt1,tinkerforge1,exec,rfxcom,http1,weather1,astro,hue"  
felix.fileinstall.filename="file:/var/lib/openhab2/etc/org.openhab.addons.cfg"  
package="standard"  
persistence="rrd4j"  
remote="true"  
service.pid="org.openhab.addons"  
transformation="regex,jsonpath,map,scale,javascript"  
ui="basic,paper,habpanel"
```

# RFXCOM Binding

## Notes

Some notes regarding the RFXCOM Binding to get an understanding of the concept.

A RFXCOM Device requires the openHAB [RFXCOM Binding](#) (must read).

The openHAB Paper UI is used to configure the Binding, Things and Channels.

The openHAB Textual Configuration is used to configure the Items.

The RFXCOM Binding has a number of Things from which two are used for this Home Automation solution:

- Thing **RFXCOM USB Transceiver Bridge** - used for manual configuration (rfxcom:bridge).
- Thing **RFXCOM Rfy Actuator** - used for configuring the Somfy RTS Devices (rfxcom:rfy).

### *Things, Channels, Items*

When the RFXCOM USB Transceiver gets online, new Things are created and stored in the openHAB Inbox (depending devices found).

These new Things can be added from the Paper UI Inbox to the openHAB Things list.

*(Things)* Based on this solution, the new Things added are:

3x RFXCOM Temperature-Humidity Sensor and 1x RFXCOM Wind Sensor (see previous hardware list).

*(Channels)* Each of these Things has one or more Channels:

Example Channels for the Thing TEMPERATURE\_HUMIDITY-10254:

Temperature, Humidity, Signal Strength, Battery Level, Low Battery.

*(Items Paper UI)* From these Channels, Linked Items can be created.

Example Item Temperature: Temperature

(TEMPERATURE\_HUMIDITY10254\_Temperature)

The Items are used in the Paper UI Control Panel.

### *Notes*

This option is not used, because the items are configured using the Textual Configuration Files.

*(Items Textual Configuration)* From these Channels, Items are created via Textual Configurations. More details in the section Textual Item Configuration. An example of a textual configuration in the file /etc/openhab2/items/homeautomation.items:

```
Rollershutter aHA_BedRoom_Shutter "Shutter" <rollershutter> (gHA_BedRoom) {  
channel="rfxcom:rfy:6c53bd1e:shutter" }
```

In addition, manual configuration is done for devices which are not automatically found by the RFXCOM USB Transceiver Bridge.

These are devices which are not transmitting regular signals.

## Binding Installation

In the openHAB2 Paper UI: Goto Add-ons > Bindings > RFXCOM  
Install the binding (used binding-rfxcom - 2.2.0).

## Thing USB Transceiver Add & Configure

In the openHAB Paper UI:

Go to Configuration > Things > + > RFXCOM Binding > Manually Add Thing > RFXCOM USB Transceiver.

Configure parameters for the Thing:

- Name: RFXCOM USB Transceiver
- Location: Leave empty
- Serial Port: /dev/ttyUSBBrfxtrx1
- Skip Transceiver configuration

Add the RFXCOM USB Transceiver Thing to become online.

### Check the log (openhabianpi:9001)

Some example log entries

```
2017-12-30 14:56:24.264 [hingStatusInfoChangedEvent] - 'rfxcom:bridge:e64474fc'
changed from NULL to INITIALIZING
2017-12-30 14:56:24.271 [hingStatusInfoChangedEvent] - 'rfxcom:bridge:e64474fc'
changed from INITIALIZING to OFFLINE
2017-12-30 14:56:24.839 [INFO ] [g.rfxcom.handler.RFXComBridgeHandler] - RFXCOM
transceiver/receiver type: _433_92MHZ_TRANSCEIVER, hw version: 3.1, fw version: 22
2017-12-30 14:56:24.886 [hingStatusInfoChangedEvent] - 'rfxcom:bridge:e64474fc'
changed from OFFLINE to ONLINE
```

As the device is online, connected devices are recognized and messages are send to the openHAB2 Paper UI Inbox stating new Things found.

Look at the openHAB log openhabianpi:9001 and search for entries containing rfxcom, i.e.

```
2017-12-31 11:09:57.383 [INFO ] [g.rfxcom.handler.RFXComBridgeHandler] - RFXCOM
transceiver/receiver type: _433_92MHZ_TRANSCEIVER, hw version: 3.1, fw version: 22
2017-12-31 11:09:57.436 [hingStatusInfoChangedEvent] - 'rfxcom:bridge:e64474fc'
changed from OFFLINE to ONLINE
... and many more
```

## Somfy Blinds RTS Motor Manual Configuration

To control Somfy Blinds with RTS Motors via openHAB, a Device ID is required by openHAB.

To set the Device ID for a Somfy RTS device, follow the steps as outlined in the RFXCOM User Manual (get from [homepage](#)).

### IMPORTANT:

The program button is on the back side of the remote control (little red knob) - hold down till the roller shutter moves up & down for a second.

### Device ID and Unit Code defined

Device	RFXCOM Device ID	Unit Code	openHAB Device ID
Living Room Shutter	1 09 09 Decimal: 67849	2	67849.2
Bed Room Blinds	9 09 09 Decimal: 592137	1	592137.1
<i>Notes</i>	Device ID must be unique per device	Unit Code must be unique per device	openHAB requires the decimal value of the RFXCOM Device ID

Example sending a **Program Command** from the RFXmngnr to the Somfy RTS Device

```
RFY command
=====
Packettype    = RFY
subtype       = RFY
Sequence nbr  = 8
id1-3         = 090909 decimal:592137
Unit          = 1
Command       = program
rfu1          = 00
rfu2          = 00
rfu3          = 00
Signal level  = 0  -120dBm
-----
30.12.2017 13:46:10
-----
30.12.2017 13:46:10
Packettype    = Receiver/Transmitter Message
subtype       = Transmitter Response
Sequence nbr  = 8
response      = ACK, data correct transmitted
```

# Thing Rfy Actuator Configuration

A roller shutter is not recognized as the Somfy RTS motors do not regular transmit messages.

In the openHAB Paper UI: Go to Configuration > Things > + > RFXCOM Binding > Manually Add Thing > RFXCOM Rfy Actuator.

## Configure Parameters for the Thing

- Name: RFXCOM Blinds Bedroom
- Location: Home
- Bridge Selection: RFXCOM USB Transceiver - rfxcom:bridge:e64474fc
- Configuration Parameters:
- Device ID: 592137.1
- Sub Type: RFY

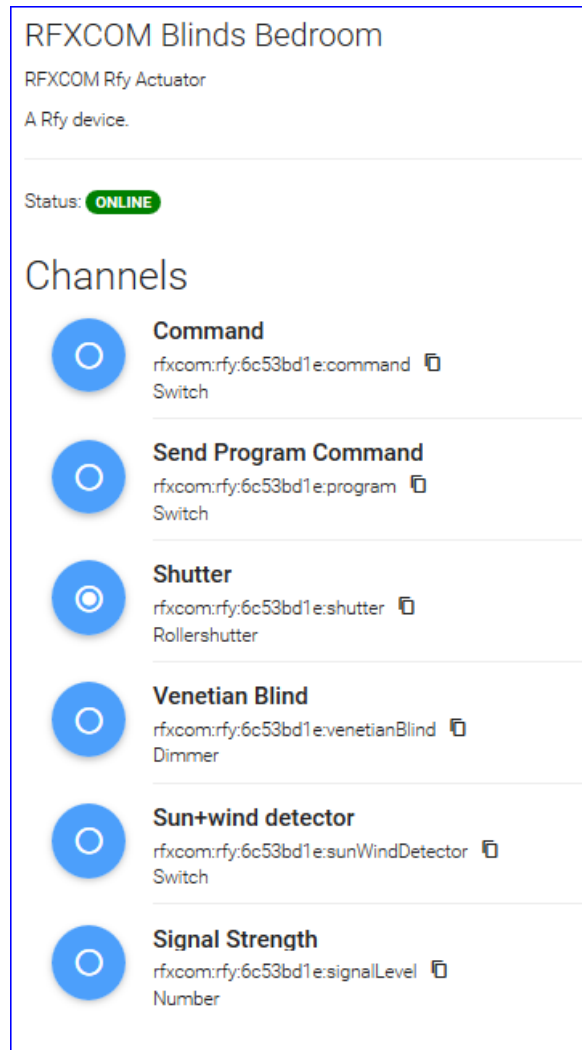
Add the Thing to become active.

Name	
RFXCOM Blinds Bedroom	
Location	
Home	
Bridge Selection	
RFXCOM USB Transceiver - rfxcom:bridge:e64474fc	
Configuration Parameters	
Configure parameters for the thing.	
Device Id	Sub Type
592137.1	RFY
Unit Id + unit code, separated by dot. Example 100.1	
Specifies device sub type.	

### Create a new Item for the Thing

Configuration > Things > RFXCOM Blinds Bedroom > Add Shutter Channel > Leave name as Shutter > Type set to Rollershutter > Add Link

Result with Channels



### Roller Shutter Item Example

Rollershutter item in the textual configuration file

(/etc/openhab2/items/homeautomation.items)

```
Rollershutter    aHA_BedRoom_Shutter    "Shutter"
<rollershutter> (gHA_BedRoom) { channel="rfxcom:rfy:6c53bd1e:shutter" }
```

### Roller Shutter Sitemap Example

The previous defined rollershutter item used as switch in the sitemap

(/etc/openhab2/sitemaps/homeautomation.sitemap)

```
Frame label="Bed Room" {
  ...
  Switch item=aHA_BedRoom_Shutter
  ...
}
```

## Thing Temperature\_Humidity (TFA Dostmann TS34C)

As mentioned, 3 TEMPERATURE\_HUMIDITY Things from type **TFA Dostmann TS34C**, have been added;

- TEMPERATURE\_HUMIDITY-10254
- TEMPERATURE\_HUMIDITY-10766
- TEMPERATURE\_HUMIDITY-17678.

### Configuration

Let's look at the configuration details of the Thing TEMPERATURE\_HUMIDITY-10254 (which applies also for the other TEMPERATURE\_HUMIDITY Things):

- Name: TEMPERATURE\_HUMIDITY-10254
- Location: Home
- Bridge Selection: RFXCOM USB Transceiver - rfxcom:bridge:e64474fc
- Configure parameters for the thing.
- Device Id: 10254
- Sub Type: TFA TS34C, Cresta

Example openHAB log device entry:

```
2017-12-31 11:48:58.210 [vent.ItemStateChangedEvent] -
TEMPERATURE_HUMIDITY10254_Humidity changed from 54 to 55
```

### Channels List

Each thing has a number of channels.

When selecting this Thing, the information on the Paper UI page listed:

TEMPERATURE\_HUMIDITY-10254, RFXCOM Temperature-Humidity Sensor, A  
Temperature-Humidity device, Status: ONLINE

Property	Channel	Type
Temperature	rfxcom:temperaturehumidity:e64474fc:10254:temperature	Number
Humidity	rfxcom:temperaturehumidity:e64474fc:10254:humidity	Number
Signal Strength	rfxcom:temperaturehumidity:e64474fc:10254:signalLevel	Number
Battery Level	rfxcom:temperaturehumidity:e64474fc:10254:batteryLevel	Number
Low Battery	rfxcom:temperaturehumidity:e64474fc:10254:lowBattery	Switch

## Channels Item Configuration

This is an example of applying the previous listed channels to items in a textual configuration file (/etc/openhab2/items/homeautomation.items):

```
Group gLivingRoom
Number nRFXLivingRoom_Temperature "Temperature [%.2f °C]" <temperature>
(gLivingRoom) { channel="rfxcom:temperaturehumidity:e64474fc:10254:temperature" }

Number nRFXLivingRoom_Humidity "Humidity [%.2f %]" <humidity> (gLivingRoom) {
channel="rfxcom:temperaturehumidity:e64474fc:10254:humidity" }
Group gLivingRoomTempHum

Number nRFXLivingRoom_SignalLevel "Signal Level [%.0f]" <energy>
(gLivingRoomTempHum) {
channel="rfxcom:temperaturehumidity:e64474fc:10254:signalLevel"}

Number nRFXLivingRoom_BatteryLevel "Battery Level [%.0f %]" <batterylevel>
(gLivingRoomTempHum) {
channel="rfxcom:temperaturehumidity:e64474fc:10254:batteryLevel"}

Switch aRFXLivingRoom_LowBattery "Low Battery" <lowbattery> (gLivingRoomTempHum) {
channel="rfxcom:temperaturehumidity:e64474fc:10254:lowBattery"}
```



## Thing Wind (TFA Dostmann 30.3168)

A **TFA Dostmann 30.3168 Wind Meter** (speed, direction, temperature) is connected to the RFXCOM RFXtrx433E.

### Configuration

The configuration details of the Thing WIND40214 in the Paper UI:

- Name: WIND-40214
- Location: Home
- Bridge Selection: RFXCOM USB Transceiver - rfxcom:bridge:e64474fc
- Configure parameters for the thing.
- Device Id: 40214
- Sub Type: TFA

Example openHAB log device entry

```
2018-01-05 13:46:16.859 [vent.ItemStateChangedEvent] - WIND40214_WindDirection
changed from 45.0 to 112.0
```

### Channels List

The thing has a number of channels. When selecting this Thing, the information on the Paper UI page listed:

WIND-40214, RFXCOM Wind Sensor, A Wind device, Status: ONLINE

Property	Channel	Type
Average Wind Speed	rfxcom:wind:e64474fc:40214:avgWindSpeed	Number
Wind Gust	rfxcom:wind:e64474fc:40214:windSpeed	Number
Wind Direction	rfxcom:wind:e64474fc:40214:windDirection	Number
Temperature	rfxcom:wind:e64474fc:40214:temperature	Number
Chill Temperature	rfxcom:wind:e64474fc:40214:chillTemperature	Number
Signal Strength	rfxcom:wind:e64474fc:40214:signalLevel	Number
Battery Level	rfxcom:wind:e64474fc:40214:batteryLevel	Number
Low Battery	rfxcom:wind:e64474fc:40214:lowBattery	Switch

### Channels Item Configuration

This is an example of applying the previous listed channels to items in a textual configuration file (/etc/openhab2/items/homeautomation.items):

```
Number nRFXWind_Gust "Wind Gust [%.2f m/s]" <wind> { channel="
rfxcom:wind:e64474fc:40214:windSpeed" }
```

For usage, see function Gardenshed.

## Thing FrontDoor-Security (PB62R)

The Thing FrontDoor-Security uses a contact device **PB-62R Detector** (433Mhz Alarm System compatible with PD-906, PG-100, PG-500, other 433 MHZ systems).

### Configuration

The Paper UI configuration details:

- Name: FrontDoor-Security
- Location: Home
- Bridge Selection: RFXCOM USB Transceiver - rfxcom:bridge:e64474fc
- Configure parameters for the thing.
- Device Id:6031104
- Sub Type: X10 security door/window sensor

Example openHAB log device entry

```
2018-01-06 14:35:17.954 [vent.ItemStateChangedEvent] - Security_FrontDoor_Status
changed from NORMAL to ALARM_TAMPER
```

### Channels List

The thing has a number of channels.

When selecting this Thing, the information on the Paper UI page listed:

SECURITY1-6031104, RFXCOM Security1, rfxcom:security1:e64474fc:6031104

Property	Channel	Type
Status	rfxcom: security1:e64474fc:6031104:status	String
Contact	rfxcom:security1:e64474fc:6031104:contact	Contact
Motion	rfxcom:security1:e64474fc:6031104:motion	Switch
Signal Strength	rfxcom:security1:e64474fc:6031104:signalLevel	Number
Battery Level	rfxcom:security1:e64474fc:6031104:batteryLevel	Number
Low Battery	rfxcom:security1:e64474fc:6031104:lowBattery	Switch

### Channels Item Configuration

This is an example of applying the previous listed channels to items in a textual configuration file (/etc/openhab2/items/ homeautomation.items):

```
Group gSecurityFrontDoor "Front Door Security" <shield>
String sSecurity_FrontDoor_Status "Status" <security>
(gSecurityFrontDoor) { channel="rfxcom:security1:e64474fc:6031104:status" }
DateTime dtSecurity_FrontDoor_StatusTime "Changed [%1$td.%1$tm.%1$tY
%1$tH:%1$tM]" <time> (gSecurityFrontDoor)
Switch aSecurity_FrontDoor_Reset "Reset Status" (gSecurityFrontDoor)
Contact aSecurity_FrontDoor_Contact "Contact" {
channel="rfxcom:security1:e64474fc:6031104:contact" }
Number nSecurity_FrontDoor_BatteryLevel "Battery Level [%d]" {
channel="rfxcom:security1:e64474fc:6031104:batteryLevel" }
Number nSecurity_FrontDoor_SignalLevel "Signal Level [%d]" {
channel="rfxcom:security1:e64474fc:6031104:signalLevel" }
```

## Rules Configuration

The device sends a status changed ALARM\_TAMPER captured in the Item Security\_FrontDoor\_Status, when the contacted is opened.

The device does not reset itself, means a trigger is needed to reset the device state to NORMAL.

This is done manually via Item Switch Security\_FrontDoor\_Reset.

If the status state changes, the change time is updated in the Item Security\_FrontDoor\_StatusTime.

The rules are handling the state changes.

```
rule "Security FrontDoor Contact"
when
    sItem Security_FrontDoor_Status changed
then
    dtSecurity_FrontDoor_StatusTime.postUpdate( new DateTimeType() )
end

rule "Security FrontDoor Reset"
when
    Item aSecurity_FrontDoor_Reset changed
then
    if (aSecurity_FrontDoor_Reset.state == ON) {
        aSecurity_FrontDoor_Reset.state = OFF
        sSecurity_FrontDoor_Status.postUpdate("NORMAL")
    }
end
```

Enhance the rules to notify, for example via email, if the door is opened.

Example Rule entry using the Mail Action:

- configured via the Paper UI > Configuration > Actions > Mail Action.
- Ensure to update the configuration file /etc/openhab2/services/mail.cfg.

```
rule "Security FrontDoor Contact"
when
    Item sSecurity_FrontDoor_Status changed
then
    dtSecurity_FrontDoor_StatusTime.postUpdate( new DateTimeType() )

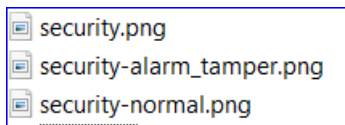
    if (sSecurity_FrontDoor_Status.state.toString == "ALARM_TAMPER") {
        val String logTime = String::format( "%1$d-%1$tm-%1$tY %1$tH:%1$tM:%1$tS",
new java.util.Date )
        sendMail("TOADDRESS", "Security FrontDoor", "Door opened at "+ logTime)
    }
end
```

## Icon Configuration

Custom Icons, in folder `/etc/openhab2/icons/classic`, are used to display the state of the device:

- `security.png` – the default icon if the state is unknown, i.e. NULL
- `security-normal.png` – the state is normal (display as NORMAL)
- `security-alarm_tamper.png` – the state is in alarm (displayed as ALARM\_TAMPER)

This feature is supported since openHAB2. See Basic UI screen shots below how the icons change.



## Device Status

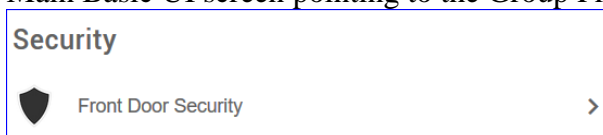
The device has a number of states. For this solution only 2 are used; NORMAL, ALARM\_TAMPER.

For next version consider to use BATLOW and trigger actions.

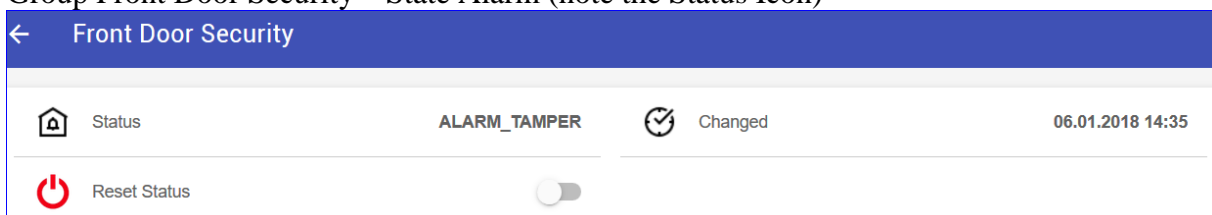
```
NORMAL(0),NORMAL_DELAYED(1),
ALARM(2),ALARM_DELAYED(3),
MOTION(4),NO_MOTION(5),
PANIC(6),END_PANIC(7),IR(8),
ARM_AWAY(9),ARM_AWAY_DELAYED(10),ARM_HOME(11),ARM_HOME_DELAYED(12),DISARM(13),
LIGHT_1_OFF(16),LIGHT_1_ON(17),
LIGHT_2_OFF(18),LIGHT_2_ON(19),
DARK_DETECTED(20),LIGHT_DETECTED(21),
BATLOW(22),
PAIR_KD101(23),
NORMAL_TAMPER(128),NORMAL_DELAYED_TAMPER(129),
ALARM_TAMPER(130),ALARM_DELAYED_TAMPER(131),
MOTION_TAMPER(132),NO_MOTION_TAMPER(133),
UNKNOWN(255)
```

## Basic UI Screen Shots

Main Basic UI screen pointing to the Group Front Door Security

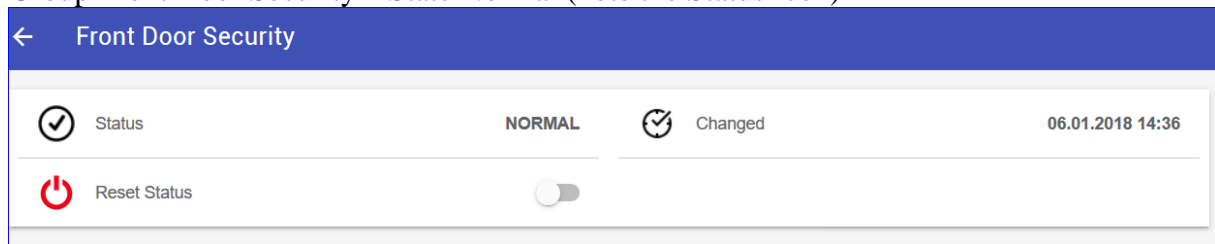


Group Front Door Security – State Alarm (note the Status Icon)



Click on the switch Reset Status, will set the Status to Normal and changes the Icon.

Group Front Door Security – State Normal (note the Status Icon)



# Weather Binding (OpenWeatherMap Service)

To obtain weather information, used the service from OpenWeatherMap (OWM).  
The openHAB Weather Binding is used to access the OWM service.

## Binding Installation

In the openHAB2 Paper UI: Goto Add-ons > Bindings > Weather Binding  
Install the binding (used binding-weather1 - 1.11.0).

## Binding Configuration

### Configuration File

```
/etc/openhab2/services/weather.cfg
```

### Configuration Content

```
apikey.OpenWeatherMap= Get from OpenWeatherMap after registration for the service
location.home.name=Home
location.home.latitude=53.636684
location.home.longitude=9.798321
location.home.provider=OpenWeatherMap
location.home.language=en
location.home.updateInterval=5
```

### Warning Incomplete location

After defining the OWM configuration, openHAB logs a WARN

```
2018-01-04 14:58:18.810 [WARN ] [eather.internal.common.WeatherConfig] -
Incomplete location config for locationId '<locationId>'. Check openhab.cfg.
```

### Workaround

- Stop openHAB
- In file /var/lib/openhab2/config/org/openHAB/weather.config, delete the incorrect entries, i.e. pointing to <locationId>.
- Start openHAB

Example cleaned up configuration file /var/lib/openhab2/config/org/openHAB/weather.config

```
apikey.OpenWeatherMap="*****"
location.home.language="en"
location.home.latitude="53.636684"
location.home.longitude="9.798321"
location.home.name="Home"
location.home.provider="OpenWeatherMap"
location.home.updateInterval="15"
service.pid="org.openhab.weather"
```

### Service Start Log Entry

Example log entry for successful start of the Weather service using the OpenWeatherMap

```

2018-01-04 15:17:21.938 [INFO ] [eather.internal.common.WeatherConfig] -
ProviderConfig[providerName=OPENWEATHERMAP,apiKey=***]
2018-01-04 15:17:21.941 [INFO ] [eather.internal.common.WeatherConfig] -
LocationConfig[providerName=OPENWEATHERMAP,language=en,updateInterval=5,latitude=5
3.636684,longitude=9.798321,woeid=<null>,locationId=home,name=Home]

2018-01-04 15:17:25.009 [INFO ] [ternal.scheduler.WeatherJobScheduler] - Starting
and scheduling weatherJob-home with interval of 5 minutes

```

## Item Configuration

The items are configured using the locationId (as defined in /etc/openhab2/services/weather.cfg), a type and property (from the type)

## Example Current Weather






```

Number    nWeather_Temperature    "Temperature [%.2f °C]"
{weather="locationId=home, type=temperature, property=current"}
Number    nWeather_Humidity       "Humidity [%d %%]"
{weather="locationId=home, type=atmosphere, property=humidity"}
Number    nWeather_Pressure       "Pressure [%.2f hPa]"
{weather="locationId=home, type=atmosphere, property=pressure"}

```

### Notes

Only some selective properties are shown.

Weather		
	Temperature	3.0 °C
	Humidity	93 %
	Pressure	987 hPa
	Wind Direction	WSW
	Windspeed Beaufort	3

## Example Weather Forecast

Show the weather forecast for next days.

```

Group    gHA_Weather_Lanzarote_Forecast "Lanzarote Forecast" <lanzarote>
DateTime dtHA_Weather_Lanzarote_Forecast1_ObservationTime    "[%1$tA
%1$td.%1$tm.%1$tY]" <time> (gHA_Weather_Lanzarote_Forecast)
{weather="locationId=lanzarote, forecast=1, type=condition,
property=observationTime"}
Number    nHA_Weather_Lanzarote_Forecast1_Clouds    "Clouds [%.0f %%]"
<sun_clouds>    (gHA_Weather_Lanzarote_Forecast)
{weather="locationId=lanzarote, forecast=1, type=clouds, property=percent"}
Number    nHA_Weather_Lanzarote_Forecast1_Temp_Min    "Temperature min [%.2f °C]"
<temperature>    (gHA_Weather_Lanzarote_Forecast) {weather="locationId=lanzarote,
forecast=1, type=temperature, property=min"}

```

```







Number nHA_Weather_Lanzarote_Forecast1_Temp_Max      "Temperature max [%.2f °C]"
<temperature> (gHA_Weather_Lanzarote_Forecast) {weather="locationId=lanzarote,
forecast=1, type=temperature, property=max"}
Number nHA_Weather_Lanzarote_Forecast1_WindSpeedBft "Windspeed Beaufort [%d]"
<wind> (gHA_Weather_Lanzarote_Forecast) {weather="locationId=lanzarote,
forecast=1, type=wind, property=speed, unit=beaufort"}

DateTime dtHA_Weather_Lanzarote_Forecast2_ObservationTime "[%1$tA
%1$td.%1$tm.%1$tY]" <time> (gHA_Weather_Lanzarote_Forecast)
{weather="locationId=lanzarote, forecast=2, type=condition,
property=observationTime"}
...

```

*Notes:*

Only some selective properties shown for the two days forecast.

← Lanzarote Forecast		
		Wednesday 17.01.2018
	Clouds	0 %
	Temperature min	17.80 °C
	Temperature max	19.02 °C
	Windspeed Beaufort	6
		Thursday 18.01.2018



# Weather Binding (Weather Underground Service)

The weather information is obtained every 10 minutes from the Weather Underground service. One Thing, Weather Home, is defined with selective channels.

## Notes

Whilst exploring the Weather Underground Service, decided not to use this service, but OpenWeatherMap service instead. Reason: OpenWeatherMap delivers more weather information. For learning purposes kept the information in this document.

## Binding Configuration

The configuration is done at each Thing and Channel.

## Thing Configuration

Name	Weather Home
Thing ID	63b65c0a
API Key	Get from Weather Underground after registration for the service (API key to access the Weather Underground service)
Location of Weather Information (latitude,longitude)	53.636684,9.798321 Hints: <ul style="list-style-type: none"> <li>• Ensure no spaces between Lat,Lon</li> <li>• This <a href="#">service</a> is handy to get the LAT,LON for a location.</li> </ul>
Language	English (Language to be used by the Weather Underground service)
Refresh interval	5 (Specifies the refresh interval in minutes)
Channels	Selected channels are used. See items file.

## Item Configuration

This is an example.





```
Group gWeather
DateTime dtWeather_ObservationTime "Observation time [%1$tH:%1$tM]" <time>
(gWeather) {channel="weatherunderground:weather:63b65c0a:current#observationTime"}
Number nWeather_Temperature "Temperature [%1f °C]" <temperature> (gWeather)
{channel="weatherunderground:weather:63b65c0a:current#temperature"}
Number nWeather_Humidity "Humidity [%d %]" <humidity> (gWeather)
{channel="weatherunderground:weather:63b65c0a:current#relativeHumidity"}
Number nWeather_Pressure "Pressure [%1f hPa]" (gWeather)
{channel="weatherunderground:weather:63b65c0a:current#pressure"}
String sWeather_WindDirection "Wind Direction [%s]" <compass> (gWeather)
{channel="weatherunderground:weather:63b65c0a:current#windDirection"}
Number nWeather_WindGust "Wind Gust [%1f km/h]" <wind> (gWeather)
{channel="weatherunderground:weather:63b65c0a:current#windGust"}
Number nWeather_WindGustBft "Wind Gust Bft [%d]" <wind>
Number nWeather_WindGustBft "Wind Gust Bft [SCALE(windspeed.scale):%s]" <wind>
(gWeather) {channel="weatherunderground:weather:63b65c0a:current#windGust"}
Number nWeather_UV "UV Index [%1f]" <uvindex>
{channel="weatherunderground:weather:63b65c0a:current#UVIndex"}
```

# Astro Binding

The Astro binding is used to obtain sun rise time, sun set time , daylight duration and the moon phase.

## Thing Astro moon data & Sun data

The Things defined via the Paper-UI are the **Astro moon data** and **Astro sun data** for the **location lat lon 53.636684,9.798321**.

	Sunrise	08:27
	Sunset	16:38
	Moon Phase	Waxing crescent
	Daylight	8 hrs 3 mins

## Item Configuration

The item configuration uses the astro:sun: cb065030 channel.

Examples

```

DateTime      dtHA_Astro_Sunrise_Time      "Sunrise [%1$tH:%1$tM]"
<sunrise>     (gHA_Weather_Home) {channel="astro:sun:cb065030:rise#start"}
DateTime      dtHA_Astro_Sunset_Time       "Sunset [%1$tH:%1$tM]"
<sunset>      (gHA_Weather_Home) {channel="astro:sun:cb065030:set#end"}
String        dtHA_Astro_MoonPhase_Name    "Moon Phase [%s]"          <moon>
(gHA_Weather_Home) {channel="astro:moon:e5b32a75:phase#name"}
Number        nHA_Astro_DayLight           "Daylight [JS(daylight.js):%s]"
<light>       (gHA_Weather_Home) {channel="astro:sun:cb065030:daylight#duration"}
Number        nHA_Astro_DayLight_Minutes   "Daylight [%d]"
{channel="astro:sun:cb065030:daylight#duration"}

```

### Daylight Calculation

The daylight in hours and minutes is calculated using JavaScript with the number of minutes as input and returned as a string.

```

(function(minutes) {
  var hours    = Math.floor(minutes / 60);
  minutes     -= Math.floor(hours * 60);
  if (hours <= 0) {
    return "0 hrs " + minutes + " mins";
  }
  return hours + " hrs " + minutes + " mins";
})(input)

```

# MQTT Binding

The MQTT binding enables openHAB to act as a MQTT client. The broker used is mosquitto.

## Binding Installation

In the openHAB2 Paper UI: Goto Add-ons > Bindings > MQTT Binding  
Install the binding (used binding-mqtt1 - 1.11.0).

## Install Mosquitto Broker

Install the Mosquitto broker:

Open terminal > openHABian-config > Add-ons Mosquitto

### *Issue Invalid Signature*

If during Apply Improvements, via the openHABian-config tool, following message appears

```
http://repo.mosquitto.org jessie InRelease: The following signatures were invalid:
KEYEXPIRED 1515017477 KEYEXPIRED 1515017477 KEYEXPIRED 1515017477
```

### **Fix**

```
wget http://repo.mosquitto.org/debian/mosquitto-repo.gpg.key
sudo apt-key add mosquitto-repo.gpg.key
```

## Install openHAB MQTT Action

Install the openHAB MQTT action:

Paper UI > Add-ons > MQTT action > Install

Check the openHAB log file for entry:

```
2018-01-10 16:00:55.719 [thome.event.ExtensionEvent] - Extension 'action-mqtt' has
been installed.
```

## Services Configuration

Add the MQTT broker to the MQTT configuration file: **/etc/openhab2/services/mqtt.cfg**

```
# Local Mosquitto Broker running on the Raspberry Pi
openhabprod.url=tcp://localhost:1883
```

Restart openHAB (sudo systemctl restart openhab2.service) and check the log for entry:

```
2018-01-10 16:37:53.378 [INFO ] [t.mqtt.internal.MqttBrokerConnection] - Starting
MQTT broker connection 'openhabprod'
```

# Hue Binding

The Hue binding is used to control Philips Hue devices via the Hue bridge, which is an IP gateway to connected ZigBee devices, like Lights, Motion sensor etc.

## Hue Bridge Installation

Install the Hue Bridge according Philips Installation Guide.

## Binding Installation

In the openHAB2 Paper UI: Goto Add-ons > Bindings > Hue Binding  
Install the binding (used binding-hue - 2.2.0).

## Thing Hue Bridge

In the Paper UI select Add new Thing.  
The Philip Hue Thing has been found and listed as:

Philips hue (192.168.1.13), Hue Bridge, hue:bridge:00178863da1d
---

Add and show the Thing.  
It shows status offline. Pressed the pairing button on the bridge (middle big button).  
Check the openHAB log entries:

<pre>[INFO ] [binding.hue.handler.HueBridgeHandler] - Creating new user on Hue bridge 192.168.0.7 - please press the pairing button on the bridge. [INFO ] [binding.hue.handler.HueBridgeHandler] - User 'iphrrzgr1LpDNGnZgq5rfSA6aeTA4mPOkBC1A-s1C' has been successfully added to Hue bridge. [me.event.ThingUpdatedEvent] - Thing 'hue:bridge:00178863da1d' has been updated. [vent.ConfigStatusInfoEvent] - ConfigStatusInfo [configStatusMessages=[]] [me.event.ThingUpdatedEvent] - Thing 'hue:bridge:00178863da1d' has been updated. [hingStatusInfoChangedEvent] - 'hue:bridge:00178863da1d' changed from OFFLINE (CONFIGURATION_ERROR): Not authenticated. Press pairing button on the hue bridge or set a valid user name in configuration. to ONLINE [home.event.InboxAddedEvent] - Discovery Result with UID 'hue:0100:00178863da1d:1' has been added. [INFO ] [g.discovery.internal.PersistentInbox] - Added new thing 'hue:0100:00178863da1d:1' to inbox.</pre>
---

The status went to online.  
The properties of the Hue Bridge: serialNumber 00178863da1d, vendor Philips  
As a result of changing state to online, a Hue while lamp is found.

## Thing Hue White Lamp

The Paper UI Inbox lists:

### Hue white lamp 1

```
Dimmable Light, hue:0100:00178863da1d:1
Added as a Thing with properties:
Hue white lamp 1, Dimmable Light, A dimmable light.
Status: ONLINE
modelId LWB010
uniqueId 00:17:88:01:03:48:4a:27-0b
Current firmware version: 1.23.0_r20156
Channels: Brightness, hue:0100:00178863da1d:1:brightness, Dimmer
```

## Thing Hue Adding More Lamps

Adding more lamps is done, by

1. Switch on the new lamp
2. Check the Paper UI Inbox
3. Select the inbox entry and add as a thing

### Examples

#### Hue ambiance candle 1

```
Color Temperature Light
A dimmable light with tunable color temperature.
Status: ONLINE, Current firmware version: 1.15.2_r19181
Channels:
Color Temperature, hue:0220:00178863da1d:2:color_temperature, Dimmer
Brightness, hue:0220:00178863da1d:2:brightness, Dimmer
```

#### Hue white lamp 1

```
Dimmable Light
A dimmable light.
Status: ONLINE, Current firmware version: 1.23.0_r20156
Channels:
Brightness, hue:0100:00178863da1d:3:brightness, Dimmer
```

#### Hue white lamp 3

```
Dimmable Light
A dimmable light.
Status: ONLINE, Current firmware version: 1.23.0_r20156
Channels:
Brightness, hue:0100:00178863da1d:4:brightness, Dimmer
```

## Thing Log Examples

These are openHAB logging examples for switching power OFF and ON for a Light Thing (the Hue White Lamp 1).

### Light's power switched OFF

```
2018-01-20 11:46:47.018 [hingStatusInfoChangedEvent] - 'hue:0100:00178863da1d:1'
changed from ONLINE to OFFLINE: Hue bridge reports light as not reachable.
```

### Light's power switched ON

```
2018-01-20 11:47:47.208 [hingStatusInfoChangedEvent] - 'hue:0100:00178863da1d:1'
changed from OFFLINE: Hue bridge reports light as not reachable. to ONLINE

2018-01-20 11:47:47.221 [vent.ItemStateChangedEvent] - aHA_LivingRoom_HueLight1
changed from 19 to 100
```

### Changing the Brightness 100 to 23 to 0

```
2018-01-20 11:48:06.344 [ome.event.ItemCommandEvent] - Item
'aHA_LivingRoom_HueLight1' received command 23
2018-01-20 11:48:06.365 [vent.ItemStateChangedEvent] - aHA_LivingRoom_HueLight1
changed from 100 to 23
2018-01-20 11:48:12.606 [ome.event.ItemCommandEvent] - Item
'aHA_LivingRoom_HueLight1' received command 0
2018-01-20 11:48:12.628 [vent.ItemStateChangedEvent] - aHA_LivingRoom_HueLight1
changed from 23 to 0
```

## Hue Light Control Examples

Examples to control a Hue Light, in this case a Philips Hue WHITE E27 LED:

### Slider Control

Control the light intensity via slider with value range 0 (OFF) – 100 (FULL).



### Items Configuration

```
Dimmer aHA_LivingRoom_HueLight1 "Light1" <light> (gHA_LivingRoom) {  
channel="hue:0100:00178863da1d:1:brightness" }
```

### Sitemap Configuration

```
Slider item=aHA_LivingRoom_HueLight1
```

### Switch Control

Control the light intensity via 4 switches OFF, LOW, NORMAL, HIGH.



### Items Configuration

```
Dimmer aHA_LivingRoom_HueLight1 "Light 1" <light>  
(gHA_LivingRoom) { channel="hue:0100:00178863da1d:1:brightness" }
```

### Sitemap Configuration

```
Switch item=aHA_LivingRoom_HueLight1 mappings=[0="Off", 25="Low", 75="Normal",  
100="High"]
```

## Rule Control

Just to show setting the brightness via a Rule

### Items Configuration

```
Dimmer aHA_LivingRoom_HueLight1 "Light 1"          <light>  
(gHA_LivingRoom)  { channel="hue:0100:00178863da1d:1:brightness" }
```

### Sitemap Configuration

```
Switch  item=aHA_LivingRoom_HueLight1
```

### Rule Configuration

```
rule "Light Brightness Set"  
when  
  Item aHA_LivingRoom_HueLight1 received command  
then  
  switch receivedCommand{  
    case 0:{  
      aHA_LivingRoom_HueLight1.sendCommand(0)  
    }  
    case 1:{  
      aHA_LivingRoom_HueLight1.sendCommand(25)  
    }  
    case 2:{  
      aHA_LivingRoom_HueLight1.sendCommand(75)  
    }  
    case 3:{  
      aHA_LivingRoom_HueLight1.sendCommand(100)  
    }  
  }  
end
```



## MQTT Control

The purpose is to test controlling the brightness of a Hue Light via an MQTT topic. This solution can for example be used to control the light via an ESP8266 controller or Arduino with Pushbuttons, LCD Keypad Shield or Potentiometer.

### Set Hue Light Brightness

#### Topic to set the brightness

```
homeautomation/livingroom/huelight1/set
```

As a first option, tried by using an item definition only, but this option is not working. The log states, that the state has changed, but nothing happens with the light

```
Dimmer aHA_LivingRoom_HueLight1m "Light 1" <light> (gHA_LivingRoom)
{ channel="hue:0100:00178863da1d:1:brightness",
mqtt="<[openhabprod:homeautomation/livingroom/ huelight1/set:state:default:.*]" }
```

#### Workaround

Create Number Item to hold the brightness level between 0 – 100.

```
Number nHA_LivingRoom_HueLight1 { mqtt="<[openhabprod:homeautomation/livingroom/
huelight1/set:state:default:.*]" }
```

The Number Item nHA\_LivingRoom\_HueLight1, will set the Actuator Item aHA\_LivingRoom\_HueLight1 with the Hue Bridge channel:

```
Dimmer aHA_LivingRoom_HueLight1      " Light 1"          <light>
(gHA_LivingRoom)  { channel="hue:0100:00178863da1d:1:brightness" }
```

#### Create a Rule to set the brightness

The rule listens to state change of the number item to then send a command to the item connected to the Hue Bridge channel.

```
rule "HueLight1 Brightness Set"
when
  Item nHA_LivingRoom_HueLight1 changed
then
  logInfo("HUELIGHT1", "*** Brightness set to " +
nHA_LivingRoom_HueLight1.state.toString)
  aHA_LivingRoom_HueLight1.sendCommand(nHA_LivingRoom_HueLight1.state.toString)
end
```

#### Test via terminal mosquitto command

Set the brightness to 50 followed by setting to 0.

```
mosquitto_pub -t homeautomation/livingroom/huelight1/set -m 50
mosquitto_pub -t homeautomation/livingroom/huelight1/set -m 0
```

#### openHAB Log Example

```
[vent.ItemStateChangedEvent] - nHA_LivingRoom_HueLight1 changed from NULL to 50
```

```
[INFO ] [pse.smarthome.model.script.HUELIGHT1] - *** Brightness set to 50
[vent.ItemStateChangedEvent] - aHA_LivingRoom_HueLight1 changed from 20 to 50
[vent.ItemStateChangedEvent] - aHA_LivingRoom_HueLight1_Steps changed from 20 to 50
[vent.ItemStateChangedEvent] - nHA_LivingRoom_HueLight1 changed from 50 to 0
[INFO ] [pse.smarthome.model.script.HUELIGHT1] - *** Brightness set to 0
[ome.event.ItemCommandEvent] - Item 'aHA_LivingRoom_HueLight1' received command 0
[vent.ItemStateChangedEvent] - aHA_LivingRoom_HueLight1 changed from 50 to 0
[vent.ItemStateChangedEvent] - aHA_LivingRoom_HueLight1_Steps changed from 50 to 0
```

## Publish Brightness value

This is an example of publishing a topic which holds the brightness, if the Hue Light changes.

The brightness is set by Item aHA\_LivingRoom\_HueLight1 which has map values 0, 25, 60, 100 (see sitemap)

### Topic

```
homeautomation/livingroom/huelight1
```

### Rule Configuration

```
rule "HueLight1 Brightness Changed"
when
  Item aHA_LivingRoom_HueLight1 changed
then
  // Publish the message to topic using the specified MQTT broker.
  publish("openhabprod", "homeautomation/livingroom/huelight1",
aHA_LivingRoom_HueLight1.state.toString)
end
```

### Test Subscribe to Brightness change

The brightness is set to LOW, which value is mapped to 25 in the sitemap.

```
mosquitto_sub -v -t homeautomation/livingroom/huelight1
```

### Test Result

```
homeautomation/livingroom/huelight1 25
```

## Node-RED Control

The purpose is to test controlling the brightness of a Hue Light via [Node-RED](#) using the MQTT topics for a Hue Light.

### Node-RED

- Enables flow-based programming (FBP) for the Internet of Things.
- Can be used as an alternative openHAB Rules Engine (TODO ADD CHAPTER).
- Has a variety of add-ons from which the [Dashboard UI](#) is used.

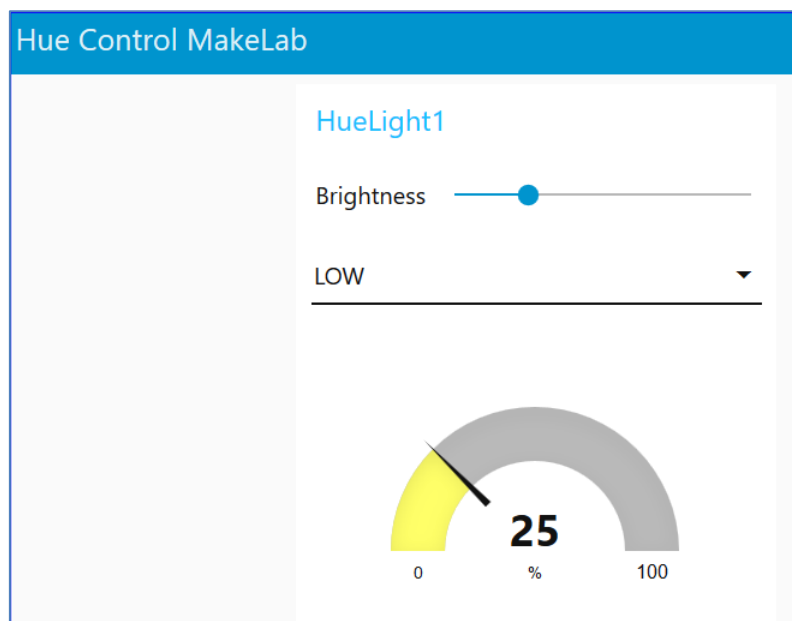
For this example, the dimmable Hue Light is located in the MakeLab.

The MQTT topics used are:

- homeautomation/makelab/huelight1 = contains the brightness set (Subscribe)
- homeautomation/makelab/huelight1/set = sets the brightness between 0 – 100 (Publish)

Controlled is the Hue Light via an external Raspberry Pi 3 with Node-RED installed.

### Node-RED Dashboard UI



The Dashboard has 3 UI Nodes: Slider, Drop Down and Gauge.

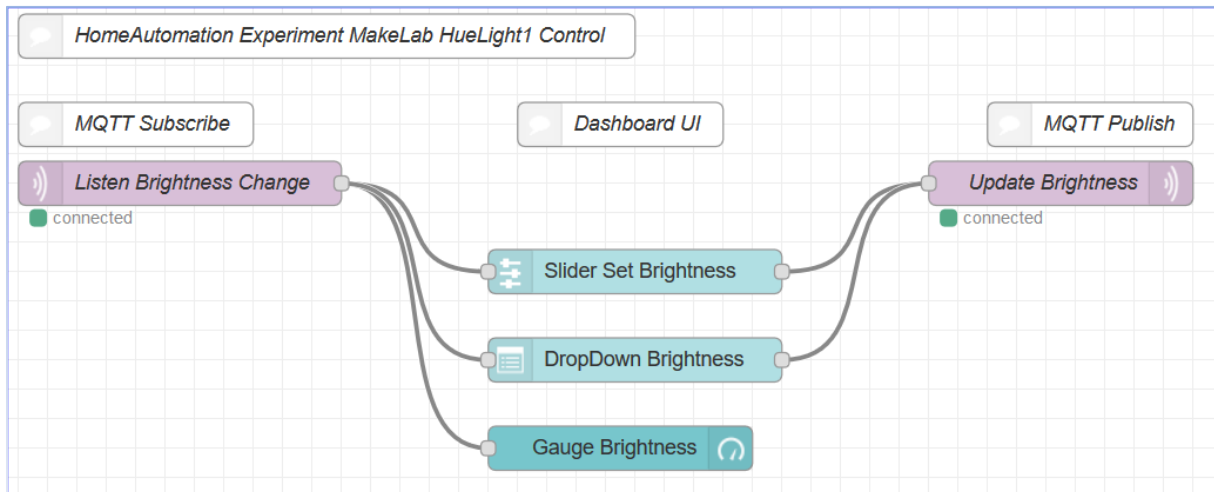
**Access the Dashboard** from a Browser by using URL.

<code>http://192.168.N.NN:1880/ui</code>
--

#### Notes

If there are more flows with several tabs defined; select the Tab Hue Control MakeLab to access this dashboard.

## Node-RED Flow



- **Node “Listen Brightness Change”** (mqtt-in) listens to brightness changes via topic homeautomation/makelab/huelight1 and sends the topic payload to the 3 Dashboard UI nodes.
- **Node “Slider Set Brightness”** (ui\_slider) sets the slider position to the message payload coming in, but also enables to set the brightness.  
The new message payload is send to the Node “Update Brightness” (mqtt-out) to send a new message with topic homeautomation/makelab/huelight1/set and payload between 0 -100.  
Disable option “If msg arrives on input, pass through to output”.
- **Node “DropDown Brightness”** (ui\_dropdown) enables to set 4 brightness values OFF (0), LOW (25), NORMAL (50), HIGH (100).  
The new message payload is send to the Node “Update Brightness” (mqtt-out) to send a new message with topic homeautomation/makelab/huelight1/set and payload between 0 -100.  
Disable option “If msg arrives on input, pass through to output”.
- **Node “Gauge Brightness”** (ui\_gauge) listens to changes from the Node mqtt-in and sets the gauge needle position accordingly.
- **Node “Update Brightness”** (mqtt-out) to publish the topic with the new payload: topic homeautomation/makelab/huelight1/set payload between 0 -100.
- *Notes*  
The MQTT Nodes are using the MQTT Broker running on the Home Automation Server. The Broker is running on IP:Port 192.168.N.NN;1883.

# Basic UI - Home Automation

## Objectives

To create an openHAB web interface to obtain information from and control connected devices. The layout has to be kept simple focussing on core items. It is not intended to use all possible configuration settings.

The user access is via

- Web Browser (URL `openhabianpi:8080/basicui/app?sitemap=homeautomation`) or
- openHAB Android application (native client for openHAB)

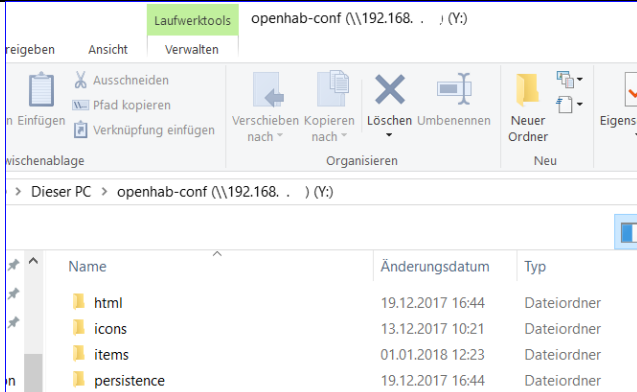
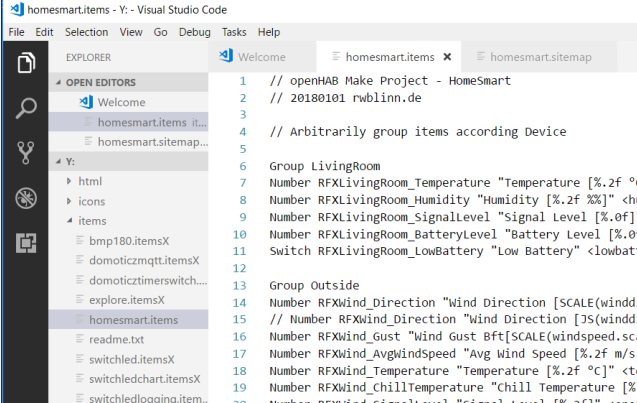
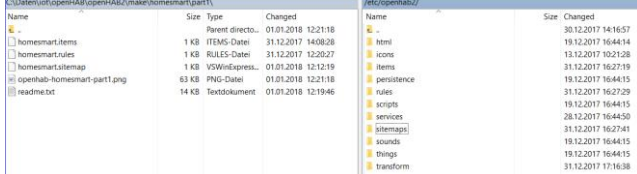
## Configuration Setup

### Development

The configuration for the Basic UI is based on textual files.

The configuration files are located on the Raspberry Pi in folder `/etc/openhab2/` with several subfolders.

The configuration files are developed on a Windows 10 device using VSCode.

<p>In Windows 10 a network drive (Y:) has been mapped to the Raspberry Pi openHAB configuration folder (<a href="http://192.168.N.NN/openhab-conf">\\192.168.N.NN\openhab-conf</a>).</p>	
<p>This folder is used in VSCode – the VSCode explorer lists the content. Select a file and edit.</p>	
<p>A backup of the configuration files is stored on the Windows 10 device using WinSCP.</p>	



## Configuration Files

The list of textual configuration files in sub folders of folde `/etc/openhab2/...`

File	Content
items/homeautomation.items	The items used to display data or as local variables used in f.e. chart period selection.
sitemap/homeautomation.sitemap	The sitemap accessed via web browser or Android app.
rules/homeautomation.rules	The rules applied to set item values.
persistence/rrd4j.persist	Define the rrd4j Chart strategy and items.
icons/classic	Custom icons used in the items and displayed via the sitemap.
services/weather.cfg	Define the parameter to obtain data from the OpenWeatherMap service via the Weather Binding.
services/mail.cfg	Define the send mail parameter used by the Mail Binding.
transform/windspeedkmhbft.scale	(test only)

# Naming Conventions

The naming convention for unique Groups and Items has 3 elements:

TypePrefix_Topic_Location_Function		
<b>Type</b>	Type of the Item (lowercase)	<ul style="list-style-type: none"> <li>• a = Actuator (Switch in Item, Contact, Rollershutter)</li> <li>• dt = DateTime, d = Date, t = Time</li> <li>• g = Group</li> <li>• s = String</li> <li>• n = Number (Switch in Rule use n)</li> <li>• v = Val</li> </ul>
<b>Prefix</b>	Unique Identifier for this configuration (UPPERCASE)	Example <b>HA_</b> for this Home Automation project
<b>Topic</b>	Dedicated topic or leave blank if Location is sufficient (CamelCase)	
<b>Location</b>	Location of the item or leave blank if Topic is sufficient (CamelCase)	
<b>Function</b>	Function of the item (CamelCase)	

## IMPORTANT

The name must be unique across all items files located in the openHAB Configuration folders (/etc/openhab2/items, /etc/openhab2/sitemaps ...).

## Examples

### Items

```

Group      gHA_Basement
Group      gHA_Security_FrontDoor
Number     nHA_Basement_Temperature
String     nHA_Security_FrontDoor_Status
Switch     aHA_Security_FrontDoor_Reset
DateTime   dtHA_Security_FrontDoor_StatusTime

```

### Rules

```

val  vRPiCPUTemperature = executeCommandLine("<script>", 5000)
var  Number             nRPiCPUTemperatureMax = 45

```



## Custom Icons

In the Item configuration, additional custom bitmap icons in PNG format with size 32x32 are used.

### Icons Folder

```
/etc/openhab2/icons/classic
```

To be able to display the icons, the openHAB Basic UI Configuration must be changed to support the Icons Bitmap Format.

Go to openHAB > Paper UI > Configuration > Services > Basic UI:  
Set the Icon Format to Bitmap.

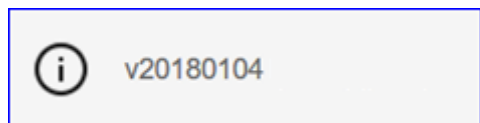
Basic UI item configuration example for icon info.png 32x32 with size 598 bytes

```
String sHomeAutomation_Version "v20180104 by Robert W.B. Linn" <info>
```

Basic UI sitemap configuration

```
Text item=sHomeAutomation_Version
```

### Basic UI Screenshot with the Custom Icon info



### Hint

The icons used are from [icons8](https://icons8.com).

## Layout Concept

The layout makes use of groups, means the items are grouped according a topic.

The topics defined as Groups (Configuration file `homeautomation.items`):

Group gHA\_Info  
Group gHA\_WeatherHome  
Group gHA\_WeatherLanzarote  
Group gHA\_LivingRoom  
Group gHA\_BedRoom  
Group gHA\_Basement  
Group gHA\_Security\_FrontDoor  
Group gHA\_Energy

Per group, items are defined. The items are not explained in details in this document, only where a rule is used using an item.

The sitemap (Configuration file `homeautomation.sitemap`) uses the groups defined.

## Sitemap Access

The url to access the sitemap from a Web browser:

`openhabianpi:8080/basicui/app?sitemap=homeautomation`

An option is to rename the sitemap to default, which will load Home Automation when calling:

`openhabianpi:8080/basicui`

## Parameter

Parameters are used for

- thresholds to indicate certain conditions
- notification recipients
- settings switch states

The values are hard coded in various textual configuration files.

### TODO

Seeking for a solution to centrally set parameter in an external file or database and load during start of the sitemap.

Parameter	Action	Defined
Maximum CPU Temperature	Shutdown the Raspberry Pi if above threshold	var Number nRPiCPUTemperatureMax = 60 (rule) valuecolor=[>50="yellow",>60="red"] (item)
Maximum RAM Used	Reboot the Raspberry Pi to free up memory	var Number nRPiRAMUsedPctMax = 90 (rule) valuecolor=[>80="yellow",>90="red"] (item)
Maximum Disc Space used	Shutdown the Raspberry Pi	var Number nRPiDiscSpaceUsedPctMax = 80 (rule) valuecolor=[>70="yellow",>80="red"] (item)
Notification Email Receipient		var String sEmailTo = "ADDRESS" (rule)
Home Temperature	Change sitemap value color	valuecolor=[>3="blue",<=3="red"] (item)
Home Wind Bft	Change sitemap value color	valuecolor=[>7="yellow",>9="red"] (item)
Location Lat Lon	No action but used for various bindings	lat 53.636684 lon 9.798321

# Persistence Configuration

## Charts rrd4j

Define the chart strategy & items to be displayed in the rrd4j charts.

### Information

rrd4j does save the data in an adjustable number of archives, each one does hold the data backwards from now for an adjustable timeframe.

The first archive always holds the data in a resolution of one entry per minute, each following archive is calculated from the entries in archive one by compressing a number of data points (using a “consolidation-function” like: AVERAGE, MAX, MIN, etc.).

When creating a chart from an rrd4j database for a specific timeframe, the data is fetched from the archive that covers the desired timeframe.

The setup of those archives is done in the file `etc/openhab2/services/rrd4j.cfg`.

If such a file doesn't exist openHAB will use a default setting:

[consolidation function: AVERAGE, archives (covering time / resolution) 1 (8 hrs / 1 min), 2 (24 hrs / 4 min), 3 (150:16 hrs / 14 min), 4 (30 days / 1 hr), 5 ( 365 days / 12 hrs), 6 (3640 days / 7 days) ]

*Credits: Thanks to openHAB community member [opus](#) for this information.*

## Persist Configuration

### Configuration Persistence File

```
/etc/openhab2/persistence/rrd4j.persist
```

### Configuration Persistence Content

```
Strategies {
    everyMinute : "0 * * * * ?"
    everyHour   : "0 0 * * * ?"
    everyDay    : "0 0 0 * * ?"
    default = everyChange }

Items {
    nHA_Weather_Home_Temperature : strategy = everyChange, everyMinute,
restoreOnStartup
    nHA_Weather_Home_WindSpeedBft : strategy = everyChange, everyMinute,
restoreOnStartup
    nHA_Weather_Home_Pressure : strategy = everyChange, everyMinute, restoreOnStartup
    nHA_LivingRoom_Temperature : strategy = everyChange, everyMinute,
restoreOnStartup
    nHA_LivingRoom_Humidity : strategy = everyChange, everyMinute, restoreOnStartup
    nHA_Energy_PowerConsumption : strategy = everyChange, everyMinute,
restoreOnStartup
    nHA_Info_HomepageCounter : strategy = everyChange, everyMinute, restoreOnStartup
    gHA_SystemInfoCharts* : strategy = everyChange, everyMinute, restoreOnStartup }
```

## Log Warning

Check the log after saving the configuration file for:

```
2018-01-04 12:18:49.241 [WARN ] [el.core.internal.ModelRepositoryImpl] -  
Configuration model 'rrd4j.persist' has errors, therefore ignoring it: [1,1]:  
mismatched input '<EOF>' expecting 'Strategies'
```

This is caused by the editor used, in this case VSCode, to change the configuration file.

## Workaround

The warning message did not occur when editing with **nano** from a terminal. All spaces starting at the beginning of a line have been replaced by proper tabs.

## Files

The rrd files created based on the previous Items defined are located in folder

```
/var/lib/openhab2/persistence/rrd4j
```

List the files

```
ls /var/lib/openhab2/persistence/rrd4j/*.rrd  
/var/lib/openhab2/persistence/rrd4j/nLivingRoom_Humidity.rrd  
/var/lib/openhab2/persistence/rrd4j/nLivingRoom_Temperature.rrd  
/var/lib/openhab2/persistence/rrd4j/nWeather_Home_Pressure.rrd  
/var/lib/openhab2/persistence/rrd4j/nWeather_Home_Temperature.rrd  
/var/lib/openhab2/persistence/rrd4j/nWeather_Home_WindSpeedBft.rrd
```

And more...

# Transform Configuration

The openHAB Transformation Services is used to transform values to other values or names. The transformation files used in the previous rules are located in folder `/etc/openhab2/transform`.

Several transformations are used and described accordingly in the Home Automation Function, i.e. Function Anemometer.

Few examples shared here.

## Scale Winddirection

Transforms the Degree Direction to a Cardinal Direction.

### Configuration Transformation File

```
/etc/openhab2/transform/winddirection.scale
```

### Configuration Transformation Content (SCALE)

```
[348.75..11.25]=N
[11.25..33.75]=NNE
[33.75..56.25]=NE
[56.25..78.75]=ENE
[78.75..101.25]=E
[101.25..123.75]=ESE
[123.75..146.25]=SE
[146.25..168.75]=SSE
[168.75..191.25]=S
[191.25..213.75]=SSW
[213.75..236.25]=SW
[236.25..258.75]=WSW
[258.75..281.25]=W
[281.25..303.75]=WNW
[303.75..326.25]=NW
[326.25..348.75]=NNW
[..]=Undef
```

### Rule

```
/*
  Weather - Windmesser Direction
*/
rule "Windmesser Direction Changed"
when
  Item nHA_Weather_WindSensor_DirectionDeg changed
then
  var String d = transform("SCALE", "winddirection.scale",
nHA_Weather_WindSensor_DirectionDeg.state.toString)
  sHA_Weather_WindSensor_Direction.postUpdate( d )
end
```

# Scale Windspeedmsbft

Transforms the Speed in m/s to Beaufort.

## Configuration Transformation File

```
/etc/openhab2/transform/windspeedkmsbft.scale
```

## Configuration Transformation Content (SCALE)

```
[0..0.2]=0  
[0.2..1.5]=1  
[1.5..3.3]=2  
[3.3..5.4]=3  
[5.4..7.9]=4  
[7.9..10.7]=5  
[10.7..13.8]=6  
[13.8..17.1]=7  
[17.1..20.7]=8  
[20.7..24.4]=9  
[24.4..28.4]=10  
[28.4..32.6]=11  
[32.6..]=12
```

## Rule

```
/*  
  Weather - Windmesser Speed in Bft  
*/  
rule "Windmesser Speed Changed"  
when  
  Item nHA_Weather_WindSensor_Speed changed  
then  
  var String bft = transform("SCALE", "windspeedmsbft.scale",  
nHA_Weather_WindSensor_Speed.state.toString)  
  nHA_Weather_WindSensor_Bft.postUpdate( bft )  
end
```

# Node-RED as Rules Engine

## Overview

[Node-RED](#) enables Flow-based programming for the Internet of Things.

If not familiar with the Node-RED concept, strongly recommend to read the Node-RED [get started](#) documentation.

### Node-RED

- Included in openHABian
- Accessed via browser URL
  - openhabianpi:1880 (to define the flows) and
  - openhabianpi:1880/ui (to access the dashboard, if the dashboard UI add-on is used)
- Stopped and started by using the console `$node-red-stop` and `$node-red-start`
- Modules are located in folder `/usr/lib/node_modules/node-red`
- Flows are stored in the folder `/home/openhabian/.node-red` in the file `flows_openHABianPi.json` (this can be changed in the Node-RED settings JSON file).
- Settings option enables to manage the palette and other things.
- openHAB2 Node-RED [add-on](#) is available, which nodes enable to listen to state changes or update openHAB items.  
With these features, Node-RED can perform all the control logic, i.e. functioning as an alternate openHAB Rules Engine (instead of using textual rules configuration).
- Install node packages (like `node-red-contrib-rfxcom`) via Settings > Manage Palette > Install.

**Node-RED** opens many possibilities. Just a few to mention:

- Instead of sending the state change to a Debug Node, the state can be assigned to a MQTT topic with payload using an mqtt out node. The mqtt out node published the message.  
A dedicated flow could be defined handling all MQTT topic definitions and updates, i.e. Item State Change > MQTT Topic + Payload < MQTT Publish
- The state change can also be send to a function node. In the function nodes certain tasks could be performed resulting in a new message to be created and send to another node.
- Use as Rules Engine.



## Update Node-RED

Important:

Save the flows file, i.e. flows\_openHABianPi.json, prior updating.

After updating, reboot the Raspberry Pi.

```
cd /usr/lib/node_modules/node-red
update-nodejs-and-nodered
```

### Log Example

```
Fetching Node-RED update.
This script will remove versions of Node.js prior to version 6.x of Node.js and
Node-RED
and if necessary replace them with Node.js 8.x LTS (carbon) and the latest Node-
RED from Npm.

It also moves any Node-RED nodes that are globally installed into your user
~/.node-red/node_modules directory, and adds them to your package.json, so that
you can manage them with the palette manager.

It also tries to run 'npm rebuild' to refresh any extra nodes you have installed
that may have a native binary component. While this normally works ok, you need
to check that it succeeds for your combination of installed nodes.
To do all this it runs commands as root - please satisfy yourself that this will
not damage your Pi, or otherwise compromise your configuration.
If in doubt please backup your SD card first.
Are you really sure you want to do this ? [y/N] ? y
Running Node-RED update for user openhabian at /home/openhabian
This can take 20-30 minutes on the slower Pi versions - please wait.
  Stop Node-RED                ✓
  Remove old version of Node-RED ✓
  Remove old version of Node.js -
  Update Node.js LTS           ✓   Node v6.13.1   Npm 3.10.10
  Clean npm cache              ✓
  Install Node-RED core        ✓   0.18.4
  Move global nodes to local   -
  Install extra Pi nodes       -
  Npm rebuild existing nodes   ✓
  Add menu shortcut            ✓
  Update systemd script        ✓
  Update update script         ✓

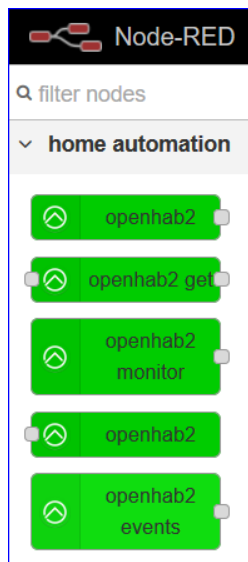
Any errors will be logged to   /var/log/nodered-install.log
All done.
  You can now start Node-RED with the command node-red-start
  or using the icon under   Menu / Programming / Node-RED
  Then point your browser to localhost:1880 or http://{your_pi_ip-address}:1880
Started Sat Mar 24 10:22:06 CET 2018 - Finished Sat Mar 24 10:30:14 CET 2018
```

## Node-RED openHAB2 Nodes

An openHAB2 Node-RED [add-on](#) is available, which nodes enable to listen to state changes or update openHAB items.

With these features, Node-RED can perform all the control logic, i.e. functioning as an alternate openHAB Rules Engine (instead of using textual rules configuration).

### The Node-RED palette with the openHAB2 Nodes



## Define the openHAB2 Controller Node

The openHAB2 nodes require an openHAB2 Controller Node. This node is added when adding for example an openhab2-in node (which listens to the state change of an item).

Edit openhab2-in node	
<div>Delete</div> <div>Cancel</div>	
▼ node properties	
Name	<input type="text" value="Name"/>
Controller	<div>Add new openhab2-controller...</div>
ItemName	<div></div>

The new openhab2-controller defined with name *OHController* and using defaults for the protocol (http), host (localhost) and port (8080) being used for openHAB.

Name	<input type="text" value="OHController"/>
Protocol	<input type="text" value="http"/>
Host	<input type="text" value="localhost"/>
Port	<input type="text" value="8080"/>
Path	<input type="text"/>
Username	<input type="text"/>
Password	<input type="text"/>

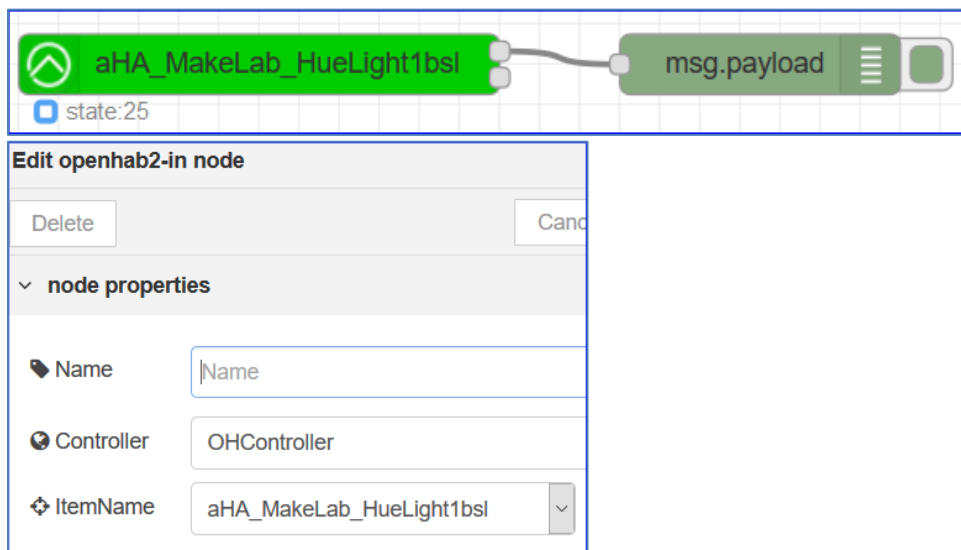
## Logging State Change MakeLab HueLight1

This flow listens to the state change of the item `aHA_MakeLab_HueLight1bsl`, which is a dimmer (slider in the sitemap) to control the Hue Light located in the MakeLab. The state is logged in the Node-RED Debug Tab.

**The item is defined as (file `/etc/openhab2/items/homeautomation.items`).**


Dimmer	aHA_MakeLab_HueLight1bsl	"Brightness"	<light>
(gHA_MakeLab)	{ channel="hue:0100:00178863da1d:1:brightness" }		


**The Node-RED Flow with the two nodes `openhab2-in` and `debug`.**




The item state is changed by moving the slider in the Sitemap from 0 to 25. State 25 is the value if the item.

### Make Lab

 Hue Make Lab

 Brightness



Slider moving from 0 to 25 is logged in the Node-RED Debug Tab by the Debug Node displaying the message payload.

info

debug

dash

all nodes

2/25/2018, 7:32:56 PM node: 6377ffa2.201a28

msg.payload : string[1]

"0"

2/25/2018, 7:33:11 PM node: 6377ffa2.201a28

msg.payload : string[2]

"25"

## Node-RED as MQTT Publisher

Instead of defining MQTT topics, a Node-RED flow is an alternative to publish MQTT topics with payload.

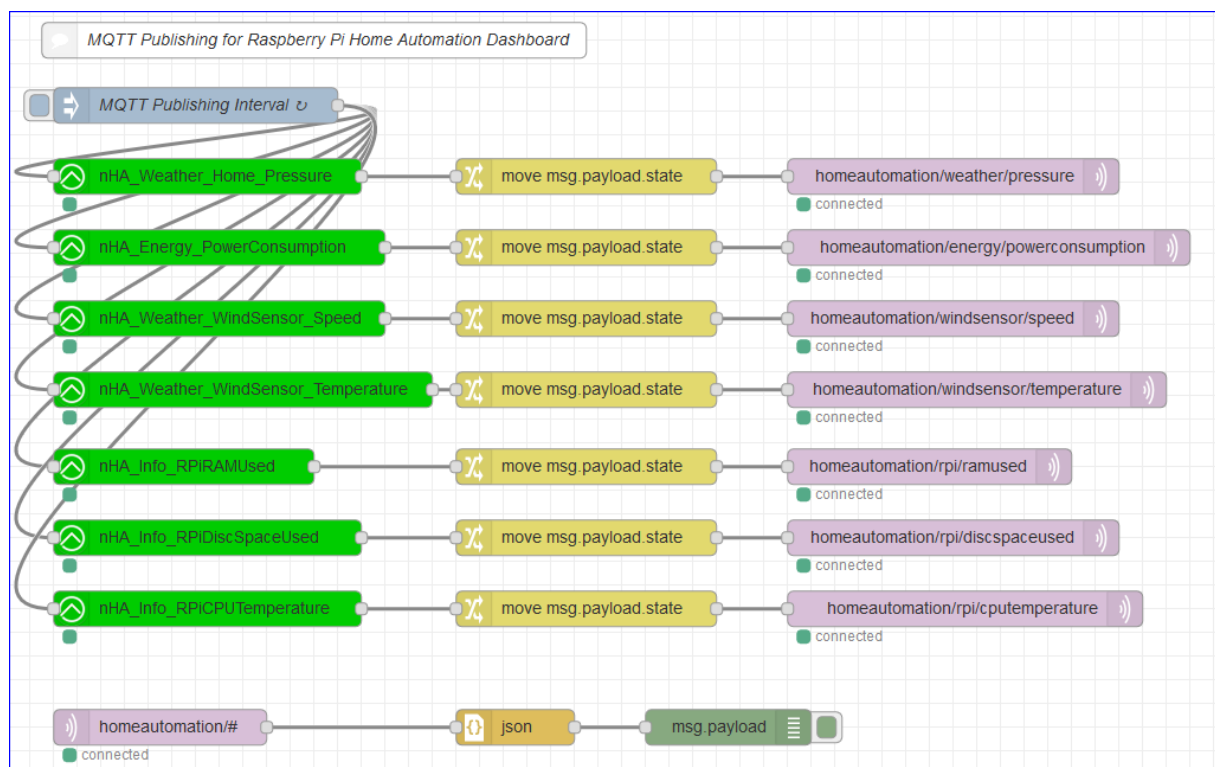
This flow example reads every minute sensor data from openhab2-get nodes (Gets an openHAB Item on an input message).

The openhab2-get node payload JSON property state contains the value of the sensor, i.e. state 1014 in this message:

```
{ "link": "http://localhost:8080/rest/items/nHA_Weather_Home_Pressure", "state": "1014.00", "stateDescription": { "pattern": "%d hPa", "readOnly": false, "options": [], "editable": false, "type": "Number", "name": "nHA_Weather_Home_Pressure", "label": "Pressure", "category": "pressure", "tags": [], "groupNames": ["gHA_Weather_Home"]} }
```

The change node moves the msg.payload.state to msg.payload, which is published by the mqtt-out node.

In summary the value of openHAB item nHA\_Weather\_Home\_Pressure is MQTT published as topic homeautomation/weather/pressure.



The mqtt-in node listens to homeautomation topics, converts the JSON string to its JavaScript object representation.

## Dashboard UI

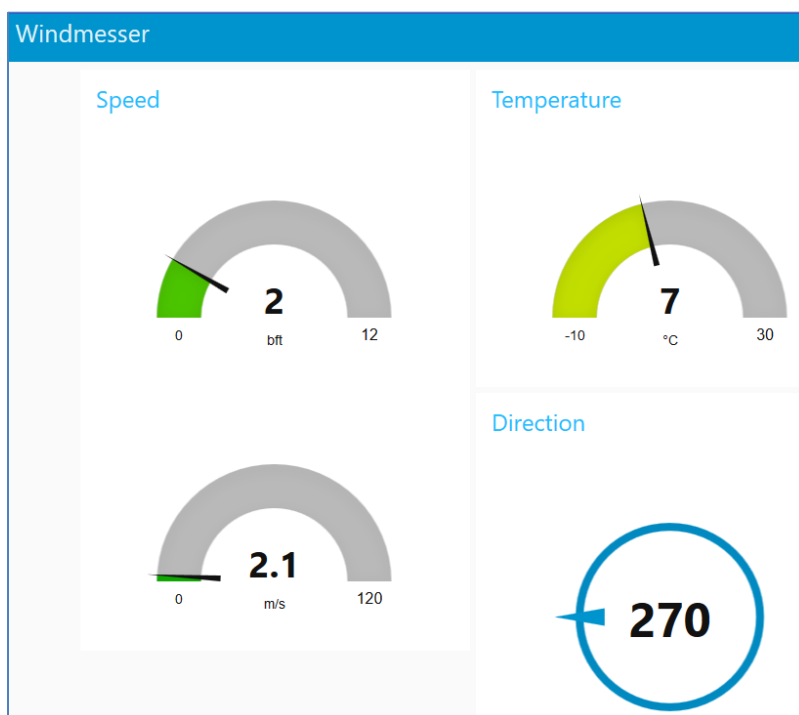
The Node-RED [Dashboard](#) module offers UI nodes to build a Dashboard.

Build a simple example of a dashboard using the Anemometer (“Windmesser”) data wind speed (m/s), wind speed (bft), wind direction (deg), temperature (°C).

## Dashboard Gauges

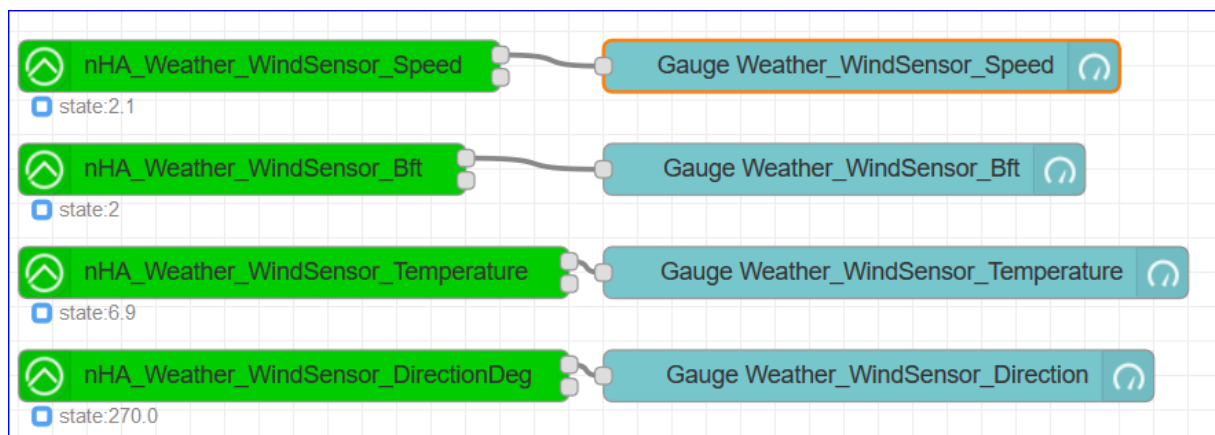
(URL: [openhabianpi:1880/ui/#/0](http://openhabianpi:1880/ui/#/0))

### Dashboard



### Flow Process

(URL: [openhabianpi:1880](http://openhabianpi:1880))



The openhab2-in nodes are using selective Items for the WindSensor as defined in `/etc/openhab2/items/homeautomation.items`.

```
Group gHA_Weather_Home_WindSensor "Windmesser Gartenhaus" <wind>
Number nHA_Weather_WindSensor_DirectionDeg "Direction [%0.0f deg]" <compass> {
channel="rfxcom:wind:e64474fc:40214:windDirection" }
Number nHA_Weather_WindSensor_Speed "Speed [%0.1f m/s]" <speed>
(gHA_Weather_Home_WindSensor) { channel="rfxcom:wind:e64474fc:40214:windSpeed" }
Number nHA_Weather_WindSensor_Bft "Bft [%0.0f]" <speed>
(gHA_Weather_Home_WindSensor)
Number nHA_Weather_WindSensor_Temperature "Temperature [%0.1f C]" <temperature>
(gHA_Weather_Home_WindSensor) { channel="rfxcom:wind:e64474fc:40214:temperature" }
```

## Flow JSON

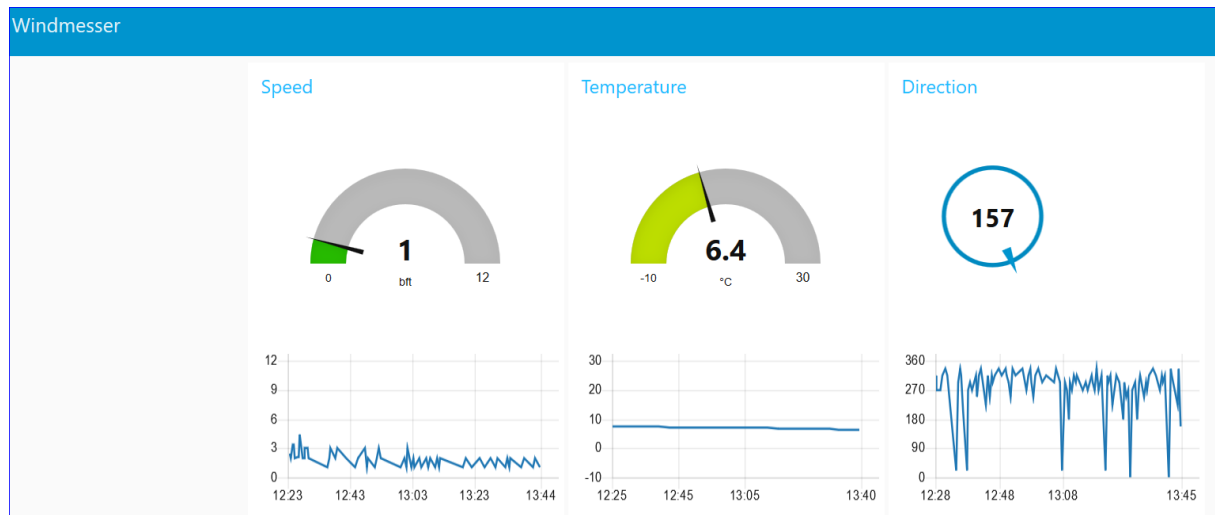
The flow exported in JSON format.

```
[{"id":"ad1ca8e2.55e9f8","type":"openhab2-in","z":"b65617da.c10108","name":"","controller":"28f21eba.d33c62","itemname":"nHA_Weather_WindSensor_Speed","x":180,"y":140,"wires":[["9f4b76d4.26c4c8"],[]]},{id:"9f4b76d4.26c4c8","type":"ui_gauge","z":"b65617da.c10108","name":"Gauge Weather_WindSensor_Speed","group":"1690f0cc.e6cd17","order":2,"width":0,"height":0,"gtype":"gage","title":"","label":"","format":"{{value | number:1}}","min":0,"max":120,"colors":["#00b500","#e6e600","#ca3838"],"seg1":"","seg2":"","x":530,"y":140,"wires":[[]]},{id:"40867510.574da4","type":"openhab2-in","z":"b65617da.c10108","name":"","controller":"28f21eba.d33c62","itemname":"nHA_Weather_WindSensor_Bft","x":170,"y":200,"wires":[["33dc2864.ddbd4"],[]]},{id:"33dc2864.ddbd4","type":"ui_gauge","z":"b65617da.c10108","name":"Gauge Weather_WindSensor_Bft","group":"1690f0cc.e6cd17","order":1,"width":0,"height":0,"gtype":"gage","title":"","label":"bft","format":"{{value | number:0}}","min":0,"max":12,"colors":["#00b500","#e6e600","#ca3838"],"seg1":"","seg2":"","x":520,"y":200,"wires":[[]]},{id:"840925ea.56b308","type":"ui_gauge","z":"b65617da.c10108","name":"Gauge Weather_WindSensor_Temperature","group":"f3715001.062b5","order":0,"width":0,"height":0,"gtype":"gage","title":"","label":"°C","format":"{{value | number:0}}","min":-10,"max":30,"colors":["#00b500","#e6e600","#ca3838"],"seg1":"","seg2":"","x":550,"y":260,"wires":[[]]},{id:"78694d1a.2048ac","type":"openhab2-in","z":"b65617da.c10108","name":"","controller":"28f21eba.d33c62","itemname":"nHA_Weather_WindSensor_Temperature","x":200,"y":260,"wires":[["840925ea.56b308"],[]]},{id:"14e25012.9f8af8","type":"ui_gauge","z":"b65617da.c10108","name":"Gauge Weather_WindSensor_Direction","group":"89d7eb3.dc4f018","order":0,"width":0,"height":0,"gtype":"compass","title":"","label":"","format":"{{value | number:0}}","min":0,"max":360,"colors":["#00b500","#e6e600","#ca3838"],"seg1":"","seg2":"","x":540,"y":320,"wires":[[]]},{id:"ca28272c.46f18","type":"openhab2-in","z":"b65617da.c10108","name":"","controller":"28f21eba.d33c62","itemname":"nHA_Weather_WindSensor_DirectionDeg","x":200,"y":320,"wires":[["14e25012.9f8af8"],[]]},{id:"28f21eba.d33c62","type":"openhab2-controller","z":"","name":"OHController","protocol":"http","host":"localhost","port":"8080","path":"","username":"","password":"","id":"1690f0cc.e6cd17","type":"ui_group","z":"","name":"Speed","tab":"fb9c2682.2d526","disp":true,"width":6,"collapse":false},{id:"f3715001.062b5","type":"ui_group","z":"","name":"Temperature","tab":"fb9c2682.2d526","disp":true,"width":6,"collapse":false},{id:"89d7eb3.dc4f018","type":"ui_group","z":"","name":"Direction","tab":"fb9c2682.2d526","disp":true,"width":6,"collapse":false},{id:"fb9c2682.2d526","type":"ui_tab","z":"","name":"Windmesser","icon":"dashboard"}]
```

## Dashboard Gauges & Charts

The dashboard module includes also simple charts. Applying to the Items wind speed bft, wind direction deg and temperature for a period of 1 day.

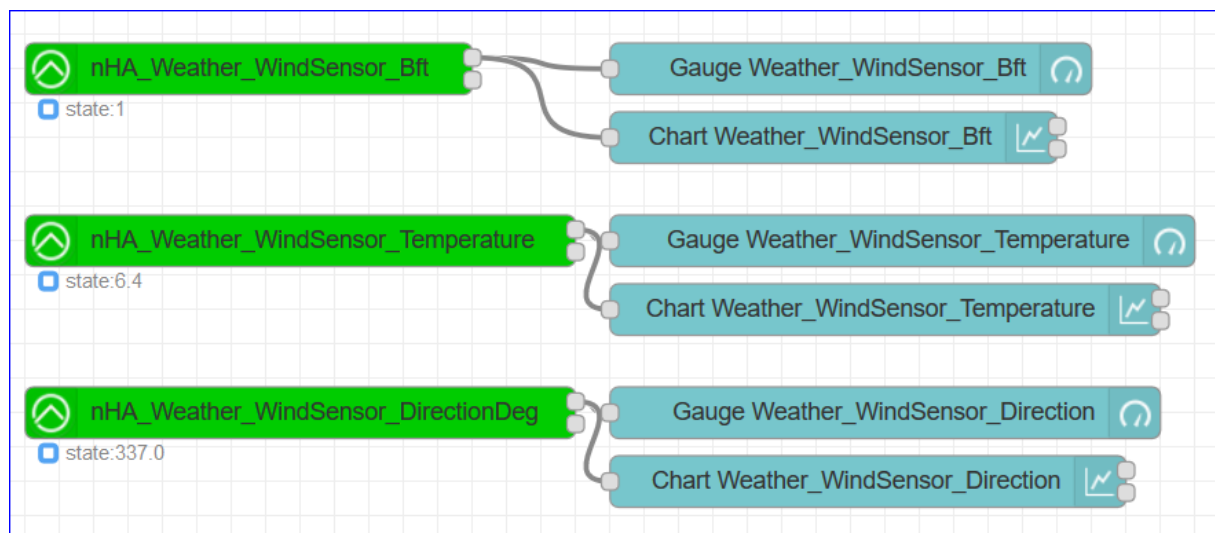
### Dashboard



*Notes:*

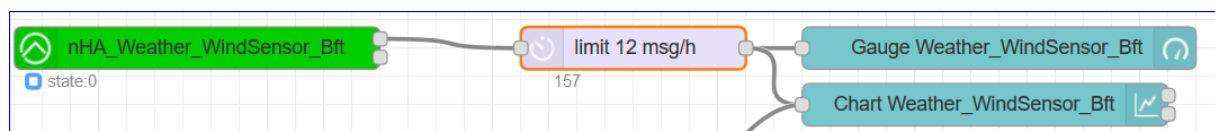
The charts only show few hours of data as an illustration.

### Flow Process



*Notes:*

The charts display each data change which could result in a rather frequent update. To avoid many updates, use a Delay Node which limits the number of messages to for example 12 messages per hour (Properties: "pauseType": "rate", "rate": "12", "nbRateUnits": "1", "rateUnits": "hour").



## Flow JSON

```
[{"id": "40867510.574da4", "type": "openhab2-in", "z": "b65617da.c10108", "name": "", "controller": "28f21eba.d33c62", "itemname": "nHA_Weather_WindSensor_Bft", "x": 170, "y": 100, "wires": [{"33dc2864.ddbd4", "b2eb7248.7d9aa", []}], {"id": "33dc2864.ddbd4", "type": "ui_gauge", "z": "b65617da.c10108", "name": "Gauge Weather_WindSensor_Bft", "group": "1690f0cc.e6cd17", "order": 1, "width": 0, "height": 0, "gtype": "gage", "title": "", "label": "bft", "format": "{{value | number:0}}", "min": 0, "max": 12, "colors": ["#00b500", "#e6e600", "#ca3838"], "seg1": "", "seg2": "", "x": 520, "y": 100, "wires": []}, {"id": "840925ea.56b308", "type": "ui_gauge", "z": "b65617da.c10108", "name": "Gauge Weather_WindSensor_Temperature", "group": "f3715001.062b5", "order": 0, "width": 0, "height": 0, "gtype": "gage", "title": "", "label": "°C", "format": "{{value | number:1}}", "min": -10, "max": 30, "colors": ["#00b500", "#e6e600", "#ca3838"], "seg1": "", "seg2": "", "x": 550, "y": 200, "wires": []}, {"id": "78694d1a.2048ac", "type": "openhab2-in", "z": "b65617da.c10108", "name": "", "controller": "28f21eba.d33c62", "itemname": "nHA_Weather_WindSensor_Temperature", "x": 200, "y": 200, "wires": [{"840925ea.56b308", "2b1aac06.0f3f7c", []}], {"id": "14e25012.9f8af8", "type": "ui_gauge", "z": "b65617da.c10108", "name": "Gauge Weather_WindSensor_Direction", "group": "89d7eb3.dc4f018", "order": 0, "width": 4, "height": 4, "gtype": "compass", "title": "", "label": "", "format": "{{value | number:0}}", "min": 0, "max": 360, "colors": ["#00b500", "#e6e600", "#ca3838"], "seg1": "", "seg2": "", "x": 540, "y": 300, "wires": []}, {"id": "ca28272c.46f18", "type": "openhab2-in", "z": "b65617da.c10108", "name": "", "controller": "28f21eba.d33c62", "itemname": "nHA_Weather_WindSensor_DirectionDeg", "x": 200, "y": 300, "wires": [{"14e25012.9f8af8", "9b4517e2.6705d", []}], {"id": "b2eb7248.7d9aa", "type": "ui_chart", "z": "b65617da.c10108", "name": "Chart Weather_WindSensor_Bft", "group": "1690f0cc.e6cd17", "order": 0, "width": 0, "height": 0, "label": "", "chartType": "line", "legend": "false", "xformat": "HH:mm", "interpolate": "linear", "nodata": "", "dot": false, "ymin": 0, "ymax": 12, "removeOlder": 1, "removeOlderPoints": "", "removeOlderUnit": "86400", "cutout": 0, "useOneColor": false, "colors": ["#1f77b4", "#aec7e8", "#ff7f0e", "#2ca02c", "#98df8a", "#d62728", "#ff9896", "#9467bd", "#c5b0d5"], "useOldStyle": false, "x": 510, "y": 140, "wires": [], {"id": "2b1aac06.0f3f7c", "type": "ui_chart", "z": "b65617da.c10108", "name": "Chart Weather_WindSensor_Temperature", "group": "f3715001.062b5", "order": 0, "width": 0, "height": 0, "label": "", "chartType": "line", "legend": "false", "xformat": "HH:mm", "interpolate": "linear", "nodata": "", "dot": false, "ymin": -10, "ymax": 30, "removeOlder": 1, "removeOlderPoints": "", "removeOlderUnit": "86400", "cutout": 0, "useOneColor": false, "colors": ["#1f77b4", "#aec7e8", "#ff7f0e", "#2ca02c", "#98df8a", "#d62728", "#ff9896", "#9467bd", "#c5b0d5"], "useOldStyle": false, "x": 540, "y": 240, "wires": [], {"id": "9b4517e2.6705d", "type": "ui_chart", "z": "b65617da.c10108", "name": "Chart Weather_WindSensor_Direction", "group": "89d7eb3.dc4f018", "order": 0, "width": 0, "height": 0, "label": "", "chartType": "line", "legend": "false", "xformat": "HH:mm", "interpolate": "linear", "nodata": "", "dot": false, "ymin": 0, "ymax": 360, "removeOlder": 1, "removeOlderPoints": "", "removeOlderUnit": "86400", "cutout": 0, "useOneColor": false, "colors": ["#1f77b4", "#aec7e8", "#ff7f0e", "#2ca02c", "#98df8a", "#d62728", "#ff9896", "#9467bd", "#c5b0d5"], "useOldStyle": false, "x": 530, "y": 340, "wires": [], {"id": "28f21eba.d33c62", "type": "openhab2-controller", "z": "", "name": "OHController", "protocol": "http", "host": "localhost", "port": "8080", "path": "", "username": "", "password": "", {"id": "1690f0cc.e6cd17", "type": "ui_group", "z": "", "name": "Speed", "tab": "fb9c2682.2d526", "disp": true, "width": "6", "collapse": false}, {"id": "f3715001.062b5", "type": "ui_group", "z": "", "name": "Temperature", "tab": "fb9c2682.2d526", "disp": true, "width": "6", "collapse": false}, {"id": "89d7eb3.dc4f018", "type": "ui_group", "z": "", "name": "Direction", "tab": "fb9c2682.2d526", "disp": true, "width": "6", "collapse": false}, {"id": "fb9c2682.2d526", "type": "ui_tab", "z": "", "name": "Windmesser", "icon": "dashboard"}]
```



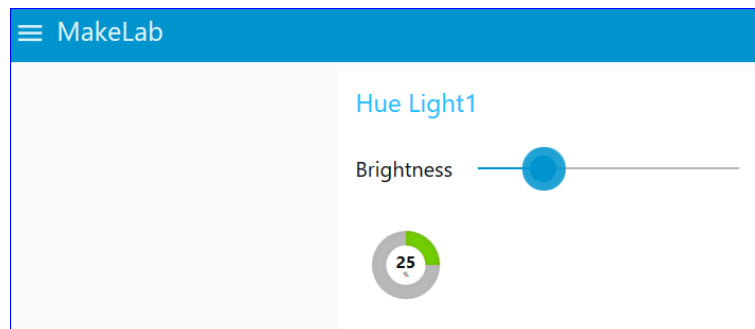
## Control Brightness MakeLab HueLight1

Example controlling the Brightness of the Hue Light located in the MakeLab by using Dashboard UI nodes Slider (to set the brightness between 0- 100%) and a Gauge (to display the Brightness).

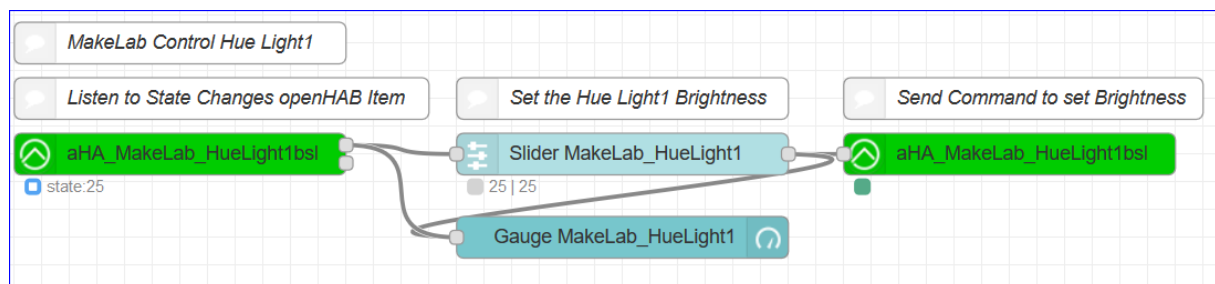
Both the slider and the gauge are updated when the brightness is changed by another option, i.e. openHAB Basic UI or openHAB Android App.

The Dashboard UI tab is called MakeLab with group Hue Light1.

### Dashboard



### Flow Process



### Notes

The openHAB controller used as defined previous for the openhab2-in and out nodes.

The openhab2-in node updates the slider and the gauge. The slider does not pass the input value to the output.

### Flow JSON

```
[{"id":"c5506ebb.7c439","type":"ui_slider","z":"c0c69178.56fba","name":"Slider MakeLab_HueLight1","label":"Brightness","group":"40590dc2.e527a4","order":0,"width":0,"height":0,"passthru":false,"topic":"","min":0,"max":100,"step":5,"x":460,"y":120,"wires":[["b7299df.bee6a6","df9b172.ae035e8"]]}, {"id":"b7299df.bee6a6","type":"openhab2-out","z":"c0c69178.56fba","name":"","controller":"28f21eba.d33c62","itemname":"aHA_MakeLab_HueLight1bsl","topic":"ItemCommand","payload":"","x":740,"y":120,"wires":[]}, {"id":"abd63258.98215","type":"openhab2-in","z":"c0c69178.56fba","name":"","controller":"28f21eba.d33c62","itemname":"aHA_MakeLab_HueLight1bsl","x":140,"y":120,"wires":[["c5506ebb.7c439","df9b172.ae035e8"]]}, {"id":"df9b172.ae035e8","type":"ui_gauge","z":"c0c69178.56fba","name":"Gauge MakeLab_HueLight1","group":"40590dc2.e527a4","order":0,"width":2,"height":2,"gtype":"donut","title":"","label":"%", "format":"{{value}}","min":0,"max":100,"colors":["#00b500","#e6e600","#ca3838"],"seg1":"","seg2":"","x":460,"y":180,"wires":[]}, {"id":"dc3fcb5c.cb1228","type":"comment","z":"c0c69178.56fba","name":"MakeLab Control Hue Light1","info":"","x":140,"y":40,"wires":[]}, {"id":"95880861.8c9d48","type":"comment","z":"c0c69178.56fba","name":"Listen to State Changes openHAB Item","info":"","x":170,"y":80,"wires":[]}, {"id":"2e5bcc8e.6f3afc","type":"comment","z":"c0c69178.56fba","name":"Set the Hue Light1"}]
```

```
Brightness","info":"","x":460,"y":80,"wires":[]},{ "id":"6999226b.3c121c","type":"comment","z":"c0c69178
.56fba","name":"Send Command to set
Brightness","info":"","x":750,"y":80,"wires":[]},{ "id":"40590dc2.e527a4","type":"ui_group","z":"","name
":"Hue
Light1","tab":"d292d089.4a1c68","disp":true,"width":"6","collapse":false},{ "id":"28f21eba.d33c62","type
":"openhab2-
controller","z":"","name":"OHController","protocol":"http","host":"localhost","port":"8080","path":"","
username":"","password":""},{ "id":"d292d089.4a1c68","type":"ui_tab","z":"","name":"MakeLab","icon":"das
hboard"}]
```

# Functions Introduction

This Home Automation solution has a modular setup, which are called **functions**.

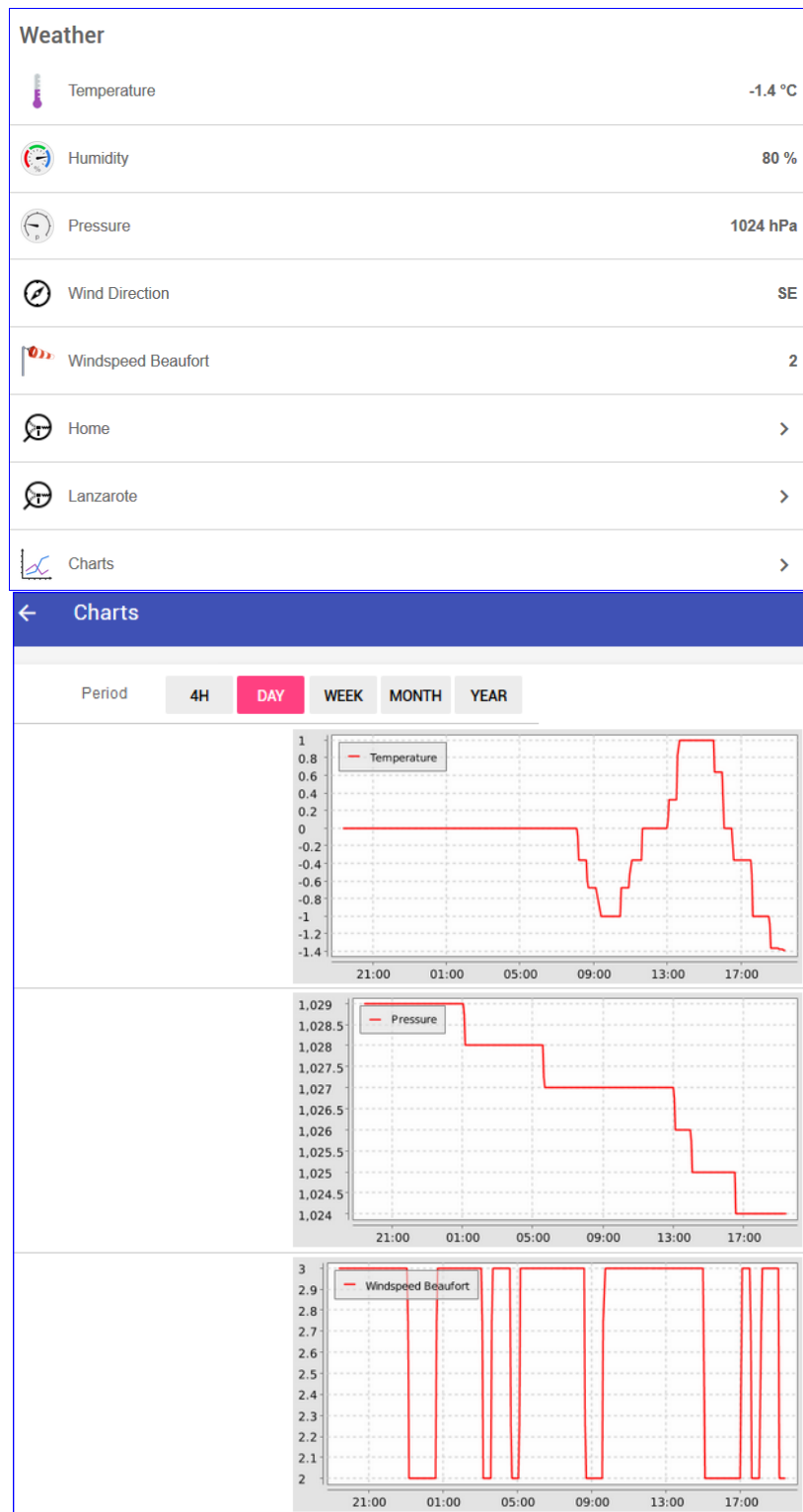
Each function

- has a specific goal
- makes selective use of the bindings described previous
- is defined in textual configuration for the Items, Sitemap and Rules etc.

# Function Weather

## Goal

Obtain weather information from OpenWeatherMap – see Weather Binding (OpenWeatherMap Service) - service and display selective data and charts.



## Items Configuration

Define a Group and assign selective weather data using channels.

*Notes*

Not all weather data is listed below – full list in items configuration file  
/etc/openhab2/homeautomation.items.

```
Group gHA_Weather_Home "Home" <details>
// Weather with data from OpenWeatherMap 15 minutes
DateTime dtHA_Weather_Home_ObservationTime "Observation time [%1$td.%1$tm.%1$tY
%1$tH:%1$tM]" <time> (gHA_WeatherHome) {weather="locationId=home, type=condition,
property=observationTime"}
Number nHA_Weather_Home_Temperature "Temperature [%1f °C]" <temperature>
(gHA_WeatherHome) {weather="locationId=home, type=temperature, property=current"}
Number nHA_Weather_Home_Humidity "Humidity [%d %%]" <humidity>
(gHA_WeatherHome) {weather="locationId=home, type=atmosphere, property=humidity"}
Number nHA_Weather_Home_Pressure "Pressure [%d hPa]" <pressure>
(gHA_WeatherHome) {weather="locationId=home, type=atmosphere, property=pressure"}
...
```

## Sitemap Configuration

The sitemap uses the previous group defined plus the chart configuration for selective data  
Temperature, Humidity and Pressure.

*Notes*

Not all weather data is listed – full list in items configuration file  
/etc/openhab2/homeautomation.sitemap.

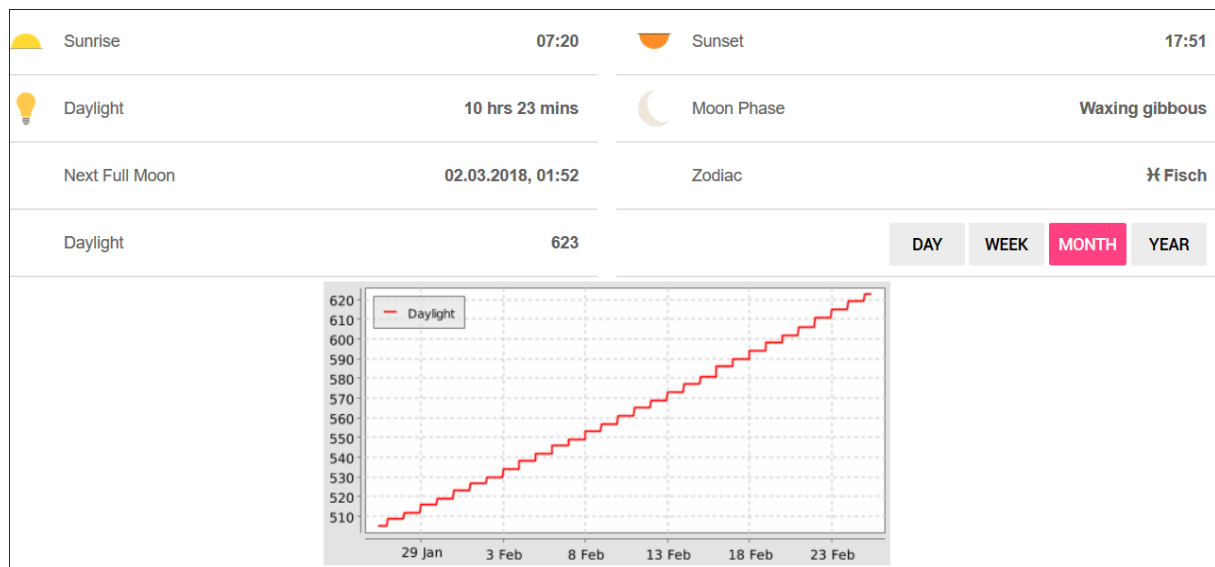
```
Frame label="Weather" {
  Text item=nHA_Weather_Home_Temperature
  Text item=nHA_Weather_Home_Humidity
  Text item=nHA_Weather_Home_Pressure
  Text item=nHA_Weather_Home_WindDirection
  Text item=nHA_Weather_Home_WindSpeedBft
  Group item=gHA_Weather_Home
  Group item=gHA_Weather_Lanzarote
  Text label="Charts" icon="line-incline" {
    Switch item=nHA_ChartPeriod_WeatherHome
    mappings=[0="4H",1="Day",2="Week",3="Month", 4="Year"]
    Chart item=nHA_Weather_Home_Temperature period=4h refresh=60000
    legend=true
    visibility=[nHA_ChartPeriod_WeatherHome==0,nHA_ChartPeriod_WeatherHome=="NULL"]
    ...
  }
}
```

# Function Astro Information

## Goal

To display Astro data sun rise, sun set, daylight duration, moon phase and other info.

The Things defined via the Paper-UI are the **Astro moon data** and **Astro sun data** for the **location lat lon 53.636684,9.798321**.



## Item Configuration

```

Group      gHA_Astro
DateTime   dtHA_Astro_Sunrise_Time      "Sunrise [%1$tH:%1$tM]"
<sunrise> (gHA_Astro) {channel="astro:sun:cb065030:rise#start"}
DateTime   dtHA_Astro_Sunset_Time       "Sunset [%1$tH:%1$tM]"          <sunset>
(gHA_Astro) {channel="astro:sun:cb065030:set#end"}
String     sHA_Astro_MoonPhase_Name     "Moon Phase [%s]"          <moon>
(gHA_Astro) {channel="astro:moon:e5b32a75:phase#name"}
DateTime   dtHA_Astro_Moon_Next_Full     "Next Full Moon [%1$td.%1$tm.%1$tY,
%1$tH:%1$tM]" <moon> (gHA_Astro) {channel="astro:moon:e5b32a75:phase#full"}
DateTime   dtHA_Astro_Moon_Next_New      "Next New Moon [%1$td.%1$tm.%1$tY,
%1$tH:%1$tM]" <moon> (gHA_Astro) {channel="astro:moon:e5b32a75:phase#new"}
Number     nHA_Astro_DayLight            "Daylight [JS(daylight.js):%s]" <light>
(gHA_Astro) {channel="astro:sun:cb065030:daylight#duration"}
Number     nHA_Astro_DayLight_Minutes    "Daylight [%d]"
(gHA_Astro) {channel="astro:sun:cb065030:daylight#duration"}
String     sHA_Astro_Zodiac_Sign         "Zodiac [MAP(astro.map):%s]" <zodiac>
(gHA_Astro) {channel="astro:sun:cb065030:zodiac#sign"}
Number     nHA_ChartPeriod_Weather_Astro " "
  
```

# Transformation Configuration

## Daylight

The Daylight, in hours and minutes is calculated using JavaScript Transformation with the number of minutes as input. The function returns a string “NN hrs NN mins”.

The input is taken from the channel astro:sun:cb065030:daylight#duration.

The function “daylight.js” is called by Item nHA\_Astro\_DayLight:

```
Number nHA_Astro_DayLight "Daylight [JS(daylight.js):%s]"<light>
(gHA_Weather_Home) {channel="astro:sun:cb065030:daylight#duration"}
```

### Transformation Configuration File

```
/etc/openhab2/transform/daylight.js
```

### Transformation Configuration Content (JS)

```
(function(minutes) {
  var hours    = Math.floor(minutes / 60);
  minutes      -= Math.floor(hours * 60);
  if (hours <= 0) {
    return "0 hrs " + minutes + " mins";
  }
  return hours + " hrs " + minutes + " mins";
})(input)
```

## Astro

The mapping for Astro items are taken from [this](#) tutorial (many thanks to the author). Important: The mapping needs to be saved in an UTF8 encoded text file as the mapping contains Unicode symbols.

### Transformation Configuration File

```
/etc/openhab2/transform/astro.map
```

### Transformation Configuration Content (MAP)

```
//
ARIES=♈ Widder
TAURUS=♉ Stier
GEMINI=♊ Zwilling
CANCER=♋ Krebs
LEO=♌ Löwe
VIRGO=♍ Jungfrau
LIBRA=♎ Waage
SCORPIO=♏ Skorpion
SAGITTARIUS=♐ Schütze
CAPRICORN=♑ Steinbock
```

```

AQUARIUS=♈ Wassermann
PISCES=♉ Fisch

//
SPRING=Frühling
SUMMER=Sommer
AUTUMN=Herbst
WINTER=Winter

//
SUN_RISE=Sonnenaufgang
ASTRO_DAWN=astronomische Morgendämmerung
NAUTIC_DAWN=nautische Morgendämmerung
CIVIL_DAWN=zivile Morgendämmerung
CIVIL_DUSK=zivile Abenddämmerung
NAUTIC_DUSK=nautische Abenddämmerung
ASTRO_DUSK=astronomische Abenddämmerung
SUN_SET=Sonnenuntergang
DAYLIGHT=Tag
NOON=Abend
NIGHT=Nacht

//
NEW=● Neumond
WAXING_CRESCENT=●→○ zunehmender Halbmond
FIRST_QUARTER=○ erstes Viertel
WAXING_GIBBOUS=○→○ zunehmender Mond
FULL=○ Vollmond
WANING_GIBBOUS=○→● abnehmender Mond
THIRD_QUARTER=● letztes Viertel
WANING_CRESCENT=●→● abnehmender Halbmond

//
NULL=unbekannt !?
-=unbekannt !?

```

## Sitemap Configuration

The Astro data are displayed as separate Group gHA\_Astro.  
In addition, the daylight in minutes is displayed as a chart.

```

Text label="Astro" icon="astro" {
  Text item=dtHA_Astro_Sunrise_Time
  Text item=dtHA_Astro_Sunset_Time
  Text item=nHA_Astro_DayLight
  Text item=sHA_Astro_MoonPhase_Name
  Text item=dtHA_Astro_Moon_Next_Full icon="fullmoon"
    visibility=[sHA_Astro_MoonPhase_Name == WAXING_CRESCENT,
      sHA_Astro_MoonPhase_Name == FIRST_QUARTER,
      sHA_Astro_MoonPhase_Name == WAXING_GIBBOUS,
      sHA_Astro_MoonPhase_Name == FULL]
  Text item=dtHA_Astro_Moon_Next_New icon="fullmoon"
    visibility=[sHA_Astro_MoonPhase_Name == WANING_GIBBOUS,
      sHA_Astro_MoonPhase_Name == THIRD_QUARTER,
      sHA_Astro_MoonPhase_Name == WANING_CRESCENT,
      sHA_Astro_MoonPhase_Name == NEW]
  Text item=sHA_Astro_Zodiac_Sign
}

```



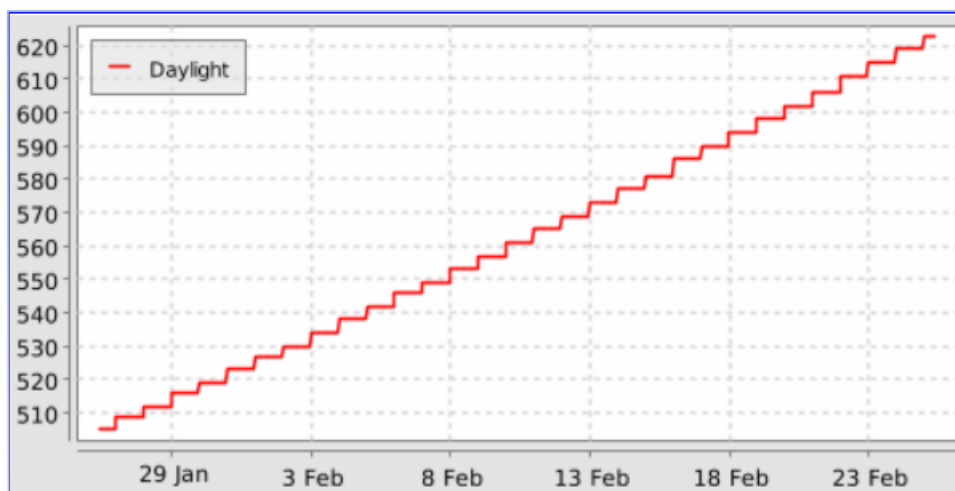
```

Text item=nHA_Astro_DayLight_Minutes
Switch item=nHA_ChartPeriod_Astro mappings=[0="Day",1="Week",2="Month",3="Year"]
  Chart item=nHA_Astro_DayLight_Minutes period=D refresh=60000
legend=true visibility=[nHA_ChartPeriod_Astro==0,nHA_ChartPeriod_Astro=="NULL"]
  Chart item=nHA_Astro_DayLight_Minutes period=W refresh=60000
legend=true visibility=[nHA_ChartPeriod_Astro==1]
  Chart item=nHA_Astro_DayLight_Minutes period=M refresh=60000
legend=true visibility=[nHA_ChartPeriod_Astro==2]
  Chart item=nHA_Astro_DayLight_Minutes period=Y refresh=60000
legend=true visibility=[nHA_ChartPeriod_Astro==3]
}

```

## Chart Configuration

The daylight in minutes is displayed in a chart.



## rrd4j Configuration Entry

Configuration file `/etc/openhab2/persistence/rrd4j.persist`

```

Items
{
...
nHA_Astro_DayLight_Minutes : strategy = everyChange, everyMinute, restoreOnStartup
...
}

```

Sitemap

See previous under Sitemap.

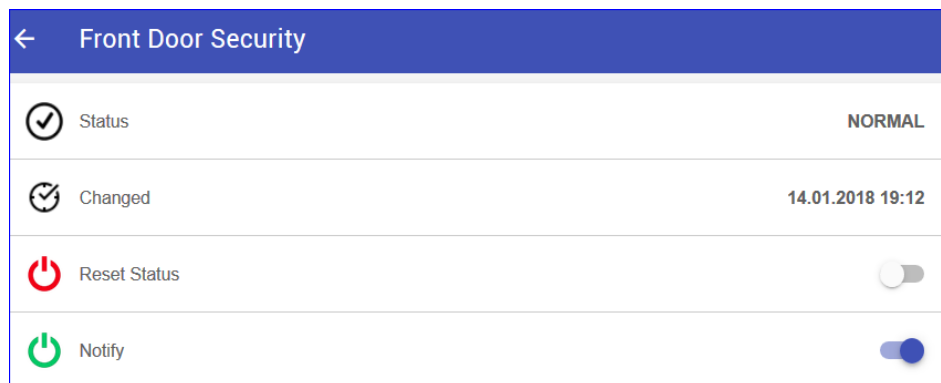
# Function Security Front Door

## Goal

To watch the Front Door, open & close and notify users via email.

This function can be applied to any contact.

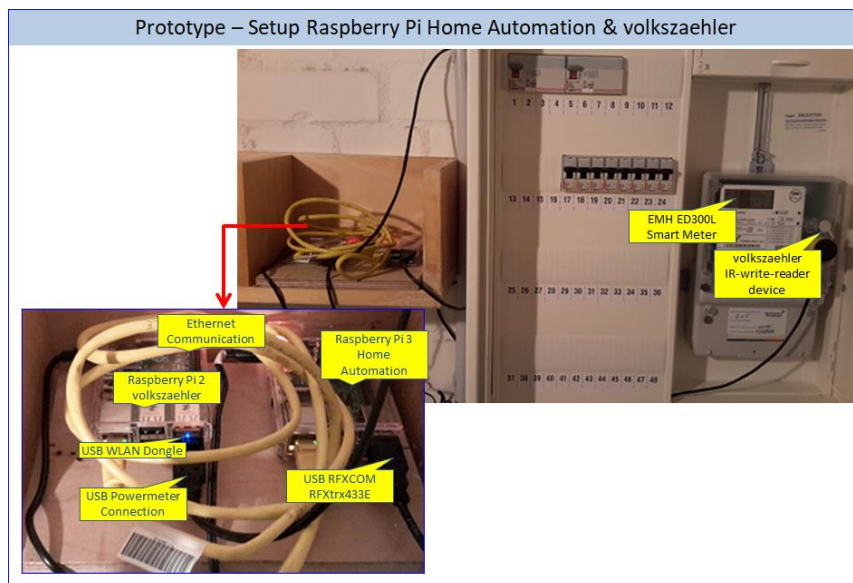
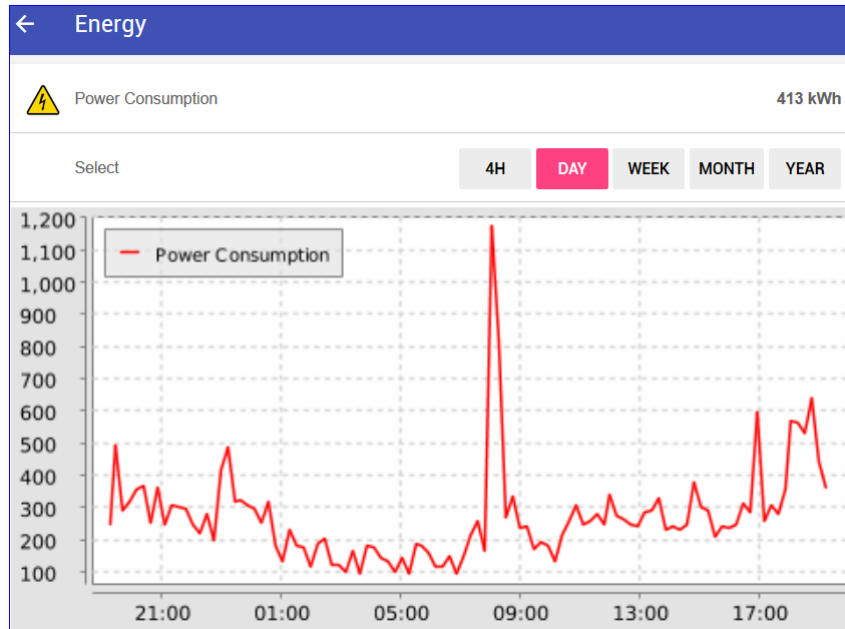
See Thing FrontDoor-Security (PB62R) how this function has been setup.



## Function Power Consumption

## Goal

Measure the Energy Power Consumption (kWh) obtained from a **Volkszaehler** - Open Source Smart Meter, connected to a Raspberry Pi 2.



The power consumption is requested every 5 minutes via a HTTP GET Request. The requests uses a **from** timestamp which is **now – 5 minutes** and a **to** timestamp which is **now**.

## Items Configuration

```
Number nHA_Energy_PowerConsumption "Power Consumption [%.0f kWh]" <energy>
Number nHA_ChartPeriod_PowerConsumption "Period"
```

## Sitemap Configuration

```
Text label="Energy" icon="energy" {
  Text item=nHA_Energy_PowerConsumption
  Switch item=nHA_ChartPeriod_PowerConsumption
  mappings=[0="4H",1="Day",2="Week",3="Month",4="Year"]
  Chart item=nHA_Energy_PowerConsumption period=4h refresh=60000 legend=true
  visibility=[nHA_ChartPeriod_PowerConsumption==0,nHA_ChartPeriod_PowerConsumption==
  "NULL"]
  Chart item=nHA_Energy_PowerConsumption period=D refresh=60000 legend=true
  visibility=[nHA_ChartPeriod_PowerConsumption==1]
  Chart item=nHA_Energy_PowerConsumption period=W refresh=60000 legend=true
  visibility=[nHA_ChartPeriod_PowerConsumption==2]
  Chart item=nHA_Energy_PowerConsumption period=M refresh=60000 legend=true
  visibility=[nHA_ChartPeriod_PowerConsumption==3]
  Chart item=nHA_Energy_PowerConsumption period=Y refresh=60000 legend=true
  visibility=[nHA_ChartPeriod_PowerConsumption==4]
}
```

## Persistence Configuration

Add the Item HA\_Energy\_PowerConsumption to the rrd4j persistconfiguration  
(/etc/openhab2/persis/rrd4j.persist)

```
Items
{
  nHA_Energy_PowerConsumption : strategy = everyChange, everyMinute,
  restoreOnStartup
}
```

## Rules Configuration

```
/*
  Energy Power Consumption Updates every 5 minutes
  Note
  To get the power consumption from now to now use as parameter ?from=now&To=now
*/
rule "Energy Power Consumption Update"
when
  Time cron "0 */5 * ? * *"
then
  // Set the from time = now - 5 minutes
  val Number nTimeFrom = now.millis - 300000;

  // Request the current powerconsumption between now - N minutes and now
  // Two communication options have been tested; WLAN (not used) and Direct
  Ethernet (used)
  try {
    // IP ETH0 = Direct connection between the volkszaehler Raspberry Pi and the
    Home Automation Raspberry Pi
```

```

    var String sResult =
sendHttpRequest("http://169.254.87.85/middleware.php/data/23844c00-8895-11e5-
a642-6d8ce36c9a44.json?from=" + nTimeFrom.toString + "&To=now")

    // IP WLAN0 = WLAN connection between the volkszaehler Raspberry Pi and the
Home Automation Raspberry Pi
    // var String sResult =
sendHttpRequest("http://192.168.0.39/middleware.php/data/23844c00-8895-11e5-
a642-6d8ce36c9a44.json?from=" + nTimeFrom.toString + "&To=now")

    // Parse the json result for the key data.average
    var String sPowerConsumption = transform("JSONPATH", "$.data.average", sResult)

    // Update the item
    nHA_Energy_PowerConsumption.postUpdate(sPowerConsumption)
    logInfo("POWERCONSUMPTION UPDATE", "Powerconsumption: " + sPowerConsumption)
}
catch(Throwable t) {
    logError("POWERCONSUMPTION UPDATE", "Error updating Power Consumption: " +
t.toString)
    // sendMail(...)
}
finally {
    // Runs always
}
end

/*
MQTT Publish Power Consumption
*/
rule "MQTT Publish Power Consumption"
when
    Item nHA_Energy_PowerConsumption changed
then
    // Publish the message to topic using the specified MQTT broker.
    publish("openhabprod", "homeautomation/energy/powerconsumption",
nHA_Energy_PowerConsumption.state.toString)
    // logInfo("MQTT", "*** Published homeautomation/energy/powerconsumption")
end
...

```

### Example HTTP Get Request JSON Result

The key used from the JSON string is data.average with value 208.173.

```

{
"version": "0.3",
"data":
{
    "tuples":
        [
            [1515485939605, 209.059, 1],
            [1515485941348, 206.54, 1],
            [1515485943071, 208.938, 1]
        ],
    "uuid": "23844c00-8895-11e5-a642-6d8ce36c9a44",
    "from": 1515485937883,
    "to": 1515485943071,
    "min": [1515485941348, 206.54044673488],
    "max": [1515485939605, 209.05923461769],

```

```
    "average":208.173,  
    "consumption":0.3,  
    "rows":4  
  }  
}
```

### Example Log Entry

```
2018-01-09 15:40:00.224 [INFO ] [model.script.POWERCONSUMPTION UPDATE] -  
Powerconsumption: 316.051
```

## Notes

If the Raspberry Pi volkszaehler reboots, the Raspberry Pi Home Automation must also reboot.

When openHAB starts, an error is logged for the Rule which sends HTTP request for the power consumption, to the volkszaehler Raspberry Pi.

This happens once and next time the power consumption is requested, the information is correct.

```
[ERROR] [ntime.internal.engine.ExecuteRuleJob] - Error during the execution of  
rule 'Energy Power Consumption Update': The name 'HA_Energy_PowerConsumption'  
cannot be resolved to an item or type; line 141, column 3, length 26
```

# Function Charts rrd4j

## Goal

The rrd4j persistent service is used to present time series data quickly by using the chart element type.

## Persistence Configuration

### Configuration Persistence File

```
/etc/openhab2/persistence/rrd4j.persist
```

### Configuration Persistence Content

The items are defined the configuration file `/etc/openhab2/items/homeautomation.items`. To avoid huge data collection, only the items for which charts are created are defined.

```
Strategies
{
  // for rrd charts, a cron strategy with every Minute is a must have.
  everyMinute : "0 * * * * ?"
  // get the data reduced for older values to keep database small
  everyHour : "0 0 * * * ?"
  everyDay : "0 0 0 * * ?"
  default = everyChange
}

// Information
// Use everyChange instead of everyUpdate
// Additional persist Items for groups
// gWeatherChart* : strategy = everyMinute, everyChange, restoreOnStartup
Items
{
  nHA_Weather_Home_Temperature : strategy = everyChange, everyMinute,
  restoreOnStartup
  nHA_Weather_Home_WindSpeedBft: strategy = everyChange, everyMinute, restoreOnStartup
  nHA_Weather_Home_Pressure : strategy = everyChange, everyMinute, restoreOnStartup
  nHA_LivingRoom_Temperature : strategy = everyChange, everyMinute, restoreOnStartup
  nHA_LivingRoom_Humidity : strategy = everyChange, everyMinute, restoreOnStartup
  nHA_Energy_PowerConsumption: strategy = everyChange, everyMinute, restoreOnStartup
  nHA_Info_HomepageCounter : strategy = everyChange, everyMinute, restoreOnStartup
  gHA_SystemInfoCharts* : strategy = everyChange, everyMinute, restoreOnStartup
}
```

### Chart Files Location

For the chart items, the chart files created are located in folder.

```
/var/lib/openhab2/persistence/rrd4j/
```

List the files (to confirm if the data is captured for the items defined in the configuration).

```
ls /var/lib/openhab2/persistence/rrd4j/*
/var/lib/openhab2/persistence/rrd4j/nHA_Energy_PowerConsumption.rrd
/var/lib/openhab2/persistence/rrd4j/nHA_LivingRoom_Humidity.rrd
```

... and more chart files depending configuration

## Items Configuration

This is an example for one item, nHA\_Energy\_PowerConsumption, on how charts are created.

```
Number nHA_Energy_PowerConsumption "Power Consumption [%.0f kWh]" <energy>
(gHA_Energy)

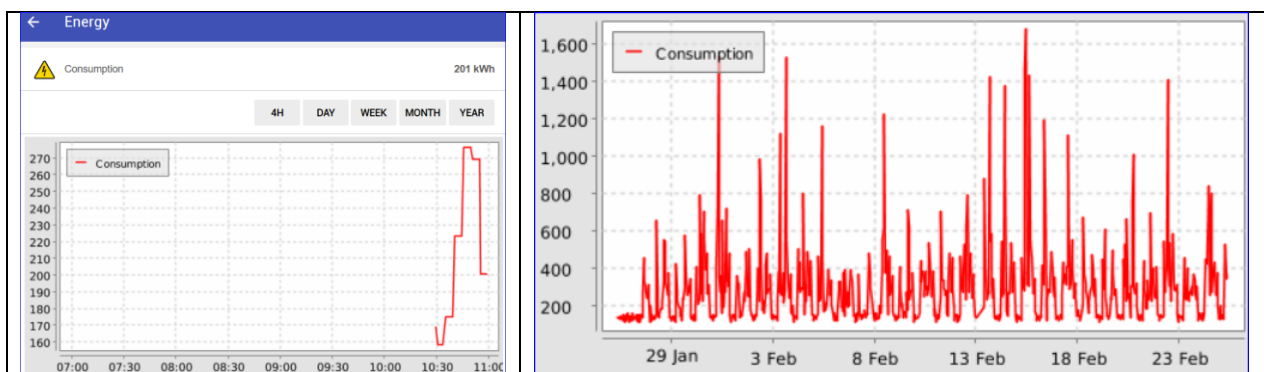
// ChartPeriod (Number) Set Mapping in sitemap
Number nHA_ChartPeriod_PowerConsumption "Period"
```

## Sitemap Configuration

```
Text label="Energy" icon="energy" {
  Text item=nHA_Energy_PowerConsumption
  Switch item=nHA_ChartPeriod_PowerConsumption
  mappings=[0="4H",1="Day",2="Week",3="Month",4="Year"]
  Chart item=nHA_Energy_PowerConsumption period=4h refresh=60000 legend=true
  visibility=[nHA_ChartPeriod_PowerConsumption==0,nHA_ChartPeriod_PowerConsumption==
  "NULL"]
  Chart item=nHA_Energy_PowerConsumption period=D refresh=60000 legend=true
  visibility=[nHA_ChartPeriod_PowerConsumption==1]
  Chart item=nHA_Energy_PowerConsumption period=W refresh=60000 legend=true
  visibility=[nHA_ChartPeriod_PowerConsumption==2]
  Chart item=nHA_Energy_PowerConsumption period=M refresh=60000 legend=true
  visibility=[nHA_ChartPeriod_PowerConsumption==3]
  Chart item=nHA_Energy_PowerConsumption period=Y refresh=60000 legend=true
  visibility=[nHA_ChartPeriod_PowerConsumption==4]
}
```

## Screen Shots

The charts show a fresh start of data capturing and consumption for a whole month.





## Delete Chart Data

### Clean Chart Files

Recommended to clean the chart files (.rrd extension) if changes made to the rrd4j configuration

To delete the files, stop openHAB, remove the files and start openHAB again.

### Example

```
sudo systemctl stop openhab2.service

ls /var/lib/openhab2/persistence/rrd4j/*
/var/lib/openhab2/persistence/rrd4j/nLivingRoom_Humidity.rrd
/var/lib/openhab2/persistence/rrd4j/nLivingRoom_Temperature.rrd
...

rm -r /var/lib/openhab2/persistence/rrd4j/*
rm: remove write-protected regular file
'/var/lib/openhab2/persistence/rrd4j/Readme.txt'? n

sudo systemctl start openhab2.service
```

## Selective Charts

Selective charts are created opting for a period Hour, Day, Week, Month, Year.

### Item Configuration

In the homeautomation.items a number ChartPeriod is defined which is used by a switch in the sitemap to select the period.

*This is just an example – the actual configuration uses different naming & mappings.*

```
// ChartPeriod (Number)
// Mapping in sitemap: mappings=[0="Hour", 1="Day", 2="Week", 3="Month", 4="Year"]
Number nChartPeriod "Chart Period"
```

### Sitemap Configuration

Charts are grouped and for each of the groups a switch is used to select the period. In case more than one chart, the legend property is set.

```
Text label="Living Room Charts" icon="line-incline" {
  Switch item=nChartPeriod label="Period"
  mappings=[0="Hour",1="Day",2="Week",3="Month", 4="Year"]
  Chart item=nLivingRoom_Temperature period=h refresh=60000 legend=true
  visibility=[nChartPeriod==0,nChartPeriod=="NULL"]
  Chart item=nLivingRoom_Temperature period=D refresh=60000 legend=true
  visibility=[nChartPeriod==1]
  Chart item=nLivingRoom_Temperature period=W refresh=60000 legend=true
  visibility=[nChartPeriod==2]
  Chart item=nLivingRoom_Temperature period=M refresh=60000 legend=true
  visibility=[nChartPeriod==3]
  Chart item=nLivingRoom_Temperature period=Y refresh=60000 legend=true
  visibility=[nChartPeriod==4]
}
```

## Period Values

H = 1 hour, 4h= 4 hours, 12h= 12 hours

D =1 day, 3D =3 days,

W = 1 week, 2W = 2 weeks,

M = 1 month, 2M = 2 month, 4M = 4 month

Y = 1 year

## Refresh Period

### IMPORTANT:

Do not set the refresh period (ms) to

o low; else the PNG file can be created resulting in an I/O error.

A value of for example 15ms (refresh=15) will cause errors (I/O, Broken Pipe). Good value is 60000ms (refresh=60000).

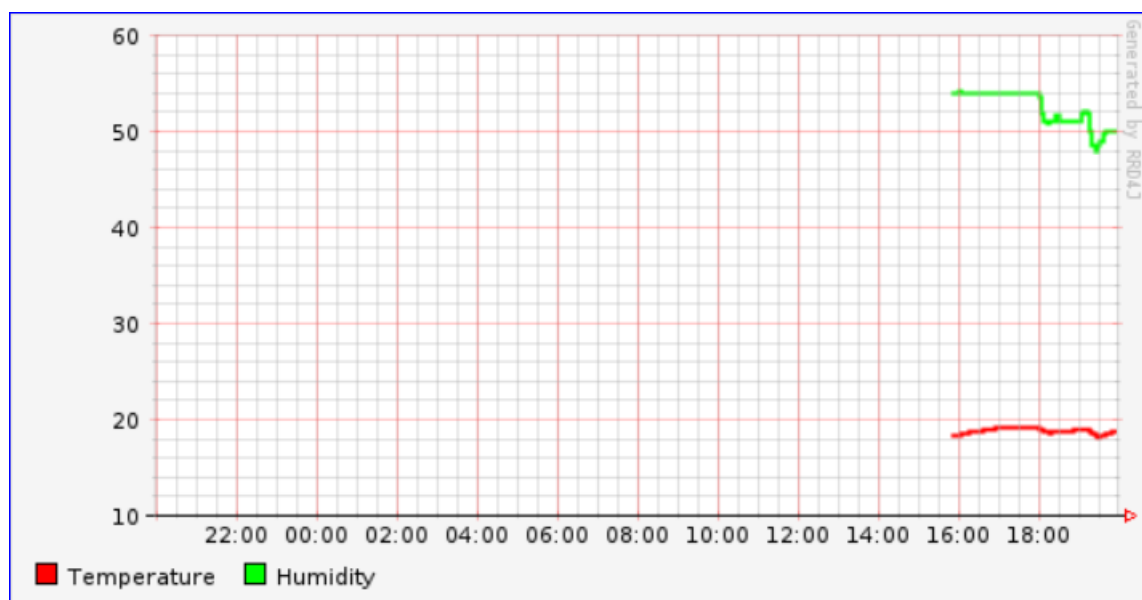
## Display Chart in Web Browser

It is possible to display a chart in a web browser by using an URL with openHAB server and parameters rrdchart.png with the items and period to display.

### Example

This is just an example – the actual configuration uses different naming & mappings.

[http://192.168.N.NN:8080/rrdchart.png?items=nLivingRoom\\_Temperature,nLivingRoom\\_Humidity&period=H](http://192.168.N.NN:8080/rrdchart.png?items=nLivingRoom_Temperature,nLivingRoom_Humidity&period=H)



# Function MQTT

## Goal

To publish or subscribe to MQTT messages via a Items or Rules.

Example: Publish the power meter value for Item HA\_Energy\_PowerConsumption.

## Overview Topics Published

Function	Topic	Payload
Get Front Door Alarm state	homeautomation/security/frontdoor/state	ALARM_TAMPER NORMAL
Reset alarm state front door contact (ON = state reset)	homeautomation/security/frontdoor/reset/set	ON OFF
Set front door contact notification	homeautomation/security/frontdoor/notify/set	ON OFF
Get Power Consumption	homeautomation/energy/powerconsumption	Number
Get Brightness Hue Light 1	homeautomation/makelab/huelight1	0 – 100
Set Brightness Hue Light 1	homeautomation/makelab/huelight1/set	0 – 100

### Notes

The MQTT topics use the same syntax as the Item definition.

Instead of the prefix HA\_, the prefix homeautomation is used for better readability.

**Example** Item HA\_Energy\_PowerConsumption =

Topic homeautomation/energy/powerconsumption

Important:

Topics are defined in lowercase.

Topics are tested in a terminal with mosquitto commands.

```

mosquitto_pub -t homeautomation/security/frontdoor/notify/set -m ON

mosquitto_pub -t homeautomation/security/frontdoor/notify/set -m OFF

mosquitto_pub -t homeautomation/security/frontdoor/reset/set -m ON

mosquitto_pub -t homeautomation/makelab/huelight1/set -m 10

mosquitto_sub -v -t homeautomation/makelab/huelight1
homeautomation/ makelab /huelight1 19

mosquitto_sub -v -t homeautomation/powerconsumption
homeautomation/powerconsumption 123.56

```

# Message Energy Power Consumption

Publish the message Energy Power Consumption value.

## Topic

Definition: **homeautomation/energy/powerconsumption**

Payload: Number, i.e. 273.12

## Rule

```
rule "MQTT Publish Power Consumption"
when
  Item nHA_Energy_PowerConsumption changed
then
  publish("openhabprod","homeautomation/energy/powerconsumption",
nHA_Energy_PowerConsumption.state.toString)
  logInfo("MQTT", "*** Published homeautomation/energy/powerconsumption")
end
```

## Test

In a terminal subscribe to the topic.

```
mosquitto_sub -v -t homeautomation/powerconsumption
```

## Test Result

```
homeautomation/energy/powerconsumption 480.354
```

## Test Log

```
2018-01-10 16:40:00.532 [INFO ] [model.script.POWERCONSUMPTION UPDATE] -
Powerconsumption: 480.354
2018-01-10 16:40:00.566 [INFO ] [.eclipse.smarthome.model.script.MQTT] - ***
Published homeautomation/energy/powerconsumption
```

## Message Security Front Door State

If the state of the Security Front Door changes, then publish a message, via a Rule, containing the alarm state of the Front Door.

### Topic

Definition: **homeautomation/security/frontdoor/state**

Payload: ALARM\_TAMPER | NORMAL

### Rule

```
/*
  Security FrontDoor
  If the status changes:
  * publish a MQTT message with topic homeautomation/security/frontdoor/state and
  payload state
  * is ALARM_TAMPER and notify flag is ON, send email using the Mail Action
*/
rule "Security FrontDoor Contact"
when
  Item sHA_Security_FrontDoor_Status changed
then
  publish("openhabprod","homeautomation/security/frontdoor/state",
sHA_Security_FrontDoor_Status.state.toString)
  logInfo("Security", "*** FrontDoor State changed to " +
sHA_Security_FrontDoor_Status.state.toString)
  dtHA_Security_FrontDoor_StatusTime.postUpdate( new DateTimeType() )
  if (sHA_Security_FrontDoor_Status.state.toString == "ALARM_TAMPER") {
    if (aHA_Security_FrontDoor_Notify.state == ON) {
      val String logTime = String.format( "%1$td-%1$tm-%1$tY %1$tH:%1$tM:%1$tS",
new java.util.Date )
      sendMail("TOADDRESS", "Security FrontDoor - Open", "Door opened at "+ logTime)
    }
  }
}
end
```

### Test

Subscribe to the topic holding the state of the frontdoor security.

When the rule notifies a changes in state, it publishes a message which content is shown.

```
mosquitto_sub -v -t homeautomation/security/frontdoor/state
```

### Test Result

```
homeautomation/security/frontdoor/state NORMAL
homeautomation/security/frontdoor/state ALARM_TAMPER
```

### Test Log

```
2018-01-10 19:06:20.064 [vent.ItemStateChangedEvent] -
HA_Security_FrontDoor_Status changed from ALARM_TAMPER to NORMAL
2018-01-10 19:06:20.075 [INFO ] [pse.smarthome.model.script.Frontdoor] - ***
FrontDoor State changed to NORMAL
```



# Message Security Front Door Reset

Subscribe to message resetting the alarm state of the Front Door via an Item. The message can be published via a rule or other application.

## Topic

Definition: **homeautomation/security/frontdoor/reset/set**

Payload: ON | OFF

## Item

```
Switch aHA_Security_FrontDoor_Reset_Remote "Reset Status Remote"
{
  mqtt="<[openhabprod:homeautomation/security/frontdoor/reset/set:state:default:.*]"
}
```

## Notes

Messages published to the topic "homeautomation/security/frontdoor/reset/set" set the state to the message. The Switch Item converts 1 to ON and 0 to OFF – no transform needed.

## Rule

```
/*
  Reset the status of the security frontdoor alarm switch using MQTT message.
  The MQTT message is defined as an Item.
*/
rule "Security FrontDoor Reset Remote"
when
  Item aHA_Security_FrontDoor_Reset_Remote changed
then
  aHA_Security_FrontDoor_Reset.state = aHA_Security_FrontDoor_Reset_Remote.state
  logInfo("FrontDoor", "*** FrontDoor Remote Reset to " +
    HA_Security_FrontDoor_Reset_Remote.state.toString)
end
```

## Test

```
mosquitto_pub -t homeautomation/security/frontdoor/reset/set -m ON
```

## Test Result

No information as message is published.

## Test Log

```
2018-01-10 19:06:20.003 [ome.event.ItemCommandEvent] - Item
'HA_Security_FrontDoor_Reset' received command ON
2018-01-10 19:06:20.018 [vent.ItemStateChangedEvent] - HA_Security_FrontDoor_Reset
changed from OFF to ON
2018-01-10 19:06:20.042 [vent.ItemStateChangedEvent] - HA_Security_FrontDoor_Reset
changed from ON to OFF
2018-01-10 19:06:20.064 [vent.ItemStateChangedEvent] -
HA_Security_FrontDoor_Status changed from ALARM_TAMPER to NORMAL
2018-01-10 19:06:20.075 [INFO ] [pse.smarthome.model.script.Frontdoor] - ***
FrontDoor State changed to NORMAL
```





## Message Security Front Door Notify

Set the Security Front Door Notify switch to ON or OFF by subscribing to the topic in an Item. The message can be published via a rule or other application.

### Topic

Definition: **homeautomation/security/frontdoor/notify/set**

Payload: ON | OFF

### Item

```
Switch aHA_Security_FrontDoor_Notify_Remote "Notify Remote"
{
  mqtt="<[openhabprod:homeautomation/security/frontdoor/notify/set:state:default:.*]
  "}
```

### Rule

```
/*
  Set the switch of the security frontdoor notify using MQTT message.
  The MQTT message is defined as an Item.
*/
rule "Security FrontDoor Notify Remote"
when
  Item aHA_Security_FrontDoor_Notify_Remote changed
then
  aHA_Security_FrontDoor_Notify.state = aHA_Security_FrontDoor_Notify_Remote.state
  logInfo("Security","*** FrontDoor Notify set to "+
aHA_Security_FrontDoor_Notify_Remote.state.toString)
end
```

### Test

```
mosquitto_pub -t homeautomation/security/frontdoor/notify/set -m ON
mosquitto_pub -t homeautomation/security/frontdoor/notify/set -m OFF
```

### Test Result

No information as message is published. See log.

### Test Log

```
[INFO ] [ipse.smarthome.model.script.Security] - *** FrontDoor Notify set to ON
[vent.ItemStateChangedEvent] - HA_Security_FrontDoor_Notify_Remote changed from ON
to OFF
[INFO ] [ipse.smarthome.model.script.Security] - *** FrontDoor Notify set to OFF
```

## Message HueLight1 Brightness

Set or get (via an Item) the brightness of the HueLight1.  
The message can be published via a rule or other application.

### Topic

#### Set the brightness (publish)

Definition: **homeautomation/makelab /huelight1/set**

Payload: 0 – 100

#### Get the brightness (subscribe)

Definition: **homeautomation/makelab/huelight1**

Payload: 0 – 100

### Item

```
Number nHA_MakeLab_HueLight1
{ mqtt="<[openhabprod:homeautomation/makelab/huelight1/set:state:default:.*]" }
```

### Rules

```
rule "HueLight1 Brightness Set"
when
  Item nHA_MakeLab_HueLight1 changed
then
  logInfo("HUELIGHT1", "*** Brightness set to " +
nHA_LivingRoom_HueLight1.state.toString)
  aHA_LivingRoom_HueLight1.sendCommand(nHA_LivingRoom_HueLight1.state.toString)
end

rule "HueLight1 Brightness Changed"
when
  Item aHA_MakeLab_HueLight1 changed
then
  publish("openhabprod","homeautomation/makelab/huelight1",
aHA_LivingRoom_HueLight1.state.toString)
end
```

### Test

```
mosquitto_pub -t homeautomation/makelab/huelight1/set -m 10

mosquitto_sub -v -t homeautomation/makelab/huelight1
homeautomation/makelab/huelight1 19
```

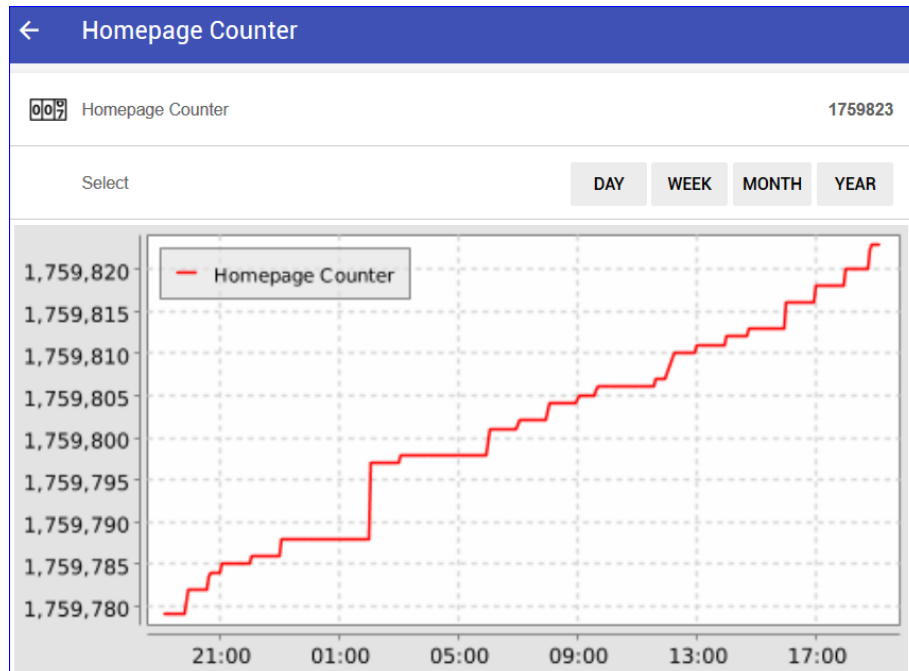
### Test Log

```
[ome.event.ItemCommandEvent] - Item 'aHA_MakeLab_HueLight1' received command 10
[vent.ItemStateChangedEvent] - aHA_MakeLab_HueLight1 changed from 100 to 10
[vent.ItemStateChangedEvent] - aHA_MakeLab_HueLight1s changed from 100 to 10
[ome.event.ItemCommandEvent] - Item 'aHA_MakeLab_HueLight1s' received command 19
[vent.ItemStateChangedEvent] - aHA_MakeLab_HueLight1s changed from 10 to 19
[vent.ItemStateChangedEvent] - aHA_MakeLab_HueLight1 changed from 10 to 19
```

# Function Homepage Counter

## Goal

To show a homepage counter value and rrd4j charts period day, week, month and year.



## Concept

The homepage count is captured in a text file located on the server. The text file contains only the count as number. Via a HTTP request, the content is read and assigned to an item. The HTTP request is executed every hour via cron job.

## Items Configuration

```
Number nHA_Info_HomepageCounter "Homepage Counter [%d]" <homepagecounter>
(gHA_Homepage)

// ChartPeriod (Number)
Number nHA_ChartPeriod_HomePageCounter "Period"
```

## Sitemap Configuration

```
Text label="Homepage Counter" icon="homepagecounter" {
  Text item=nHA_Info_HomepageCounter
  Switch item=nHA_ChartPeriod_HomePageCounter
  mappings=[0="Day",1="Week",2="Month",3="Year"]
  Chart item=nHA_Info_HomepageCounter period=D refresh=60000 legend=true
  visibility=[nHA_ChartPeriod_HomePageCounter==0,
  nHA_ChartPeriod_HomePageCounter=="NULL"]
  Chart item=nHA_Info_HomepageCounter period=W refresh=60000 legend=true
  visibility=[nHA_ChartPeriod_HomePageCounter==1]
  Chart item=nHA_Info_HomepageCounter period=M refresh=60000 legend=true
  visibility=[nHA_ChartPeriod_HomePageCounter==2]
```

```
Chart item=nHA_Info_HomepageCounter period=Y refresh=60000 legend=true
visibility=[aHA_ChartPeriod_HomePageCounter==3]
}
```

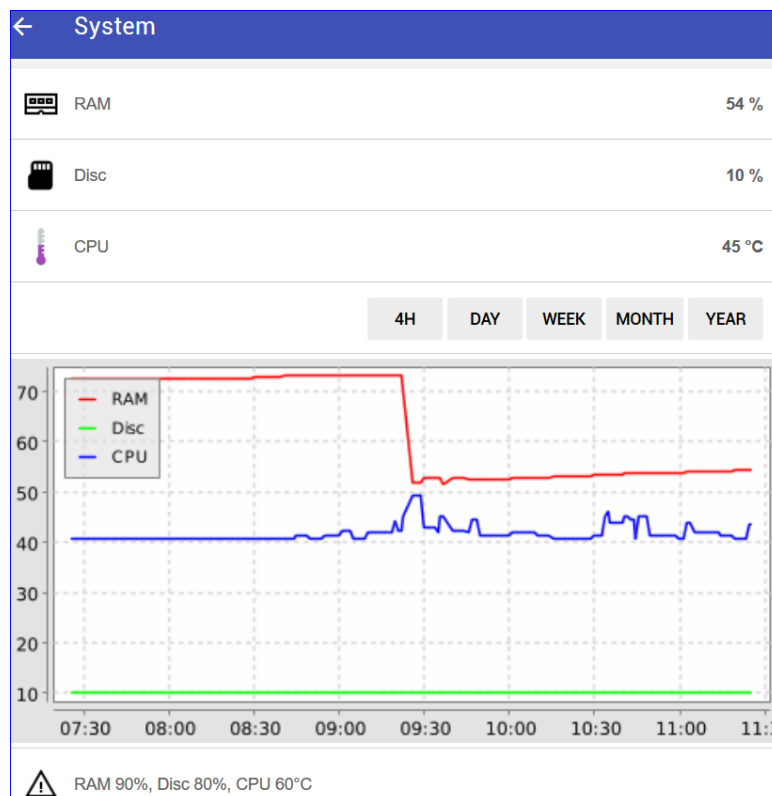
## Rules Configuration

```
rule "Homepage Counter Hourly Update"
when
  System started or
  Time cron "0 0 0/1 ? * * *"
then
  // Get Homepage Counter content
  var String result = sendHttpRequest("https://ipaddress/pagecounterfile")
  var Number cnt
  logInfo("HOMEPAGECOUNTER UPDATE", "Pagecounter: " + result)
  cnt = Integer::parseInt(result)
  nHA_Info_HomepageCounter.postUpdate(cnt)
end
```

# Function Raspberry Pi System Information

## Goal

To measure & show specific Raspberry Pi system information and trigger alarm if a value is above a certain threshold.



## Concept

The Raspberry Pi System information is measured by executing Bash Scripts located in folder

`/etc/openhab2/scripts.`

Each measurement has its own Bash Script.

The Scripts are defined in the **Rule "RPi System Information Update"** and executed via the method **executeCommandLine** every 5 minutes with timeout of 5 seconds.

The Rule checks a measured value against its threshold and sends a notification (email) if value is above the threshold.

In addition, scripts are used to reboot or shutdown the Raspberry Pi in case of thresholds reached for RAM used (reboot) & Disc Space used (shutdown).  
For the CPU temperature, a notification is send.

## Measurement & Action

Item	Bash Script Content	Bash Script Name	Output
CPU Temperature	<code>#!/bin/bash cat /sys/class/thermal/thermal_zone0/temp</code>	<code>rpicutemperature.sh</code>	NNNN Example: 40726 Divide by 1000 for °C
Disc Space used	<code>#!/bin/bash df -H   grep 'root'   awk '{ print \$5 }'   tr % " "   xargs</code>	<code>rpiscspaceused.sh</code>	NN Example: 10
RAM used	<code>#!/bin/bash free -o -h   head -n2   tail -n1   awk '{print \$3}'   tr -d [=M=]</code>	<code>rpiramused.sh</code>	NNN Example: 606
RAM free	<code>#!/bin/bash free -o -h   head -n2   tail -n1   awk '{print \$4}'   tr -d [=M=]</code>	<code>rpiramfree.sh</code>	NNN Example: 271
RAM total	<code>#!/bin/bash free -o -h   head -n2   tail -n1   awk '{print \$2}'   tr -d [=M=]</code>	<code>rpiramtotal.sh</code>	NNN Example: 970
Reboot	<code>#!/bin/bash sudo reboot now echo PASSWORD</code>	<code>rpireboot.sh</code>	
Shutdown	<code>#!/bin/bash sudo shutdown -h 0 echo PASSWORD</code>	<code>rpishutdown.sh</code>	

### Notes

- Create the scripts with the nano editor (to avoid any format errors like \r found)
- Make the script executable and run for test.
- Example Rule:  
val ramused = executeCommandLine("/etc/openhab2/scripts/rpiramused.sh", 5000)

### Example Script `rpiscspaceused.sh`

```
#Set script folder
cd /etc/openhab2/scripts

#Edit the script
nano rpiscspaceused.sh
with content:
df -H | grep 'root' | awk '{ print $5 }' | tr % " " | xargs

#Make the script executable
sudo chmod a+x rpiscspaceused.sh

#Run the script for test
./rpiscspaceused.sh

#Script output example
10%
```

## Items Configuration

Group	gHA_SystemInfo	"System"	<systeminfo>
Group	gHA_SystemInfoCharts	"System Charts"	
Number	nHA_Info_RPiRAMUsed	"RAM [%.0f %]"	<ram>
(gHA_SystemInfo,gHA_SystemInfoCharts)			
Number	nHA_Info_RPiDiscSpaceUsed	"Disc [%.0f %]"	<sdcard>
(gHA_SystemInfo,gHA_SystemInfoCharts)			
Number	nHA_Info_RPiCPUTemperature	"CPU [%.0f °C]"	<temperature>
(gHA_SystemInfo,gHA_SystemInfoCharts)			
String	sHA_Info_RpiThresholds	"RAM 90%, Disc 80%, CPU 60°C"	<warning>

## Sitemap Configuration

```
Text    item=sHA_System_Message

Text label="System" icon="systeminfo" {
  Text item=nHA_Info_RPiRAMUsed
  Text item=nHA_Info_RPiDiscSpaceUsed
  Text item=nHA_Info_RPiCPUTemperature
  Switch item=nHA_ChartPeriod_SystemInfo
  mappings=[0="4H",1="Day",2="Week",3="Month",4="Year"]
  Chart item=gHA_SystemInfo period=4h refresh=60000 legend=true
  visibility=[nHA_ChartPeriod_SystemInfo==0,nHA_ChartPeriod_SystemInfo=="NULL"]
  Chart item=gHA_SystemInfo period=D refresh=60000 legend=true
  visibility=[nHA_ChartPeriod_SystemInfo==1]
  Chart item=gHA_SystemInfo period=W refresh=60000 legend=true
  visibility=[nHA_ChartPeriod_SystemInfo==2]
  Chart item=gHA_SystemInfo period=M refresh=60000 legend=true
  visibility=[nHA_ChartPeriod_SystemInfo==3]
  Chart item=gHA_SystemInfo period=Y refresh=60000 legend=true
  visibility=[nHA_ChartPeriod_SystemInfo==4]
  Text label="RAM,Disc 80%, CPU 45°C" icon="warning"
}
```

## Rules Configuration

```
// Thresholds
// Max CPU temp is 85°C, above this the Pi will thermal throttle to bring the
// temperature back down; higher risks damaging the SoC.
var    Number nRPiCPUTemperatureMax = 60
var    Number nRPiRAMUsedPctMax = 90
var    Number nRPiDiscSpaceUsedPctMax = 80

var String    sEmailTo = "EMAILADDRESS"

/*
  System Information running every 5 minutes with timeout 5 secs
  If a measure is above threshold, an action is triggered:
  CPU Temperature: send notification
  RAM Used: send notification & reboot
  Disc Space Used: send notification & shutdown
  The notification is updating the Basic UI status message and sending an email.
*/
rule "RPi System Information Update"
when
  System started or
  Time cron "0 */5 * ? * *"

```

```

then
// Reset the system message to OK
SHA_System_Message.postUpdate("OK")

// CPU Temperature - check if above threshold - if YES then notify
val vRPiCPUTemperature =
executeCommandLine("/etc/openhab2/scripts/rpicputemperature.sh", 5000)
// CPU Temperature result must be divided by 1000 to get in C
var Number nRPiCPUTemperature = Float::parseFloat(vRPiCPUTemperature) / 1000
nHA_Info_RPiCPUTemperature.postUpdate(nRPiCPUTemperature)
if (nRPiCPUTemperature > nRPiCPUTemperatureMax) {
    SHA_System_Message.postUpdate("SHUTDOWN HIGH CPU Temperature " +
String::format( "%.0f", nRPiCPUTemperature) + "°C")
    sendMail(sEmailTo, "SHUTDOWN - Home Automation HIGH CPU Temperature", "CPU
Temperature " + String::format( "%.0f", nRPiCPUTemperature) + " above threshold "
+ nRPiCPUTemperatureMax)
    logInfo("SYSTEMINFO", "*** SHUTDOWN HIGH CPU Temperature " + String::format(
"%0f", nRPiCPUTemperature) + " (Max:" + nRPiCPUTemperatureMax + ")")
    return
} else {
    logInfo("SYSTEMINFO", "*** CPU Temperature: " + String::format( "%.0f",
nRPiCPUTemperature) + " (Max:" + nRPiCPUTemperatureMax + ")")
}

// RAM total
val vRPiRAMTotal = executeCommandLine("/etc/openhab2/scripts/rpiramtotal.sh",
5000)
// logInfo("SYSTEMINFO", "*** RAM Total: " + vRPiRAMTotal)

// RAM used
val vRPiRAMUsed = executeCommandLine("/etc/openhab2/scripts/rpiramused.sh", 5000)
// logInfo("SYSTEMINFO", "*** RAM Used: " + vRPiRAMUsed)

// RAM used in pct check if above threshold - if YES then notify & reboot to
release memory
var Number nRPiRAMUsedPct = (Float::parseFloat(vRPiRAMUsed) /
Float::parseFloat(vRPiRAMTotal)) * 100;
nHA_Info_RPiRAMUsed.postUpdate(nRPiRAMUsedPct)
if (nRPiRAMUsedPct > nRPiRAMUsedPctMax) {
    SHA_System_Message.postUpdate("REBOOTING HIGH RAM " + String::format( "%.0f",
nRPiRAMUsedPct) + "%.")
    sendMail(sEmailTo, "REBOOT - Home Automation HIGH RAM", "Memory Used " +
String::format( "%.0f", nRPiRAMUsedPct) + " above threshold " + nRPiRAMUsedPctMax
+ ". System reboot initiated.")
    logInfo("SYSTEMINFO", "*** REBOOTING HIGH RAM " + String::format( "%.0f",
nRPiRAMUsedPct) + " (Max:" + nRPiRAMUsedPctMax + ")")
    executeCommandLine("/etc/openhab2/scripts/rpireboot.sh", 5000)
    return
} else {
    logInfo("SYSTEMINFO", "*** RAM Used %: " + String::format( "%.0f",
nRPiRAMUsedPct) + " (Max:" + nRPiRAMUsedPctMax + ")")
}

// Disc Space Used - check if used above threshold - if YES then notify &
shutdown
val vRPiDiscSpaceUsedPct =
executeCommandLine("/etc/openhab2/scripts/rpidiscspaceused.sh", 5000)
var Number nRPiDiscSpaceUsedPct = Float::parseFloat(vRPiDiscSpaceUsedPct)
nHA_Info_RPiDiscSpaceUsed.postUpdate(nRPiDiscSpaceUsedPct)
if (nRPiDiscSpaceUsedPct > nRPiDiscSpaceUsedPctMax) {

```



```
    SHA_System_Message.postUpdate("SHUTDOWN DISK FULL " + String::format( "%.0f",
nRPiDiscSpaceUsedPct) + "%.")
    sendMail(sEmailTo, "SHUTDOWN - Home Automation DISC FULL", "Disc Space Used " +
String::format( "%.0f", nRPiDiscSpaceUsedPct) + " above threshold " +
nRPiDiscSpaceUsedPctMax + ". Shutting down.")
    logInfo("SYSTEMINFO", "*** SHUTDOWN DISK FULL " + String::format( "%.0f",
nRPiDiscSpaceUsedPct) + " (Max:" + nRPiDiscSpaceUsedPctMax + ")")
    executeCommandLine("/etc/openhab2/scripts/rpishutdown.sh", 5000)
    return
} else {
    logInfo("SYSTEMINFO", "*** Disc Used %: " + String::format( "%.0f",
nRPiDiscSpaceUsedPct) + " (Max:" + nRPiDiscSpaceUsedPctMax + ")")
}
}
end
```

## Bash Script for Tests

This script can be used for tests. The output of each check is written to a file.

```
#!/bin/bash
#Note: sudo chmod a+x rpiinfo.sh. Then run ./rpiinfo.sh

#Set folder to store the outputs
cd /etc/openhab2/scripts

#Get CPU Temperature - output NNNNN - divide by 1000 to get the temperature
cat /sys/class/thermal/thermal_zone0/temp > rpicputemp.inf

#Get Disc space used - output NN
df -H | grep 'root' | awk '{ print $5 }' | tr % " " | xargs > rpidiscspaceused.inf

#Get RAM used - output NNN
free -o -h | head -n2 | tail -n1 | awk '{print $3}' | tr -d [=M=] > rpimemused.inf

#Get RAM free - output NNN
free -o -h | head -n2 | tail -n1 | awk '{print $4}' | tr -d [=M=] > rpimemfree.inf

#Get RAM total - output NNN
free -o -h | head -n2 | tail -n1 | awk '{print $2}' | tr -d [=M=] >
rpimemtotal.inf
```

# Function Hue

## Goal

To control devices connected to the Philips Hue Bridge.

Each of the Hue devices is defined as a Thing via the Paper-UI.

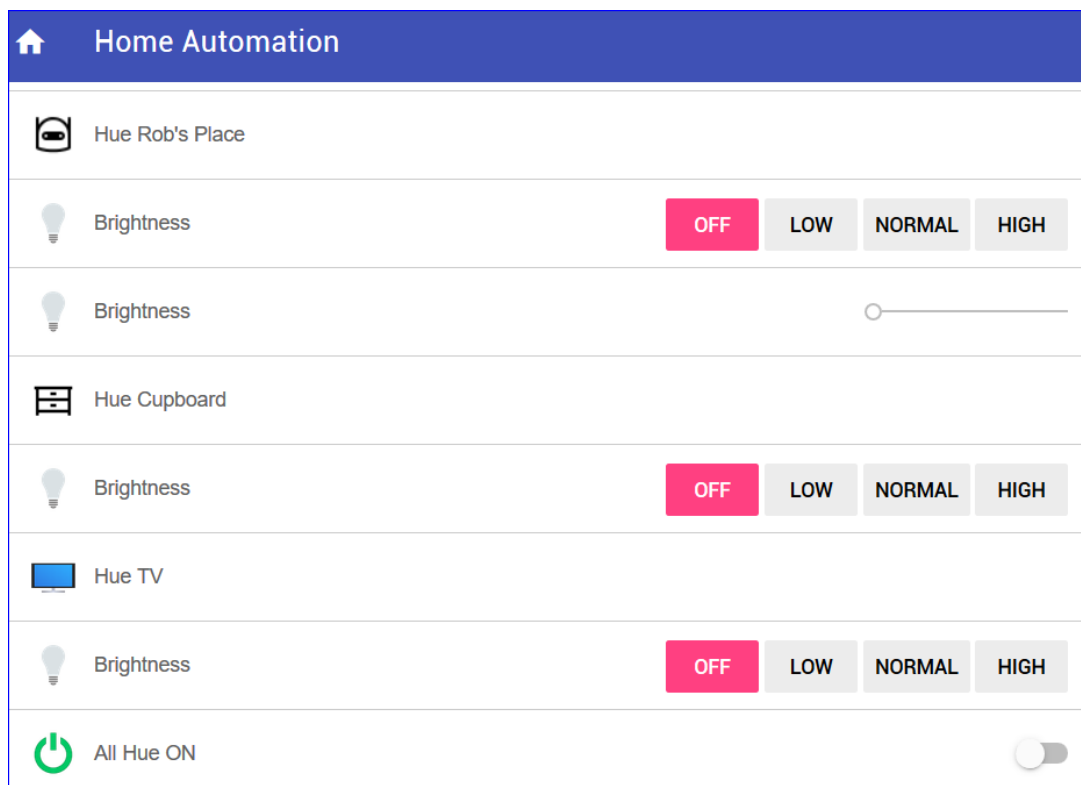
These Things are controlled by using the dedicated channel.

The channel syntax uses: Hue:thingtype:bridgeid:deviceid:function

The list of Hue Things defined, in order of their device id (1,2,3,4...):

Hue Thing	Item Channel(s)	Location
Hue white lamp 1 Dimmable Light	hue:0100:00178863da1d:1:brightness	Make Lab
Hue ambiance candle 1 Color Temperature Light A dimmable light with tunable color temperature.	hue:0220:00178863da1d:2:brightness Dimmer hue:0220:00178863da1d:2:color_temperature Dimmer	Living Room
Hue white lamp 2 Dimmable Light	hue:0100:00178863da1d:3:brightness	Living Room
Hue white lamp 3 Dimmable Light	hue:0100:00178863da1d:4:brightness	Living Room
<ADD MORE>		

## Example Sitemap



## Items Configuration

This is an example of a Hue Item.

Not the full Items configuration is shown here as changing, but also getting too big.

String	sHA_MakeLab_HueLight1	"Hue Make Lab"	<rob>
Dimmer	aHA_MakeLab_HueLight1bsw	"Brightness"	<light>
(gHA_MakeLab)	{ channel="hue:0100:00178863da1d:1:brightness" }		
Dimmer	aHA_MakeLab_HueLight1bsl	"Brightness"	<light>
(gHA_MakeLab)	{ channel="hue:0100:00178863da1d:1:brightness" }		
Number	nHA_MakeLab_HueLight1	{	
mqtt="<[openhabprod:homeautomation/makelab/huelight1/set:state:default:.*]" }			

### Define the Group Heading

String	sHA_MakeLab_HueLight1	"Hue Make Lab"	<rob>
--------	-----------------------	----------------	-------

String = String Item

sHA = String for Sitemap HA (=Home Automation)

MakeLab = Location of the Hue light

HueLight1 = Hue light with ID 1

"Hue Make Lab" = Label

<rob> = Icon

### Define the Brightness Switch

Dimmer	aHA_MakeLab_HueLight1bsw	"Brightness"	<light>
(gHA_MakeLab)	{ channel="hue:0100:00178863da1d:1:brightness" }		

Dimmer = Dimmer Item

aHA = Actuator for Sitemap HA (=Home Automation)

MakeLab = Location of the Hue light

HueLight1bsw = Hue light with ID 1 set the brightness (b) via switch (sw)

"Brightness" = Label

<light> = Icon

gHA\_MakeLab = Group for Sitemap HA (=Home Automation) location MakeLab

{ channel="hue:0100:00178863da1d:1:brightness" } = Channel used to set the brightness

### Define the Brightness Slider

Dimmer	aHA_MakeLab_HueLight1bsl	"Brightness"	<light>
(gHA_MakeLab)	{ channel="hue:0100:00178863da1d:1:brightness" }		

Same as the Switch with one change:

HueLight1bsl = Hue light with ID 1 set the brightness (b) via slider (sl)

## Define the MQTT Subscription

```
Number          nHA_MakeLab_HueLight1      {
mqtt="<[openhabprod:homeautomation/makelab/huelight1/set:state:default:.*]" }
```

### Number

Number Item to hold the brightness level 0 – 100 as set by the MQTT topic payload

### nHA

Number for Sitemap HA (=Home Automation)

### MakeLab

Location of the Hue light

### HueLight1

Hue light with ID 1

```
mqtt="<[openhabprod:homeautomation/makelab/huelight1/set:state:default:.*]"
```

<

Incoming message (subscribing to a topic)

### openhabprod

MQTT Broker as defined in the MQTT configuration in /etc/openhab2/services/mqtt.cfg

### homeautomation/makelab/huelight1/set

MQTT topic to listen to

### state:default:\*

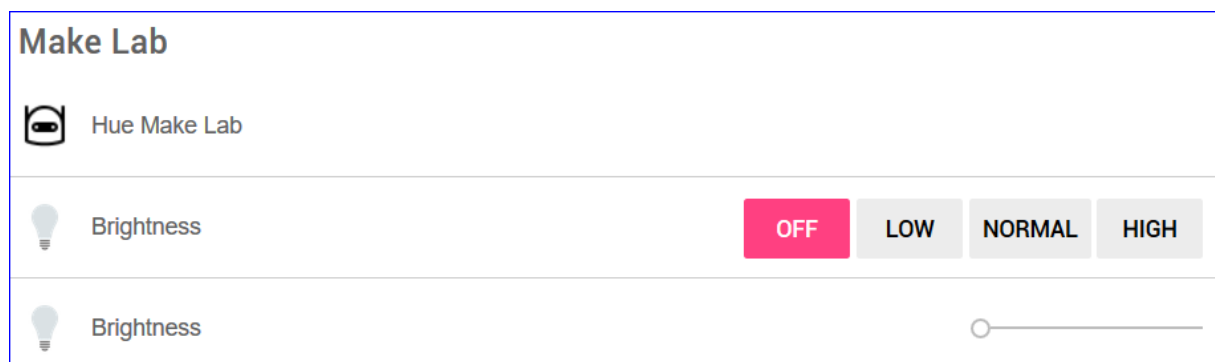
Change the state of the Hue light to the value given 0 – 100

A message will only be published to MQTT when the Item receives a command.  
(See under Rules Configuration below).

## Sitemap Configuration

```
Frame label="Make Lab" {
  Text      item=SHA_MakeLab_HueLight1
  Switch    item=aHA_MakeLab_HueLight1bsw mappings=[0="Off",25="Low",60="Normal",
100="High"]
  Slider    item=aHA_MakeLab_HueLight1bsl
}
```

### Example Hue Light 1 “Hue Make Lab” turned off



# Rules Configuration

## MQTT

The brightness of Hue Light1 “Hue Make Lab” can be set and read via MQTT.

### Subscribe

To set the brightness publish

- **topic** “homeautomation/makelab/huelight1/set” with
- **payload** between 0 – 100.

Publishing of the topic can be done via an external application. A nice example is changing the brightness via an ESP8266 controller with potentiometer (TODO DESCRIBE).

The Item will change accordingly as defined in the previous Items Configuration (listening to the changes of the topic):

```
Number          nHA_MakeLab_HueLight1      {
mqttt="<[openhabprod:homeautomation/makelab/huelight1/set:state:default:.*]" }
```

If the Number Item *nHA\_MakeLab\_HueLight1* changes, following rule is triggered which sets the state of the Switch Item *aHA\_MakeLab\_HueLight1bsl* (the slider).

```
rule "HueLight1 Brightness Set"
when
  Item nHA_MakeLab_HueLight1 changed
then
  logInfo("HUELIGHT1", "*** Brightness set to " +
nHA_MakeLab_HueLight1.state.toString)
  aHA_MakeLab_HueLight1bsl.sendCommand(nHA_MakeLab_HueLight1.state.toString)
end
```

### Publish

If the state of the Switch Item *aHA\_MakeLab\_HueLight1bsl* (the slider) changes, the value of the slider is published as

- **topic** “homeautomation/makelab/huelight1” with
- **payload** between 0 – 100.

```
rule "HueLight1 Brightness Changed"
when
  Item aHA_MakeLab_HueLight1bsl changed
then
  // Publish the message to topic using the specified MQTT broker.
  publish("openhabprod", "homeautomation/makelab/huelight1",
aHA_MakeLab_HueLight1bsl.state.toString)
end
```

# Function Anemometer

## Goal

The goal is to measure the wind speed (m/s, bft), direction and air temperature.

The anemometer used is the **TFA Dostmann 30.3168 Wind Meter** and placed on the garden shed ("Gartenhaus").

The anemometer is connected to the RFXCOM RFXtrx433E (see Thing Wind (TFA Dostmann 30.3168)).

## Example Sitemap

← Windmesser Gartenhaus		
	Direction	NNW
	Speed	1.1 m/s
	Bft	1
	Temperature	5.3 C
	Battery	100 %

## Items Configuration

This is an example of the items configuration related to the previous screen shot.

```
// Weather Home TFA Dostman Windsensor
Group gHA_Weather_Home_WindSensor "Windmesser Gartenhaus" <wind>
String sHA_Weather_WindSensor_Direction "Direction [%s]" <compass>
(gHA_Weather_Home_WindSensor)
Number nHA_Weather_WindSensor_DirectionDeg "Direction [%0f deg]" <compass> {
channel="rfxcom:wind:e64474fc:40214:windDirection" }
Number nHA_Weather_WindSensor_Speed "Speed [%0.1f m/s]" <speed>
(gHA_Weather_Home_WindSensor) { channel="rfxcom:wind:e64474fc:40214:windSpeed" }
Number nHA_Weather_WindSensor_Bft "Bft [%0f]" <speed>
(gHA_Weather_Home_WindSensor)
Number nHA_Weather_WindSensor_Temperature "Temperature [%0.1f C]" <temperature>
(gHA_Weather_Home_WindSensor) { channel="rfxcom:wind:e64474fc:40214:temperature" }
Number nHA_Weather_WindSensor_Battery "Battery [%0f %]" <battery>
(gHA_Weather_Home_WindSensor) { channel="rfxcom:wind:e64474fc:40214:batteryLevel"
}
```

## Sitemap Configuration

```
Frame label="Weather Pinneberg" {  
  ...  
  Group item=gHA_Weather_Home_WindSensor  
  ...  
}
```

## Rules Configuration

The rules are used to transform wind direction from degrees to cardinal and wind speed from m/s to beaufort. The Scale Transformation is used.

```
/*  
  Weather - Windmesser Direction  
*/  
rule "Windmesser Direction Changed"  
when  
  Item nHA_Weather_WindSensor_DirectionDeg changed  
then  
  var String d = transform("SCALE", "winddirection.scale",  
nHA_Weather_WindSensor_DirectionDeg.state.toString)  
  sHA_Weather_WindSensor_Direction.postUpdate( d )  
end
```

```
/*  
  Weather - Windmesser Speed in Bft  
*/  
rule "Windmesser Speed Changed"  
when  
  Item nHA_Weather_WindSensor_Speed changed  
then  
  var String bft = transform("SCALE", "windspeedmsbft.scale",  
nHA_Weather_WindSensor_Speed.state.toString)  
  nHA_Weather_WindSensor_Bft.postUpdate( bft )  
end
```

## Transformation Configuration

The transformation files used in the previous rules are located in folder  
/etc/openhab2/transform.

For transformation, the openHAB add-on Scale Transformation is used.

### Scale Winddirection

Transforms the Degree Direction to a Cardinal Direction.

#### Configuration Transformation File

```
/etc/openhab2/transform/winddirection.scale
```

#### Configuration Transformation Content (SCALE)

```
[348.75..11.25]=N  
[11.25..33.75]=NNE  
[33.75..56.25]=NE  
[56.25..78.75]=ENE  
[78.75..101.25]=E  
[101.25..123.75]=ESE  
[123.75..146.25]=SE  
[146.25..168.75]=SSE  
[168.75..191.25]=S  
[191.25..213.75]=SSW  
[213.75..236.25]=SW  
[236.25..258.75]=WSW  
[258.75..281.25]=W  
[281.25..303.75]=WNW  
[303.75..326.25]=NW  
[326.25..348.75]=NNW  
[..]=Undef
```

## Scale Windspeedmsbft

Transforms the Speed in m/s to Beaufort.

### Configuration Transformation File

```
/etc/openhab2/transform/windspeedkmsbft.scale
```

### Configuration Transformation Content (SCALE)

```
[0..0.2]=0  
[0.2..1.5]=1  
[1.5..3.3]=2  
[3.3..5.4]=3  
[5.4..7.9]=4  
[7.9..10.7]=5  
[10.7..13.8]=6  
[13.8..17.1]=7  
[17.1..20.7]=8  
[20.7..24.4]=9  
[24.4..28.4]=10  
[28.4..32.6]=11  
[32.6..]=12
```



# Function Waste Calendar












## Goal

The goal is to display waste calendar dates (“Abfall Termine”) for a city (location) and waste type.

The waste types are (in German): Rest, Bio, Gelb, Papier, Schad.

The dates are provided by the respective city waste management (“Abfallwirtschaft”) for the cities Pinneberg and Hohenfelde.

## Example Sitemap

← Abfall Termine			
	Pinneberg		Restmüll Pi 30-08-2018
	Bio Pi		23-08-2018
	Gelber Sack Pi		30-08-2018
	Papier Pi		24-08-2018
	Schadstoff Pi		-
	Hohenfelde		Restmüll Ho 27-08-2018
	Bio Ho		27-08-2018
	Gelber Sack Ho		28-08-2018
	Papier Ho		-

## Items Configuration

This is an example of the items configuration related to the previous screen shot.

```
Group    gHA_Waste                "Abfall Termine"    <garbagetruck>
String  SHA_Waste_Pi             "Pinneberg"        <pinneberg>        (gHA_Waste)
String  SHA_Waste_Pi_Rest        "Restmüll Pi [%s]"  <resttonne>        (gHA_Waste)
{mqtt="<[openhabprod:homeautomation/wastecalendar/pi/rest:state:default]"}
String  SHA_Waste_Pi_Bio         "Bio Pi [%s]"       <biotonne>         (gHA_Waste)
{mqtt="<[openhabprod:homeautomation/wastecalendar/pi/bio:state:default]"}
String  SHA_Waste_Pi_Gelb        "Gelber Sack Pi [%s]" <gelbersack>      (gHA_Waste)
{mqtt="<[openhabprod:homeautomation/wastecalendar/pi/gelb:state:default]"}
String  SHA_Waste_Pi_Papier      "Papier Pi [%s]"    <papiertonne>      (gHA_Waste)
{mqtt="<[openhabprod:homeautomation/wastecalendar/pi/papier:state:default]"}
String  SHA_Waste_Pi_Schad       "Schadstoff Pi [%s]" <schadstoff>      (gHA_Waste)
{mqtt="<[openhabprod:homeautomation/wastecalendar/pi/schad:state:default]"}
String  SHA_Waste_Ho             "Hohenfelde"       <hohenfelde>      (gHA_Waste)
String  SHA_Waste_Ho_Rest        "Restmüll Ho [%s]"  <resttonne>        (gHA_Waste)
{mqtt="<[openhabprod:homeautomation/wastecalendar/ho/rest:state:default]"}
String  SHA_Waste_Ho_Bio         "Bio Ho [%s]"       <biotonne>         (gHA_Waste)
{mqtt="<[openhabprod:homeautomation/wastecalendar/ho/bio:state:default]"}
String  SHA_Waste_Ho_Gelb        "Gelber Sack Ho [%s]" <gelbersack>      (gHA_Waste)
{mqtt="<[openhabprod:homeautomation/wastecalendar/ho/gelb:state:default]"}
String  SHA_Waste_Ho_Papier      "Papier Ho [%s]"    <papiertonne>      (gHA_Waste)
{mqtt="<[openhabprod:homeautomation/wastecalendar/ho/papier:state:default]"}
}
```

## Sitemap Configuration

```
Text label="Abfall Termine" icon="garbagetruck" {
    Text item=SHA_Waste_Pi
    Text item=SHA_Waste_Pi_Rest
    Text item=SHA_Waste_Pi_Bio
    Text item=SHA_Waste_Pi_Gelb
    Text item=SHA_Waste_Pi_Papier
    Text item=SHA_Waste_Pi_Schad
    Text item=SHA_Waste_Ho
    Text item=SHA_Waste_Ho_Rest
    Text item=SHA_Waste_Ho_Bio
    Text item=SHA_Waste_Ho_Gelb
    Text item=SHA_Waste_Ho_Papier
}
```

## Rules Configuration

The rule is used to execute, once a day or at start up, a python script (see next Python Script) to read the dates and publish as MQTT topic with payload.

### IMPORTANT

Ensure the owner of the Python script “wastecal.sh” is openhab and **NOT** root or openhabian.

```
/*
Waste Calendar
Ensure the owner of the script is openhab and NOT root or openhabian.
Make the script executable.
The CSV file contains entries starting with MQTT topic followed by the dates for
a waste type.
The dates are updated once a day at midnight.
*/
rule "Waste Calendar"
when
    Time is midnight or
    System started
then
    var String result = executeCommandLine("/etc/openhab2/scripts/wastecal.sh", 5000)
    logInfo("WasteCalendar", "*** Result: " + result)
end
```

### Log Entry executing the Python Script

```
2018-08-22 10:09:57.091 [INFO ] [smarthome.model.script.WasteCalendar] - ***
MQTT Connected with result code 0
homeautomation/wastecalendar/pi/papier 24-08-2018
homeautomation/wastecalendar/pi/rest 30-08-2018
homeautomation/wastecalendar/pi/bio 23-08-2018
Sending 1 day left notification.
homeautomation/wastecalendar/pi/gelb 30-08-2018
homeautomation/wastecalendar/ho/rest 27-08-2018
homeautomation/wastecalendar/ho/bio 27-08-2018
homeautomation/wastecalendar/ho/gelb 28-08-2018
```

## Concept

A Python script, “wastecal.py” executed via a Bash script, reads a CSV file, which contains per line an MQTT topic (waste type) and payload (waste date).

The topic is published based upon the difference in days between now and the waste date within a given interval (i.e. 14 or 31 days).

## CSV Input File

The CSV file is read line by line, using per line, the MQTT topic (field 1) to publish a date (fields 2 and more depending available dates).

A date is published if within the next 13 days from the actual date.

### *CSV Line Syntax*

homeautomation/wastecalendar/city/wastetype, interval, date1, dateN

MQTT Prefix	homeautomation/wastecalendar/
City	pi,ho
Waste Type	papier,rest,bio,gelb,schad
Interval	Number of days between waste dates, i.e. 14, 31
Date	Format Day-Month-Year dd-mm-YYYY, i.e. 09-08-2018

### Example CSV file line entry

The example shows city Hohenfelde (ho), waste type Papier (papier), an interval every 31 days and 2 dates.

homeautomation/wastecalendar/ho/papier,31,20-08-2018,17-09-2018
---

## Scripts

<b>Python Script</b>	/etc/openhab2/scripts/wastecal.py
Make script executable: sudo chmod +x /etc/openhab2/scripts/wastecal.py Make script owned by user openHAB: sudo chown openHAB /etc/openhab2/scripts/wastecal.py	
<b>Bash Script</b>	/etc/openhab2/scripts/wastecal.sh
Make script executable: sudo chmod +x /etc/openhab2/scripts/wastecal.sh Make script owned by user openHAB: sudo chown openHAB /etc/openhab2/scripts/wastecal.sh	

# Python Script

## Python MQTT Setup

The Python script, running Python 2.7, requires the paho-mqtt-python client library.

### Installation

```
cd /etc/openhab2/scripts
git clone https://github.com/eclipse/paho.mqtt.python.git
cd paho.mqtt.python
sudo python setup.py install
```

After installation, the library is stored in folder

```
/etc/openhab2/scripts/paho.mqtt.python
```

### Hint

To learn more about MQTT, read the file `/etc/openhab2/scripts/paho.mqtt.python/readme.rst`.

### Output Example

```
[19:18:30] openhabian@openHABianPi:/etc/openhab2/scripts$ git clone
https://github.com/eclipse/paho.mqtt.python.git
Cloning into 'paho.mqtt.python'...
Checking connectivity... done.
running install
running bdist_egg
running egg_info
creating src/paho_mqtt.egg-info
writing ...
Adding paho-mqtt 1.3.1 to easy-install.pth file
Installed /usr/local/lib/python2.7/dist-packages/paho_mqtt-1.3.1-py2.7.egg
Processing dependencies for paho-mqtt==1.3.1
Finished processing dependencies for paho-mqtt==1.3.1
```

### MQTT Test

Created a simple script to test MQTT topic subscription.

```
#!/usr/bin/env python
import time
import paho.mqtt.client as mqtt

def on_connect(client, userdata, flags, rc):
    print("Connected with result code " + str(rc))

    client.subscribe("homeautomation/#")

def on_message(client, userdata, msg):
    print(msg.topic + " " + str(msg.payload))

client = mqtt.Client()
client.on_connect = on_connect
client.on_message = on_message
client.connect("localhost", 1883, 60)
client.loop_forever()
```

## Output

```
python mqtttest.py
Connected with result code 0
homeautomation/windsensor/bft 1
homeautomation/windsensor/speed 0.8
homeautomation/windsensor/direction 292.0
```

## Python Script

Sample of the Python script “wastecal.py” without any comments.

```
#!/usr/bin/env python
import csv, sys, datetime, time
from datetime import date
import paho.mqtt.client as mqtt
wastecalendar = 'wastecal.csv'
if sys.version_info[0] == 2: # Not named on 2.6
    access = 'rb'
    kwargs = {}
else:
    access = 'rt'
    kwargs = {'newline':''}

def on_connect(client, userdata, flags, rc):
    print("MQTT Connected with result code " + str(rc))

client = mqtt.Client()
client.on_connect = on_connect
client.connect("localhost", 1883, 60)
client.loop_start()
s = datetime.date.today().strftime("%d-%m-%Y")
datenow = datetime.datetime.strptime(s, "%d-%m-%Y")
with open(wastecalendar, access, **kwargs) as csvfile:
    datesreader = csv.reader(csvfile, delimiter=',')
    for row in datesreader:
        for i in range(0, len(row)):
            csventry = row[i]
            if i == 0:
                topic = csventry

            if i == 1:
                interval = int(csventry)

            if i > 1:
                wastedate = csventry
                datecal = datetime.datetime.strptime(wastedate, "%d-%m-%Y")
                dnow = date(datenow.year, datenow.month, datenow.day)
                dcal = date(datecal.year, datecal.month, datecal.day)
                datesdelta = dcal - dnow
                if datesdelta.days >= 0 and datesdelta.days < interval:
                    payload = "%s" % (wastedate)
                    client.publish(topic, payload)
                    # TODO: Notification
                    if datesdelta.days == 1:
                        print 'Sending 1 day left notification.'

sys.exit()
```

The Python script is made executable using:

```
sudo chmod +x /etc/openhab2/scripts/wastecal.py
```

## Bash Script

Sample of the Bash script “wastecal.sh”.

Ensure to change directory to enable the scrip to read the file wastecal.csv.

```
#!/bin/bash  
cd /etc/openhab2/scripts  
python wastecal.py
```

The Bash script is made executable using:

```
sudo chmod +x /etc/openhab2/scripts/wastecal.sh
```

# Appendix: Setup volkszaehler

Description of the volkszaehler with the IR device, connected to a Raspberry Pi.  
[volkszaehler.org](http://volkszaehler.org) is an Open Source (GPL license) Smart Metering Hard- and Software  
Important to read the "[howto get started](#)" first to get an understanding of the concept.

## Hardware

- Raspberry Pi Model 2 with WLAN and a 16GB SD Card.
- Power Meter EMH ED300L.

## Software

- volkszaehler RPi image

## Setup Raspberry Pi

Download the volkszaehler image to a Windows PC, unpack the image, write the image using win32diskimager to a 16GB SD card. Stick the SD card in the Raspberry Pi and power on.

*Note:* This setup was done in 2015 – so things might have changed – checkout the volkszaehler website.

## Option 1 Raspberry Pi WLAN Network fixed IP address

The communication uses WLAN with a static network address.

Set a static IP address by changing `/etc/network/interfaces` and `/etc/wpa_supplicant/wpa_supplicant.conf`.

File `/etc/network/interfaces`:

Editing via **sudo nano /etc/network/interfaces**

```
auto lo
iface lo inet loopback
auto wlan0
iface wlan0 inet manual
wireless-power off
wpa-conf /etc/wpa_supplicant/wpa_supplicant.conf
iface wlan0 inet static
address 192.168.N.NN
netmask 255.255.255.0
broadcast 192.168.0.255
gateway 192.168.N.NN
```

**Edit sudo nano /etc/wpa\_supplicant/wpa\_supplicant.conf**

```
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1
network={
    ssid="YOURSSID"
    proto=RSN
    key_mgmt=WPA-PSK
```



```

pairwise=CCMP
group=CCMP
psk="YOURPSK"
}

```

## Option 2 Raspberry Pi ETH Network fixed IP address

The volkszahler Raspberry Pi and the Home Automation Raspberry Pi are direct connected via an Ethernet cable. A standard cable is used, no twisted wires required.

The communication uses ETH0 with a static network addresses on both sides.

The ETH0 network address must be different then the WLAN network address.

<b>volkszahler Raspberry Pi</b> <b>[Raspbian GNU/Linux 8 (jessie)]</b> <b>Linux 39 4.1.19-v7+ #858</b>	<b>Home Automation Raspberry Pi</b> <b>[Raspbian GNU/Linux 8 (jessie)]</b> <b>Linux openHABianPi 4.9.35-v7+ #1014</b>
eth0 inet addr:169.254.87.85 Bcast:169.254.255.255 Mask:255.255.0.0	eth0 inet addr:169.254.87.84 Bcast:169.254.255.255 Mask:255.255.0.0
Set the static Ethernet network Address sudo nano /etc/dhcpd.conf	sudo nano /etc/dhcpd.conf
Add the lines	
interface eth0 static ip_address=169.254.87.85 static routers=169.254.0.1	interface eth0 static ip_address=169.254.87.84 static routers=169.254.0.1

### Notes

- \$ uname -a to determine the Linux version
- \$ cat /etc/os-release to determine the Raspberry Pi release

The direct communication using an Ethernet cable is used.

## volkszaehler

### PowerMeter Setup

Powermeter: [EMH ED300L](#)

The powerpuls are captured using an [IR-write-reader device](#).

This device is connected via USB to the Raspberry Pi 2 Port /dev/ttyUSB22.

### Notes

- Check the connected USB devices with command **ls /dev/tty\***
- Important when changing the Raspberry Pi or adding USB devices, use the same port for the IR-write-reader device ([read](#)).

Persistent USB devices are defined using **sudo nano /etc/udev/rules.d/99-usb-serial.rules**



Ensure port is set during boot

Add this line into /etc/rc.local (by `sudo nano /etc/rc.local`)

**stty -F /dev/ttyUSB1**

**1:0:8bd:0:3:1c:7f:15:4:5:1:0:11:13:1a:0:12:f:17:16:0:0:0:0:0:0:0:0:0:0:0:0:0**

Check Data coming from the USB device (option 1): **cat /dev/ttyUSB1 | od -tx1**

Output Example:

```
00000000 2f 38 01 01 63 e4 c7 00 76 07 00 0a 00 07 db 75
00000020 62 00 62 00 72 63 07 01 77 01 0b 09 01 4d 48 00
...
```

Check Data coming from the USB device (option 2): **xxd </dev/ttyUSB1**

## Define Channels

To define the channels, reboot the RPi and access via Webbrowser the volkszaehler Frontend

<http://<RaspberryPi IP Address>/Frontend> (f.e. <http://192.168.N.NN/Frontend>)

Define Channels using the volkszaehler FrontEnd

```
Channel 1 =
{"version":"0.3","entity":{"uuid":"d14e9a80-8894-11e5-9dc8-298a9a1001ac","type":"current","active":true,"color":"aqua","description":"Strom Direktverbrauch","fillstyle":0,"public":true,"resolution":10000,"style":"steps","title":"Direktverbrauch","yaxis":"auto"}}

Channel 2 =
{"version":"0.3","entity":{"uuid":"e460bec0-8894-11e5-b0d8-3f141d19092c","type":"current","active":true,"color":"aqua","description":"Strom Gesamtverbrauch","fillstyle":0,"public":true,"resolution":10000,"style":"steps","title":"Gesamtverbrauch","yaxis":"auto"}}

Channel 3 = 1.8.0
{"version":"0.3","entity":{"uuid":"23844c00-8895-11e5-a642-6d8ce36c9a44","type":"electric meter","active":true,"color":"#0033ff","cost":0.1992,"description":"Stromverbrauch Fuchsbau","fillstyle":0.2,"initialconsumption":20,"public":true,"resolution":1000,"style":"steps","title":"Stromverbrauch","yaxis":"auto"}}

Channel 4 = 16.7.0
{"version":"0.3","entity":{"uuid":"a3c76600-8895-11e5-9432-333f81e04451","type":"current","active":true,"color":"aqua","description":"Wirkleistung","fillstyle":0,"public":true,"resolution":10000,"style":"steps","title":"Wirkleistung","yaxis":"auto"}}
```

## vzlogger Setup

The defined channels have to be defined in the file /etc/vzlogger.conf (sudo nano /etc/vzlogger.conf). Two channels are defined "Stromverbrauch" and "Wirkleistung".

```
{
"retry" : 30, /* how long to sleep between failed requests, in seconds */
"daemon": true, /* run periodically */
"verbosity" : 1, /* between 0 and 15 */
"log" : "/var/log/vzlogger.log", /* path to logfile, optional */
"local" :
{
"enabled" : false, /* start the local HTTPd for serving live readings? */
```

```
"port" : 80, /* the TCP port for the local HTTPd */
"index" : true, /* should we provide a index listing of available channels if no
UUID was requested? */
"timeout" : 30, /* timeout for long polling comet requests, 0 disables comet, in
seconds */
"buffer" : 600 /* how long to buffer readings for the local interface (secs)*/
"meters" : [{
  "enabled" : true,          /* disabled meters will be ignored */
  "protocol" : "sml",        /* see 'vzlogger -h' for list of available prot$ */
  "device" : "/dev/ttyUSB22",
  "channels": [{
    "uuid" : "a3c76600-8895-11e5-9432-333f81e04451",
    "middleware" : "http://localhost/middleware.php",
    "identifier" : "1-0:16.7.0", /* Wirkleistung */
  }, {
    "uuid" : "23844c00-8895-11e5-a642-6d8ce36c9a44",
    "middleware" : "http://localhost/middleware.php",
    "identifier" : "1-0:1.8.0", /* Stromverbrauch */
  }]
}]
}
```

## vzlogger Service

Ensure the vzlogger service is correct defined, esp the vzlogger command is using the parameter -c.

```
cd /etc/systemd/system
sudo nano vzlogger.service
[Unit]
Description=vzlogger
After=syslog.target network.target
Requires=mysql.service
[Service]
ExecStart=/usr/local/bin/vzlogger -c /etc/vzlogger.conf
ExecReload=/bin/kill -HUP $MAINPID
StandardOutput=null
[Install]
WantedBy=multi-user.target
```

## vzlogger Commands

For test purposes a vzlogger.test can be used to check data flow ([vzlogger configuration](#)).

```
{
  "retry" : 30, /* how long to sleep between failed requests, in seconds */
  "daemon": false,          /* run periodically */
  "verbosity" : 15,         /* between 0 and 15 */
  "log" : "/var/log/vzlogger.log", /* path to logfile, optional */
  "local" :
  {
    "enabled" : false, /* start the local HTTPd for serving live readings? */
    "port" : 80, /* the TCP port for the local HTTPd */
    "index" : true, /* provide a index listing of available channels? */
    "timeout" : 30, /* timeout for long polling comet requests, 0 disables comet, sec*/
    "buffer" : 600 /* how long to buffer readings for the local interface, secs */
  },
  "meters" :
  [{
    "enabled" : true,          /* disabled meters will be ignored */
```

```
"protocol" : "sml",      /* use 'vzlogger -h' for list of available protocols */
"device" : "/dev/usb-ir-lesekopf0",
},{
"enabled" : true,        /* disabled meters will be ignored */
"protocol" : "sml",      /* use 'vzlogger -h' for list of available protocols */
"device" : "/dev/usb-ir-lesekopf1",
}
}
```

vzlogger Start: `vzlogger -c /etc/vzlogger.conf`

vzlogger Check status: `systemctl status vzlogger`

Output Example:

```
? vzlogger.service - vzlogger
   Loaded: loaded (/etc/systemd/system/vzlogger.service; enabled)
   Active: active (running) since Fri 2015-11-13 11:55:53 CET; 13min ago
 Main PID: 670 (vzlogger)
    CGroup: /system.slice/vzlogger.service
            +-670 /usr/local/bin/vzlogger -c /etc/vzlogger.conf
 Nov 13 11:55:53 raspberrypi systemd[1]: Started vzlogger.
```

vzlogger Start: `systemctl start vzlogger`

vzlogger Stop: `systemctl stop vzlogger`

vzlogger Enable as a Service: `systemctl enable vzlogger`

Can also use: `service vzlogger start`, `service vzlogger stop`

vzlogger Daemon reload: `systemctl daemon-reload`

vzlogger [Debug](#): `$vzlogger -f`

### API volkszaehler Hints

API Reference: <http://wiki.volkszaehler.org/development/api/reference>

API Obtain Configuration Request:

<http://192.168.N.NN/middleware.php/entity/23844c00-8895-11e5-a642-6d8ce36c9a44.json>

Result

```
{"version":"0.3","entity":{"uuid":"23844c00-8895-11e5-a642-6d8ce36c9a44","type":"electric meter","active":true,"color":"#0033ff","cost":0.0001992,"description":"Stromzaehler","fillstyle":0.2,"initialconsumption":20,"public":true,"resolution":1000,"style":"steps","title":"Stromzaehler","yaxis":"auto"}}
```

API Channel Properties

<http://192.168.N.NN/middleware.php/channel/23844c00-8895-11e5-a642-6d8ce36c9a44.json>

Result

```
{"version":"0.3","entity":{"uuid":"23844c00-8895-11e5-a642-6d8ce36c9a44","type":"electric meter","active":true,"color":"#0033ff","cost":0.0001992,"description":"Stromzaehler","fillstyle":0.2,"initialconsumption":20,"public":true,"resolution":1000,"style":"steps","title":"Stromzaehler","yaxis":"auto"}}
```

API Entities

<http://192.168.N.NN/middleware.php/capabilities/definitions/entities.json>

API Read Data Requests

### Notes

Relative Dateformats = <http://de.php.net/manual/en/datetime.formats.relative.php>

Get All Data

<http://192.168.N.NN/middleware.php/data/23844c00-8895-11e5-a642-6d8ce36c9a44.json>

### Get One Record

<http://192.168.N.NN/middleware.php/data/23844c00-8895-11e5-a642-6d8ce36c9a44.json?tuples=1>

#### Result

```
{ "version": "0.3", "data": { "tuples": [[1447316802613, 246.491, 19046]], "uuid": "23844c00-8895-11e5-a642-6d8ce36c9a44", "from": 1447262000365, "to": 1447316802613, "min": [1447316802613, 246.49134831111], "max": [1447316802613, 246.49134831111], "average": 246.491, "consumption": 3752.3, "rows": 1 }}
```

### Get the last value

<http://192.168.N.NN/middleware.php/data/23844c00-8895-11e5-a642-6d8ce36c9a44.json?from=now&to=now>

#### Result

```
{ "version": "0.3", "data": { "tuples": [[1447317119885, 201.681, 1], [1447317121700, 198.347, 1]], "uuid": "23844c00-8895-11e5-a642-6d8ce36c9a44", "from": 1447317118100, "to": 1447317121700, "min": [1447317121700, 198.34710743513], "max": [1447317119885, 201.68067227331], "average": 200, "consumption": 0.2, "rows": 3 }}
```

### API Logging

<http://192.168.N.NN/middleware.php/data/23844c00-8895-11e5-a642-6d8ce36c9a44.json?ts=1284677961150&value=12>

Using the volkzaehler Frontend to display data

<http://192.168.N.NN/frontend/?uuid=23844c00-8895-11e5-a642-6d8ce36c9a44>

# Appendix: Test Basic UI

## Objectives

The objective is to create a Basic UI test setup with 3 items to test the communication with the RFXCOM device.

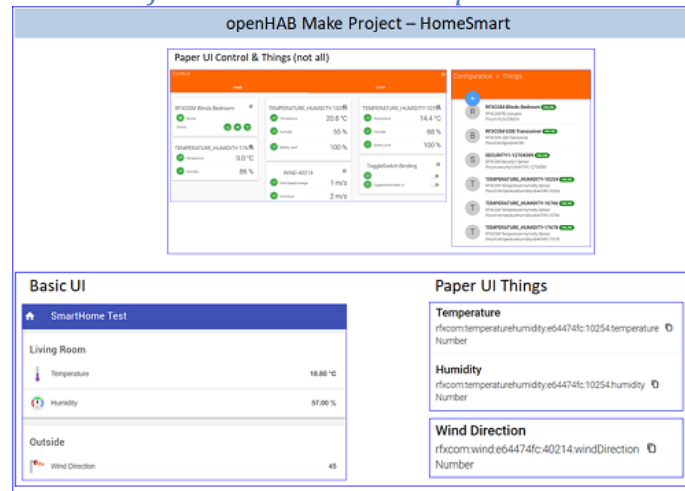
- Living Room Temperature, Humidity from the Thing TEMPERATURE\_HUMIDITY-10254.
- Outside Wind Direction from the Thing WIND-40214.

*Note*

For the “Home Automation” solution, the TFA Wind Meter is replaced by the OpenWeatherMap Service as more data is provided.

- No naming convention used for this test setup

*Screenshot of the Items in the Paper UI and Basic UI*



## Items Configuration

The Items are using selective Channels of the Things. The Channel information is listed in the Paper UI under Configuration > Things > TEMPERATURE\_HUMIDITY-10254, i.e.

```
channel="rfxcom:temperaturehumidity:e64474fc:10254:temperature"
```

### Configuration Items File

```
/etc/openhab2/items/ homeautomation.items
```

### Configuration Items Content (using the Channel information)

```
Number RFXTemp_LivingRoom "Temperature [%2f °C]" <temperature> (LivingRoom) {
  channel="rfxcom:temperaturehumidity:e64474fc:10254:temperature" }
Number RFXHum_LivingRoom "Humidity [%2f %]" <humidity> (LivingRoom) {
  channel="rfxcom:temperaturehumidity:e64474fc:10254:humidity" }
```

```
Number RFXWind_Direction "Wind Direction [%d]" <wind> (Outside) {  
channel="rfxcom:wind:e64474fc:40214:windDirection" }
```

# Sitemap Configuration

## Configuration Sitemap File

```
/etc/openhab2/sitemaps/ homeautomation.sitemap
```

## Configuration Sitemap Content (using the Items defined)

```
sitemap homeautomation label="Home Automation Test"  
{  
  Frame {  
    Text item=RFXTemp_LivingRoom  
    Text item=RFXHum_LivingRoom  
  }  
  Frame {  
    Text item=RFXWind_Direction  
  }  
}
```

# Rules Configuration

## Configuration Rules File

```
/etc/openhab2/rule/ homeautomation.rules
```

## Configuration Rules Content (using an Item)

```
rule "WindDirection Change"  
when  
  Item RFXWind_Direction changed  
then  
  logInfo("RFXCOM Binding", "*** RFXWind_Direction change state to " +  
RFXWind_Direction.state)  
end
```

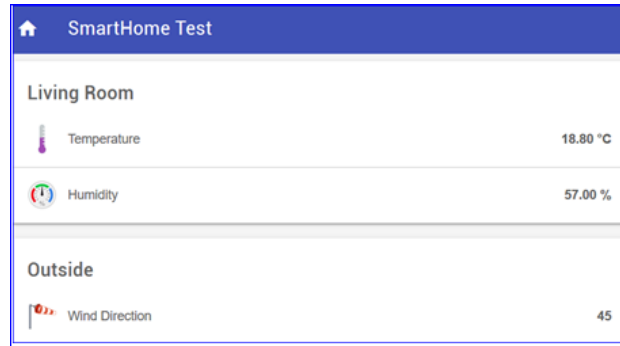
## Log entry Example

```
2017-12-31 13:27:29.664 [INFO ] [marthome.model.script.RFXCOM Binding] - ***  
RFXWind_Direction change state to 22.0
```



# Check Configuration

In the web browser, open the Basic UI, select sitemap “homeautomation” which should result in content like:



The content is updated when a channel changes, which is logged.

Example openHAB log file entry:

```
2018-01-01 15:53:09.218 [vent.ItemStateChangedEvent] - WIND40214_WindDirection
changed from 45.0 to 67.0
```

First important step taken, the RFXCOM device is working.

# Appendix: openHAB Hints

## Restart

```
sudo systemctl restart openhab2.service
```

## Clean-up Cache and Temp files

```
sudo su
systemctl stop openhab2
rm -r /var/lib/openhab2/cache/*
rm -r /var/lib/openhab2/tmp/*
```

## Create a Backup

```
cd /usr/share/openhab2/runtime/bin
sudo ./backup

#####
      openHAB 2.x.x backup script
#####

Using '/etc/openhab2' as conf folder...
Using '/var/lib/openhab2' as userdata folder...
Using '/var/lib/openhab2/backups' as backup folder...
Making Temporary Directory
Copying configuration to temporary folder...
Removing unnecessary files...
Backup Directory is inside userdata, not including in this backup!
Zipping folder...
Removing temporary files...
Success! Backup made in /var/lib/openhab2/backups/openhab2-backup-18_01_04-
19_34_14.zip
```

# Clear the Paper UI Inbox

See [openHAB Runtime Commands](#)

Open a terminal session and run the inbox clear command from the openHAB-cli console. If the items are removed, check if the Paper-UI inbox has been cleared.

```
openhab-cli console
password: habopen

Clear the Inbox
smarthome:inbox clear
```

When clearing the inbox, the items being removed are logged.

Example:

```
REMOVED [NEW]: rfxcom:temperature [label=TEMPERATURE-43011,
thingId=rfxcom:temperature:e64474fc:43011, bridgeId=rfxcom:bridge:e64474fc,
properties={deviceId=43011, subType=TEMP10}]
```

# Appendix: To-Do-List

<b>Item</b>	<b>Log Warnings after editing configuration files with VSCode Editor (openHAB 2.2)</b>
Description	Example rrd4j.persist – but happens also for rules & sitemap files: <i>2018-01-04 12:18:49.241 [WARN ] [el.core.internal.ModelRepositoryImpl] - Configuration model 'rrd4j.persist' has errors, therefore ignoring it: [1,1]: mismatched input '&lt;EOF&gt;' expecting 'Strategies'</i>
Cause	By the editor used, in this case VSCode, to change the configuration file. Although the message occurs, the file is used.
Status SOLVED	The warning message did not occur when editing with <i>nano</i> from a terminal. All spaces starting at the beginning of a line have been replaced by proper tabs.
<b>Item</b>	<b>Paper-UI remove Items (openHAB 2.2)</b>
Description	Remove items which are created by the Paper UI instead of the Basic UI Textual Configuration files.
Status SOLVED	Deleted all related entries from the jsondb files located in /var/lib/openhab2/jsondb/. To remove items: 1. Stop openHAB 2. Change the files with a text editor 3. org.eclipse.smarthome.core.items.Item.json 4. org.eclipse.smarthome.core.thing.link.ItemChannelLink 5. Start openHAB Open the Paper UI and check the Configuration > Items
<b>Item</b>	<b>Selective Charts “broken pipe” log error and the Basic UI hangs</b>
Description	If a selective chart is embedded in a frame, the chart hangs. See github issue: <a href="#">reference</a> . There is an Image IO exception when writing the PNG file, caused by an EndOfFile exception, caused by an IO Exception (broken pipe)
Cause	The sitemap refresh value is too low.
Status SOLVED	Changed from 15ms to 60000ms. Tests ok so far.
<b>Item</b>	<b>Item to open an URL (openHAB 2.2)</b>
Description	Click on an Item in the sitemap to open an URL.
Status OPEN	Have not found a way to do so.
<b>Item</b>	<b>Java UI Tool to read rrd files</b>
Description	Read the content of the rrd file and output in JSON format timestamp:value.
Status OPEN	Started to look into the source code of the rrd4j Binding to understand ways of working.

Item	Implement Error Handling in Rules
Description	<p>The rules are using methods like:  sendHttpRequest, sendMail (Mail service), publish (MQTT action)  In case of network issues, set up an <i>try .. catch .. finally</i> error handler.</p> <p>Example</p> <pre>rule "Energy Power Consumption Update" when     Time cron "0 */5 * ? * *" then     val Number timeFrom = now.millis - 300000;     try {         var String result = sendHttpRequest(...)         var String pc = transform("JSONPATH", "\$.data.average", result)         HA_Energy_PowerConsumption.postUpdate(c)         logInfo("POWERCONSUMPTION UPDATE", "Power Consumption: " + pc)     }     catch(Throwable t) {         logError("POWERCONSUMPTION UPDATE", "Error: " + t.toString)         sendMail(...)     }     finally {         // this runs always     } end</pre>
Status STARTED	

Item	Power Consumption Rule Error during openHAB Start
Description	<p>When openHAB starts, an error is logged for the Rule which sends HTTP request for the power consumption, to the volkszaehler Raspberry Pi. This happens once and next time the power consumption is requested, the information is correct.</p> <p>[ERROR] [ntime.internal.engine.ExecuteRuleJob] - Error during the execution of rule 'Energy Power Consumption Update': The name 'HA_Energy_PowerConsumption' cannot be resolved to an item or type; line 141, column 3, length 26</p>
Cause	<p>Not sure, but think the Rule 'Energy Power Consumption Update' is executed before the textual item configuration (Basic UI) has been loaded, because the log shows:</p> <p>[ERROR] [ntime.internal.engine.ExecuteRuleJob] - Error during the execution of rule 'Energy Power Consumption Update': The name 'HA_Energy_PowerConsumption' cannot be resolved to an item or type; line 141, column 3, length 26</p> <p>[INFO ] [.dashboard.internal.DashboardService] - Started dashboard at http://192.168.0.18:8080</p> <p>[INFO ] [.dashboard.internal.DashboardService] - Started dashboard at https://192.168.0.18:8443</p> <p>[INFO ] [basic.internal.servlet.WebAppServlet] - Started Basic UI at /basicui/app</p> <p>The rule is using a cron job: Time cron "0 */5 * ? * *". This could be the trigger.</p> <p>NOTE: The same happens for MQTT, i.e.</p> <p>[ERROR] [ntime.internal.engine.RuleEngineImpl] - Rule 'MQTT Publish Power Consumption': The name 'publish' cannot be resolved to an item or type; line 161, column 2, length 1052018-</p> <p>[INFO ] [penhab.io.transport.mqtt.MqttService] - MQTT Service initialization completed.</p> <p>[INFO ] [t.mqtt.internal.MqttBrokerConnection] - Starting MQTT broker connection 'openhabprod'</p>
Status OPEN	Leave for now as only happens during openHAB start.
Item	Log entry Found multiple local interfaces - ignoring
Description	<p>During openHAB start, the log shows:</p> <p>[WARN ] [g.eclipse.smarthome.core.net.NetUtil] - Found multiple local interfaces - ignoring 169.254.87.84.</p> <p>The address is configured as (ifconfig):</p> <p>eth0 inet addr:169.254.87.84 Bcast:169.254.255.255 Mask:255.255.0.0</p>
Cause	Unknown
Status OPEN	
Item	Add Remote Controlled Switches
Description	<p>Add two remote controlled switches, after trying several types, i.e. Mumbai FS300.</p> <p>Note: Need to try following as guided by RFXCOM FAQ for the FS300. The remote transmits several protocols. The protocol used by the wall plug is not received by the RFXtrx but some other protocols are received.</p> <p>Solution: For the HomeEasy EU- HE8xx series: reset the module to remove all paired remotes and pair the module with the RFXtrx433E and one of remote codes that is received.</p>

	<p>BUT must be clear that the somfy blinds need to be paired again.</p> <p>The Item configuration for testing:</p> <pre>Group gHA_MakeLab Switch HA_MakeLab_PowerSwitch_Desk "Desk Switch" (gHA_MakeLab) { channel="rfxcom:lighting2:e64474fc:2631682_1:command" } String HA_MakeLab_PowerSwitch_Desk_Info { rfxcom="&lt;2631682.1:RawData" } Switch HA_MakeLab_PowerSwitch_Desk2 "Desk Switch2" (gHA_MakeLab) { rfxcom="&lt;2631682.2:LIGHTING2.AC:Command" }</pre>
Cause	
Status OPEN	
<b>Item</b>	<b>Raspberry Pi Memory Used Continuous Increase</b>
Description	<p>Check memory usage with: <code>free -o -h</code>.</p> <p>The memory used is increasing fast. Watched over 12h from 55% to 68%. Need to investigate why?</p> <p>Various actions taken (stopped brickd, node-red) until note that indeed an</p>
Cause	<p>Found on the Raspberry Pi forum issue with the Firmware related to WiFi network driver.</p> <p>Action taken:</p> <p>Installed &amp; run tool <a href="#">rpi-update</a> to upgrade the Raspberry Pi firmware from 4.9.35-v7+ to 4.9.76-v7+</p>
Status SOLVED	