Workbook Domoticz Home Automation

Exploring Domoticz Home Automation System

by Robert Linn, Pinneberg, Germany

06.02.2019

DISCLAIMER

Table of Contents

Objectives

To build a Home Automation Solution, running on a Raspberry Pi with Domoticz Home Automation System.

As a Domoticz Beginner

- ✓ Main Purpose is to explore & learn Domoticz & Scripting.
- ✓ Get an initial Home Automation Solution up & running.
- ✓ Write up experiences during development … and build further.

Notes:
There might be better solutions for what is shared – but the solution works fine so far.

This document is a working document – todo's will never cease to exist☺.

Functions

- Display temperature & humidity measured in rooms.
- Charts for selective weather items, room temperature & humidity.
- Control Somfy roller shutters with RTS motors in rooms.
- Philips Hue Lighting System control via Hue Bridge for ZigBee devices.
- Security door & window wireless contact detectors.
- Information on some key dates (Calendar type information).
- MQTT subscribe & publish messages to trigger actions or information.
- Raspberry Pi system information with charts and threshold email notification.
- Energy Power Consumption metering from "volkszaehler" with charts.
- *More features are added whilst exploring Domoticz*

Explore How To

- Setup & configure Domoticz running on a Raspberry Pi.
- Use the RFXCOM RFXtrx433E USB RF Transceiver for
  - o Temperature & Humidity devices.
  - o External Wind device (only for RFXCOM tests).
  - o Other 433Mhz devices, i.e. door & window contacts.
- Use external services, i.e. OpenWeatherMap.
- Use MQTT messaging.
- Use bindings like MQTT, Astro, Philips Hue.
- Use Node-RED as an alternative script engine and User Interface.
- Use a, to be determined, Domoticz Android App (native client).
- Create advanced User Interfaces, i.e. Node-RED, Bootstrap …
- Scripting using Python, Lua, dzVents, JavaScript.

Credits

To the developers of Domoticz and to all sharing information about Domoticz. Without these, it would not be possible to build this project and write the workbook.

Components

Hardware and software used for this project.
More details to be found in sections, like Setup , Functions and other.

Hardware

- Raspberry Pi 3 Model B+ v1.2
- RFXCOM RFXtrx433E USB 433.92MHz Transceiver
- TFA Dostmann TS34C Temperature & Humidity device
- TFA Dostmann 30.3168 Wind Meter (speed, direction, temperature)
- Somfy Blinds RTS Pure
- PB-62R Door & window wireless contact detector
- Philips Hue Bridge
- volkszaehler Raspberry Pi 2, Smart Meter EMH ED300L
- Power Switches
- BPM280 Barometer
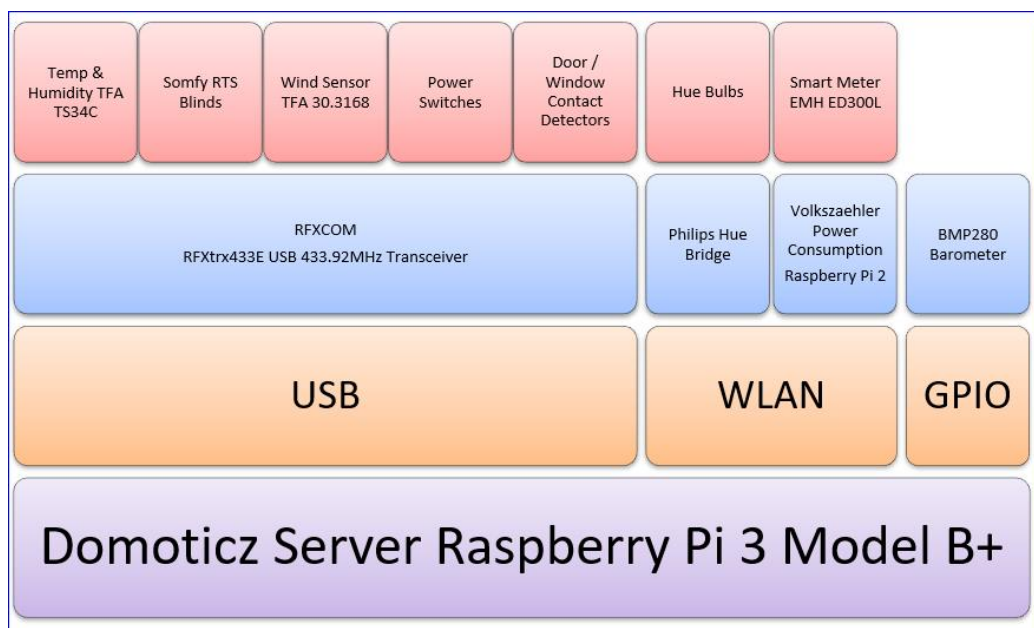
Software

- Raspbian GNU/Linux 9 (Stretch) based upon NOOBS Version 2.8.2, 2018-06-27
- Domoticz V4.1 BETA.
  Note: BETA version to ensure include latest fixes (i.e. Philips Hue).
- RFXCOM RFXflash Programmer 8.0.0.0 - to update the firmware on the RFXtrx433E.
- RFXCOM RFXmngr 18.0.0.18 - to test and manage RFXtrx433E connected devices.
- WinSCP - to exchange files between the Development Device and the Raspberry Pi's.
- PuTTY - to run terminal commands
- Thonny - Python Script development

Notes:
The software versions mentioned used during the initial setup of the project. In the mean time check out for newer versions available.

Setup

Various setup steps prior starting to build the solution in Domoticz.

Development Device

A **Notebook**, running **Windows 10**, is used to support the setup and as development device. Additional software installed:

- [RealVNC](#) - Remote access Raspberry Pi Desktop
- [Python](#) - Script development and testing
- [Thonny](#) - Python IDE for script development
- [Notepad++](#) - Text and source code editor

Raspberry Pi

Installation

The Raspberry Pi running as the Domoticz Home Automation server, has a monitor, keyboard and mouse connected. This to ease the setup.
After initial installation, the Raspberry Pi is running headless and VNC is used to remote access the Raspberry Pi desktop.

NOOBS
Download the Raspberry Pi NOOBS version from [here](#) .
NOOBS Version 2.8.2, 2018-06-27, Zip archive NOOBS_v2_8_2.zip.

On the Development PC
- insert a new SD card (used 32GB)
- format the SD card (used SDFormatter)
- unzip NOOBS_v2_8_2.zip to the SD card

On the Raspberry Pi
- insert the SD card (ensure no power connected)
- connect power to the Raspberry Pi
- from the initial installation menu
    - select Raspian
    - select expand partition (extra 512 MB)
    - set WiFi connection
    - press install
    - installation procedure starting, wait for completion (~10 minutes)
    - reboot
    - complete the Assistant steps incl. Check for Updates (~20 minutes)

Additional configuration Desktop Menu Preferences > Raspberry Pi Configuration > Interfaces enabled SSH, VNC, SPI, I2C, 1-Wire
Any other software for the project [Functions](#) are described under Functions.

WLAN Static IP

It is useful to define a static IP address for the Raspberry Pi running Domoticz, i.e. fix IP address to access Domoticz from a Browser (http://rpi-domoticz-ip:8080) and for a Samba shared folder used f.e. to edit Scripts from the Development PC or backup or read the Domoticz database.

For release Raspbian GNU/Linux 8 (stretch), set a fixed IP address using the Desktop. This is done by right clicking on the LAN icon at the top right of the desktop and select menu:

Wireless & Wired Network Settings.

```
Select Configure > Interface > wlan0:
set IPv4: NNN.NNN.N.NNN, Router: NNN.NNN.N.NNN
Apply and Close.
```

Notes
Just as an FYI, the file */etc/dhcpcd.conf* is updated by the Raspberry Pi Desktop application.

```
sudo nano /etc/dhcpcd.conf
interface wlan0
inform NNN.NNN.N.NNN
static routers=NNN.NNN.N.NNN
```

WLAN Router SSID
```
sudo nano /etc/wpa_supplicant/wpa_supplicant.conf
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1
country=DE

network={
      ssid="*****"
      psk="*****"
      key_mgmt=WPA-PSK
}
```

Reboot the Raspberry Pi and check the WLAN address

```
ifconfig
Output:
wlan0     Link encap:Ethernet  HWaddr b8:27:eb:fa:44:fc
          inet addr:NNN.NNN.N.NN  Bcast:NNN.NNN.0.255  Mask:255.255.255.0
...
```

Note
In this documentation the IP address of the Raspberry Pi Domoticz Server is referred to as:
http://rpi-domoticz-ip:8080

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
```

Output

```
Power save: on
```

Notes
The command *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 crontab

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

Reboot the Raspberry Pi and check power save mode

```
iw wlan0 get power_save
Power save: off
```

Raspian Check Version & Update

Raspian Check version installed.

```
cat /etc/os-release
PRETTY_NAME=" "
NAME="Raspbian GNU/Linux"
VERSION_ID="9"
VERSION="9 (stretch)"
ID=raspbian
ID_LIKE=debian
HOME_URL="http://www.raspbian.org/"
SUPPORT_URL="http://www.raspbian.org/RaspbianForums"
BUG_REPORT_URL="http://www.raspbian.org/RaspbianBugs"
```

Raspian Update to the latest version.

Perform regular updates (doing this once a week):

```
Desktop Select Menu > Preferences > Add / Remove Software > Options > Check for
Updates
```

Or run alternative from a Terminal:

```
sudo apt-get update && sudo apt-get-upgrade
```

Samba

Start by reading [here](#).
Hints: Do not forget to set the password and restart Samba.

```
sudo smbpasswd -a pi
sudo systemctl restart smbd.service
```

Domoticz Home Folder as a share

The Domoticz home folder /home/pi/domoticz can also be set as a shared folder.

Prework Raspberry Pi Domoticz Server

The shared folder, DoProDomoticz, uses samba and is configured with:

```
sudo nano /etc/samba/smb.conf
```

Content
```
[DoProDomoticz]
Comment = Raspberry Pi Domoticz Production folder
Path = /home/pi/domoticz
Browseable = yes
Writeable = Yes
only guest = no
create mask = 0777
directory mask = 0777
Public = yes
Guest ok = yes
```

Notes on the Samba settings:

[DoProDomoticz]	Name of the share
comment	Text is displayed as Comments in the Share detail view
path	Specifies the folder with the shared files
browsable	yes|no - Set share visible when running *net view* command and browsing network shares.
writable	yes|no - Allows user to add/modify files and folders. Default samba shares are readonly
guest ok	yes|no - Allows "non authenticated" users to access the share

After configuration: Perform following steps after the Samba configuration.

Ensure to set the directory mask for the domoticz folder to 0777:

```
sudo chmod 777 /home/pi/domoticz
```

Restart samba:

```
sudo /etc/init.d/samba restart
```

Restart domoticz:
```
sudo service domoticz.sh restart
```

Prework Windows

In Windows (Version 10 is used) connect to the shared folder using the Windows Explorer > Connect Network:

Path for drive Z: is \\ rpi-domoticz-ip\DoProDomoticz
(DoProDomoticz is previous defined in the samba configuration)

To connect from the Windows explorer:
Network drives:
Add new drive with url [\\rpi-domoticz-ip\DoProDomoticz](\\rpi-domoticz-ip\DoProDomoticz)
Username: pi, Password: ******

Example Music Shared Folder

Another example in setting up a music shared folder "MyMusic" pointing to folder /home/pi/music.

Create the folder music:

```
mkdir /home/pi/music
```

Add to the Samba configuration file, /etc/samba/smb.conf

```
[MyMusic]
 comment=Raspberry Pi Music Share
 path=/home/pi/music
 browseable=Yes
 writeable=Yes
 only guest=no
 create mask=0777
 directory mask=0777
 public=yes
```

Ensure to set the directory mask for the folder to 0777:

```
sudo chmod 777 /home/pi/music
```

Restart samba:

```
sudo /etc/init.d/samba restart
```

Restart domoticz:

```
sudo service domoticz.sh restart
```

Connect from Windows explorer using
URL: [\\rpi-domoticz-ip\MyMusic](\\rpi-domoticz-ip\MyMusic)
Username: Pi
Password: ******

Persistent USB Devices

On the Raspberry Pi, the RFXtrx433E device is connected to an USB port.
If multiple USB devices connected, the device order could change after reboot, resulting in
Domoticz using the wrong devices.
To ensure the right USB Port is used by Domoticz, a symbolic link (**symlink**) is assigned for
the USB Port associated to the RFXtrx433E.

USB Symlink for RFXCOM RFXtrx344E

Create symlink:
1. Do not plug in the RFXtrx433E.
2. List the devices: ls /dev/tty*
3. Plug in the RFXtrx433E 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, USB device information is required: *idVendor*, *idProduct*, *iSerial*.

Get USB device information

```
sudo lsusb -v | grep 'idVendor\|idProduct\|iProduct\|iSerial'
```

Seek the RFXCOM RFXtrx433 entry listed under iProduct, i.e. RFXtrx433

```
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
iProduct                2 RFXtrx433
iSerial                 3 A1YQCEQY
...
```

The data required to define the symlink are the idVendor (0x0403), idProduct (0x6001),
iSerial (A1YQCEQY).

Create the symlink
On the Raspberry Pi, edit the file /etc/udev/rules.d/file 99-usb-serial.rules and reboot!

```
sudo nano /etc/udev/rules.d/99-usb-serial.rules
Add:
SUBSYSTEM=="tty", ATTRS{idVendor}=="0403", ATTRS{idProduct}=="6001",
ATTRS{serial}=="A1YQCEQY", SYMLINK+="ttyUSB-RFX433E-A", MODE="0666"
```

Reboot the Raspberry Pi (`sudo shutdown -r now`) and **check** the USB Port after boot,
check the USB port

```
ls -l /dev/ttyUSB-RFX433E-A
Output:
lrwxrwxrwx 1 root root 7 Aug 29 19:24 /dev/ttyUSB-RFX433E-A -> ttyUSB0
```

Preparation of the RFXtrx433E is completed, next to install & configure the RFXCOM device in Domoticz.

Domoticz

Installation

The Domoticz installation is initiated by running terminal command:

```
curl -L install.domoticz.com | sudo bash
```

Answers to the questions:
Services:
http, Port: 8080.

Domoticz Folder:
/home/pi/domoticz
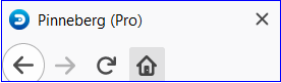
Domoticz URL:
http://rpi-domoticz-ip:8080 or
http://localhost:8080 (when using the Monitor connected to Raspberry Pi)

If everything went ok, enter the Domoticz URL in the Browser and … the Domoticz UI is shown. The Domoticz version is displayed as V4.9700.

Now READY to configure the Domoticz Home Automation Solution further.

Settings

Changes made to the default Domoticz settings:

System	
User Interface	Language: English Theme: Default
Location	Name: Pinneberg (Pro) Latitude: 53.636470 Longitude: 9.798251 *Note*: The Browser shows as tab Pinneberg (Pro)
Dashboard	Mode: Mobile (for access via Smartphone)
Local Networks	Networks: NNN.NNN.1.*
Software Updates	Check Release Channel: beta
Log History	Set to 1 days
Notifications	See under Functions where applicable
Email	
Email enabled	username <email-address>
Send Email notification alerts	Enabled
Meters / Counters	
Wind Meter	Display:Beaufort
Floorplan	See under Functions where applicable
Other	
Event System	Enabled
dzVents	Enabled
	Note: dzVents is the main Event scripting used.

Folders

Folder	Path
Domoticz Home	/home/pi/domoticz
Scripts	/home/pi/domoticz/scripts
Bash scripts -created new folder, make sure all scripts are executable, i.e. sudo chmod +x myscripts.sh	/home/pi/domoticz/scripts/bash

Events

Events are developed using dzVents (Domoticz Easy Events), which is Domoticz Next Generation Lua scripting (described here).

The Domoticz internal event editor is used.

External Lua modules have been defined, to provide common functions across the dzVents scripts.

There are a few exceptions, where an event is written in Python and triggered via crontab. These excepts are not yet converted to Lua (planned).

Functions

Introduction
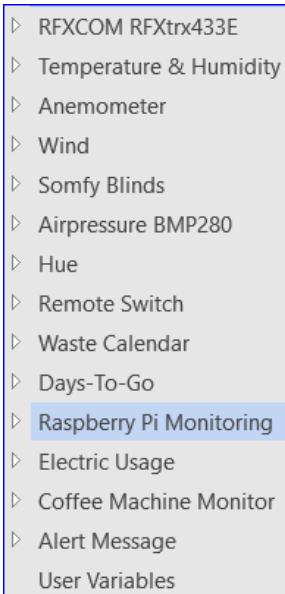
This project has a modular setup, which are called **functions**.

Each **function** has a specific purpose, makes use of required hardware & software and Domoticz configuration settings.

Function examples are
- RFXCOM - wireless receive & transmit between devices/sensors at 433.92Mhz. (Temperature Sensors, Wind Sensors, Light/Switch Devices)
- Philips Hue Light Control (Light/Switch Devices)
- Waste Calendar – inform about waste dates for residual waste, organic waste, paper waste, plastic waste (Utility Sensors)
- Days since / to calculation (Utility Sensors)
- Raspberry Pi system information (Utility Sensors)

Just a few functions to mention:

RFXCOM RFXtrx433E

Purpose

The [RFXCOM RFXtrx433E](#) Transceiver is a 433Mhz transmitter and receiver (transceiver) to control or respond to 433Mhz devices (sensors). More Product information [here](#).

Prepare RFXtrx433E

- Install the utility programs RFXflash and RFXmngr on the Development Device.
- Connect the RFXtrx433E to an USB port of the Development Device
- Flash latest firmware (RFXtrx433_Ext2_Firmware) as described in [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 cmnd   = 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.

Domoticz Configuration

The hardware device is added / updated via **Settings > Hardware**.

4	RFXtrx433e	Yes	RFXCOM - RFXtrx433 USB 433.92MHz Transceiver Version: Ext2/1022 Set Mode	/dev/ttyUSB-RFX433E-A	Disabled

Enabled:	🟢
Name:	RFXtrx433e
Type:	RFXCOM - RFXtrx433 USB 433.92MHz Transceiver
Data Timeout:	Disabled

Specifying a Data Timeout will restart the hardware device if no data is received for the specified time.
Do not enable this option for devices that do not receive data!

Serial Port:	/dev/ttyUSB-RFX433E-A

After adding the device, restart Domoticz (Settings > More Options > Restart) and check the Domoticz Log (Settings > Log), i.e.

```
2018-08-30 11:27:51.659 Status: RFXCOM: Using serial port: /dev/ttyUSB-RFX433E-A
2018-08-30 11:27:52.311 subtype = Interface Response
2018-08-30 11:27:52.311 Sequence nbr = 2
2018-08-30 11:27:52.311 response on cmnd = Get Status
2018-08-30 11:27:52.311 Transceiver type = 433.92MHz
2018-08-30 11:27:52.311 Firmware version = 1022
2018-08-30 11:27:52.311 Firmware type = Ext2
2018-08-30 11:27:52.311 Hardware version = 1.2
2018-08-30 11:27:52.311 Undec off
2018-08-30 11:27:52.311 X10 enabled
2018-08-30 11:27:52.311 ARC enabled
2018-08-30 11:27:52.311 AC enabled
2018-08-30 11:27:52.311 HomeEasy EU enabled
2018-08-30 11:27:52.311 Meiantech/Atlantic enabled
2018-08-30 11:27:52.311 Oregon Scientific enabled
2018-08-30 11:27:52.311 ATI/Cartelectronic enabled
2018-08-30 11:27:52.311 Visonic enabled
2018-08-30 11:27:52.311 Mertik enabled
2018-08-30 11:27:52.311 AD enabled
2018-08-30 11:27:52.311 Hideki enabled
2018-08-30 11:27:52.311 La Crosse enabled
2018-08-30 11:27:52.311 Legrand enabled
2018-08-30 11:27:52.311 MSG4Reserved5 enabled
2018-08-30 11:27:52.311 BlindsT0 enabled
2018-08-30 11:27:52.311 BlindsT1 enabled
2018-08-30 11:27:52.311 AE enabled
2018-08-30 11:27:52.311 RUBiCSON enabled
2018-08-30 11:27:52.311 FineOffset enabled
2018-08-30 11:27:52.311 Lighting4 enabled
2018-08-30 11:27:52.311 Conrad RSL enabled
2018-08-30 11:27:52.311 ByronSX enabled
2018-08-30 11:27:52.311 IMAGINTRONIX enabled
2018-08-30 11:27:52.311 KEELOQ enabled
2018-08-30 11:27:52.311 Home Confort enabled
2018-08-30 11:27:52.402 (RFXtrx433e) Temp + Humidity (Garage)
```

Temperature & Humidity

Purpose

To measure the temperature (°C) & humidity (%RH) in various rooms.

Device

Device (433MHz): Temperature TFA Dostmann / Wertheim TS34C.
The devices are connected to the RFXCOM RFXtrx433E.

Setup

1. Open the device battery flip and set the channel.
   Start with 1.
2. Take out the battery blocker
3. The device will be recognized automatically and listed under the devices list
4. Select the green arrow to give the device the name and the name will be shown in the list (see below)

Device List Entries (idx=14,11)

		Idx	Hardware	ID	Unit	Name	Type	SubType	Data				Last Seen
□		14	RFXCom	450E	2	Temp Outside	Temp + Humidity	Cresta, TFA TS34C	19.0 C, 55 %	4	100	⚙✎▢	2015-06-19 15:55:10
□		11	RFXCom	2A0E	1	Temp Living Room	Temp + Humidity	Cresta, TFA TS34C	17.6 C, 59 %	6	100	⚙✎▢	2015-06-19 15:54:06
□		6	GPIO Port		18	LED GPIO18	Lighting 1	Impuls	Off	7	-	⚙✎▢	2015-06-19 15:20:16
□		12	RFXCom	1F1F1F	1	Blind Living Room	RFY	RFY	Stopped	7	-	⚙✎▢	2015-06-19 15:18:55
□		13	RFXCom	1E1E1E	2	Blind Bed Room	RFY	RFY	Stopped	7	-	⚙✎▢	2015-06-19 15:06:34

Temperature & Humidity Widgets

Temp Living Room	17.7° C / 59%
Comfortable, Dew Point: 9.59° C	
Last Seen: 2015-06-19 16:04:08	
Log Edit Notifications	

Temp Outside	18.7° C / 57%
Comfortable, Dew Point: 10.01° C	
Last Seen: 2015-06-19 16:04:55	
Log Edit Notifications	

Anemometer

Purpose

To measure the wind speed (m/s, bft), direction and air temperature.

Note
Only the wind speed & direction of this device are used and assigned to a Virtual Sensor
Wind (see Function Wind) which is used for the Dashboard.

Device

Device (433MHz): Anemometer **TFA Dostmann 30.3168 Windmeter**
Location: Garden shed ("Gartenhaus").
The anemometer is connected to the RFXCOM RFXtrx433E.

Setup

1. Open the device battery flip and take out the battery blocker
2. The device will be recognized automatically and listed under the devices list
3. Select the green arrow to give the device the name and the name will be shown in the list (see below)

Anemometer Device List Entry (idx=28)

28	RFXtrx433e	9D16	0	Windmesser	Wind	TFA	293.00;WNW;9;9;14.6;14.6

Anemometer Widgets (Tabs Utility & Temperature)

Wind

Purpose

To display wind direction and wind speed obtained from the Anemometer.

Solution

The **Wind Sensor Device (Wind Meter TFA Dostmann 30.3168**) "Windmesser" measures wind direction, wind speed and temperature.
The measured temperature is not used as the temperature is provided by a **Temperature & Humidity Device (TFA Dostmann / Wertheim TS34C)**, which means that (only) the wind direction and wind speed are used and displayed on the dashboard.

The solution is to create a Virtual Sensor Type Wind named "Wind" and assign the wind direction and wind speed from the Wind Sensor.

Wind Virtual Sensor Device List Entry (idx=117)

117	VirtualSensors	140C5	1	Wind	Wind	WTGR800	0;N;0;0;0;0

Weather Widget Wind Virtual Sensor

Note
The data displayed is provided by the Event Lua Script script_device_wind.

Wind Sensor Device with actual data (idx=28)

28	RFXtrx433e	9D16	0	Windmesser	Wind	TFA	304.00;WNW;12;12;21.2;21.2

The data contains: WB,WD,WS,WG,T,TW
- WB = Wind bearing (0-359)
- WD = Wind direction (S, SW, NNW, etc.)
- WS = 10 * Wind speed [m/s]
- WG = 10 * Gust [m/s]
- T = 21.2 = Temperature
- TW = 21.2 = Temperature Windchill

Event dzVents

To update the wind direction and wind speed from the Wind Sensor Device data and assign to the Wind Virtual Sensor.
This is done via a dzVents script using the Domoticz Event Editor.

Event Name	Interpreter	Event Type	Event Active
wind_update.lua	dzVents	Device	Enabled

```
--[[
 wind_update.lua
 If the a value of the device Windmesser (idx=28) changes,
 update the value (Wind direction & speed) of
 the Virtual Sensor Wind named Wind (ix=117)
    Use print(device.dump()) ONLY ONCE to get the properties.
    Project: atHome
    Interpreter: dzVents, Device
    See: athome.pdf
    Author: Robert W.B. Linn
    Version: 20180910
]]--

-- Idx of the devices
local IDX_WINDMESSER = 28;
local IDX_WIND = 117;

-- Event handling changes of the Windmesser device
return {
 on = {
   devices = {
     IDX_WINDMESSER
   }
 },
 execute = function(domoticz, device)
  -- print(device.dump())
  -- domoticz.log('Device ' .. device.name .. ' changed ', domoticz.LOG_INFO)

  bearing = device.direction
  direction = device.directionString
  speed = device.speed
  gust = device.gust
  temperature = device.temperature
  chill = device.chill
  -- Update the virtual sensor Luftdruck
 domoticz.devices(IDX_WIND).updateWind(bearing,direction,speed,gust, temperature,chill)
 end
}
```

Domoticz Log Entry

```
2018-09-10 16:09:12.797 (RFXtrx433e) Wind (Windmesser)
2018-09-10 16:09:12.020 Status: EventSystem: Script event triggered:
/home/pi/domoticz/dzVents/runtime/dzVents.lua
2018-09-10 16:09:12.943 Status: dzVents: Info: Handling events for:
"Windmesser", value: "281.00;W;7;13;21.2;21.2"
2018-09-10 16:09:12.944 Status: dzVents: Info: ------ Start internal script:
wind_update: Device: "Windmesser (RFXtrx433e)", Index: 28
2018-09-10 16:09:12.944 Status: dzVents: Info: Device Windmesser changed
2018-09-10 16:09:12.945 Status: dzVents: Info: ------ Finished wind_update
```

Somfy Blinds

Purpose

To control (open, close, stop) Somfy Blinds (RTS Pure) using the RFXtrx433E Transceiver.

Domoticz Configuration

1. Open Domoticz URL (like http://rpi-domoticz-ip:8080)
2. Tab Switches > Select Manual Switches
3. Select Manual/Light Switch
4. Define the properties (see right)
   NOTE: **Do NOT press Add Device**
5. On the Remote RTS Pure Telis 1:
   Press the Prog Button for 3 seconds.
   The Blind will move short (=react).
6. Click Add Device
7. The new device is added to the Switches
   Note:
   When adding more devices define a new ID and set the Unit Code, i.e. 2.
   The existence of a device is checked by its unique ID + Unit Code
8. The device is listed under Settings > Devices (see below, idx 13)

	Idx	Hardware	ID	Unit	Name	Type	SubType	Data					Last Seen
	14	RFXCom	450E	2	Temp Outside	Temp + Humidity	Cresta, TFA TS34C	19.0 C, 55 %	4	100			2015-06-19 15:55:10
	11	RFXCom	2A0E	1	Temp Living Room	Temp + Humidity	Cresta, TFA TS34C	17.6 C, 59 %	6	100			2015-06-19 15:54:06
	6	GPIO Port		18	LED GPIO18	Lighting 1	Impuls	Off	7	-			2015-06-19 15:20:16
	12	RFXCom	1F1F1F	1	Blind Living Room	RFY	RFY	Stopped	7	-			2015-06-19 15:18:55
	13	RFXCom	1E1E1E	2	Blind Bed Room	RFY	RFY	Stopped	7	-			2015-06-19 15:06:34

Dashboard & Switches Widget

Dashboard
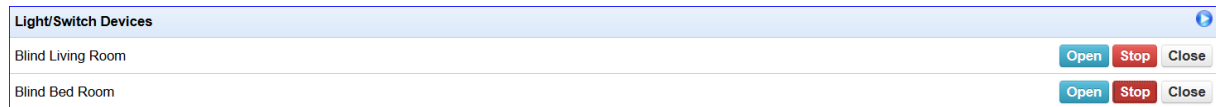
Tab Switches Widgets Blind Living & Bed Room

MQTT Messages

Example MQTT Messages for Topic="Domoticz/out", idx=13, name="Blind Bed Room".

The messages are related to switching states Open > Stop > Close > Stop, which is Payload Property nvalue.

Open	Stop	Close	Open

domoticz/out {
 "Battery" : 255,
 "RSSI" : 12,
 "description" : "",
 "dtype" : "RFY",
 "id" : "1E1E1E",
 "idx" : 13,
 "name" : "Blind Bed Room",
 "nvalue" : 1,
 "stype" : "RFY",
 "switchType" : "Blinds",
 "unit" : 2
}
	domoticz/out {
	 "Battery" : 255,
	 "RSSI" : 12,
	 "description" : "",
	 "dtype" : "RFY",
	 "id" : "1E1E1E",
	 "idx" : 13,
	 "name" : "Blind Bed Room",
	 "nvalue" : 0,
	 "stype" : "RFY",
	 "switchType" : "Blinds",
	 "unit" : 2
	}
		domoticz/out {
		 "Battery" : 255,
		 "RSSI" : 12,
		 "description" : "",
		 "dtype" : "RFY",
		 "id" : "1E1E1E",
		 "idx" : 13,
		 "name" : "Blind Bed Room",
		 "nvalue" : 3,
		 "stype" : "RFY",
		 "switchType" : "Blinds",
		 "unit" : 2
		}
			domoticz/out {
			 "Battery" : 255,
			 "RSSI" : 12,
			 "description" : "",
			 "dtype" : "RFY",
			 "id" : "1E1E1E",
			 "idx" : 13,
			 "name" : "Blind Bed Room",
			 "nvalue" : 0,
			 "stype" : "RFY",
			 "switchType" : "Blinds",
			 "unit" : 2
			}

Airpressure BMP280

Purpose

To measure the Air Pressure using a BMP280 sensor connected to the GPIO of the Raspberry Pi.

Circuit

The BMP280 sensor (Type Sunfounder), is connected to the Raspberry Pi SDA & SCL pins.

BMP180	Raspberry Pi 3B+ (wirecolor)
3.3v	3.3v (red)
SDA	SDA = I2C1 = Pin #3 (yellow)
SCL	SCL = I2C2 = Pin#5 (blue)
GND	GND (black)
I2C Address	76 Use command *i2cdetect -y 1*
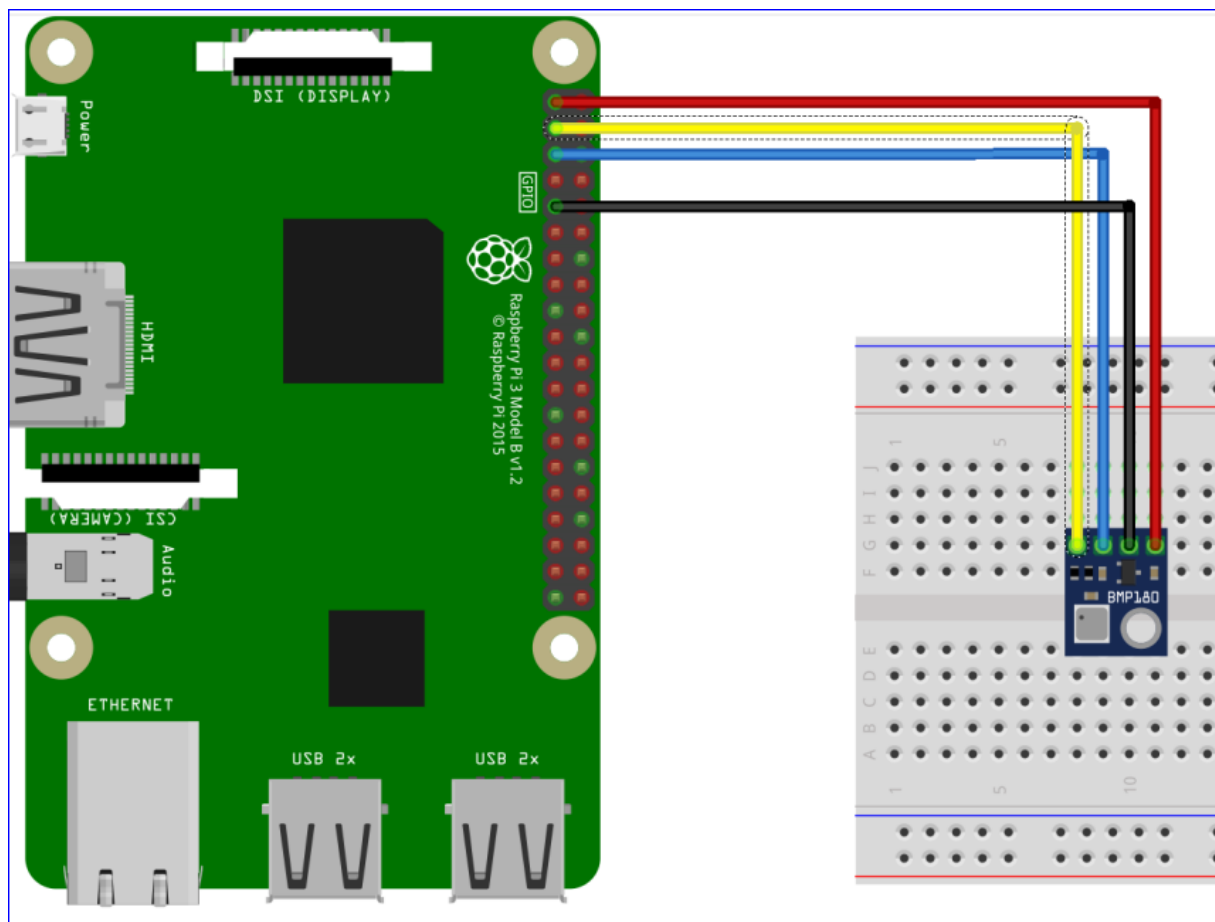
Note

Drawing created with Fritzing. Adding multiple I2C sensors, requires a unique address for each sensor and two 1.k resistors between Vcc and SDA, SCL.

I2C

Check if the I2C module (i2c_bcm2835) is loaded.
If not, activate the I2C interface in the Raspberry Pi preferences.
Below list is only an extract…

```
lsmod
Module                 Size  Used by
i2c_bcm2835           16384  0
spi_bcm2835           16384  0
w1_gpio               16384  0
wire                  40960  1 w1_gpio
i2c_dev               16384  0
ip_tables             24576  0
x_tables              32768  1 ip_tables
ipv6                 434176  58


ls -l /dev/i2c-*
crw-rw---- 1 root i2c 89, 1 Aug 30 12:08 /dev/i2c-1


i2cdetect -l
i2c-1   i2c              bcm2835 I2C adapter                    I2C adapter
i2cdetect -y 1
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:          -- -- -- -- -- -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- 76 --
```

Hardware & Device

(Setup > Hardware)

Enabled:	[green toggle]
Name:	BMP280
Type:	I2C sensors
Data Timeout:	Disabled
	Specifying a Data Timeout will restart the hardware device if no data is received for the specified time. **Do not enable this option for devices that do not receive data!**
SubType:	I2C sensor BME280 Temp+Hum+Baro
Path to I2C bus:	/dev/i2c-1 (etc. /dev/i2c-1, or empty for autodetection)

Note:
The path to the I2C Bus is taken from previous terminal command:

```
ls -l /dev/i2c-*
crw-rw---- 1 root i2c 89, 1 Aug 30 12:08 /dev/i2c-1
```

Check if the hardware has been added to the list

Idx	Name	Enabled	Type	Address	Port
13	BMP280	Yes	I2C sensor BME280 Temp+Hum+Baro	I2C	/dev/i2c-1

Check the log entry after adding the hardware

```
2018-08-31 11:34:50.758 Status: I2C Start HW witf ID: 13 Name: I2C_BME280 Address:
0 Port: /dev/i2c-1 Invert:0
2018-08-31 11:34:50.758 Status: I2C_BME280: Worker started...
```

Add the BMP280 Device

Setup > Settings > Hardware/ Devices > Allow new Devices 5 minutes

A new device is added with idx=115 (Setup > Devices)

115	BMP280	0001	1	TempHumBaro	Temp + Humidity + Baro	Weather Station	20.2 C, 0 %, 1021.0 hPa

Add the device, named BMP280 and check the widgets.

Weather Widget	**Temperature Widget**
BMP280 1021 hPa **Prediction: Sunny** Last Seen: 2018-08-31 11:42:51 Log Edit Notifications	**BMP280** 19.7° C / 0% Dry, Barometer: 1027.8 hPa, Prediction: Sunny **Dew Point: 19.70° C** Last Seen: 2018-08-31 16:00:48 Log Edit Notifications

Barometer Widget

The BMP280 measures pressure, temperature and humidity.
The temperature and the humidity are not of interest because these are measured at the
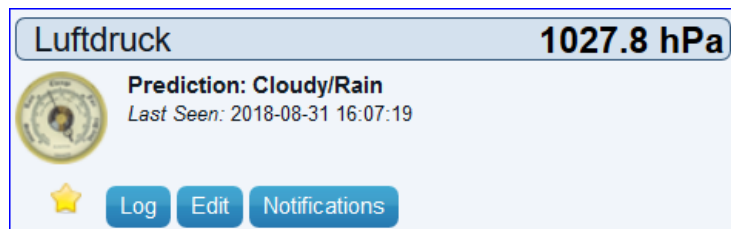Raspberry Pi location and not outside.
In the Dashboard would like to have a Barometer Widget with value taken from the BMP280.

The solution is to create a Virtual Sensor Type Barometer named "Luftdruck".

Luftdruck Virtual Sensor Device List Entry (idx=116)

116	VirtualSensors	00082116	1	Luftdruck	General	Barometer	1027.6 hPa

Weather Widget Luftdruck

Note
Default (=wrong) data as the event to obtain date is developed next.

Barometer Event (dzVents)

Created an event to update the pressure and pressure forecast from the BMP280 data.
This is done via a Lua script using the Domoticz Event Editor.

115	BMP280	0001	1	BMP280	Temp + Humidity + Baro	Weather Station	19.5 C, 0 %, 1028.0 hPa

- TEMP = Temperature
- HUM = Humidity
- BAR = Barometric pressure

Event Editor

Event properties: luftdruck_update, dzEvents, Device

dzVents Script

```
--[[
 luftdruck_update.lua
 If the a value of the BMP280 device (idx=115) has changed,
 update the value of the Virtual Sensor Barometer named Luftdruck (idx=116)
 Use print(device.dump()) ONLY ONCE to get the properties.
 Project: atHome
 Interpreter: dzVents, Device
 See: athome.pdf
 Author: Robert W.B. Linn
 Version: 20180911
]]--

-- Idx of the devices
local IDX_BMP280 = 115;
local IDX_LUFTDRUCK = 116;

-- Event handling changes of the BMP280 device
return {
  on = {
    devices = {
      IDX_BMP280
    }
  },
  execute = function(domoticz, device)
    -- print(device.dump())
    -- domoticz.log('Device ' .. device.name .. ' changed ', domoticz.LOG_INFO)
    domoticz.log('Barometer: ' .. device.barometer .. '/' .. device.forecast .. '/' ..
device.forecastString, domoticz.LOG_INFO)

    -- Round the pressure
    pressure = math.floor(device.barometer)

    -- Forecast from the BMP device to the Barometer device
    -- See source code RFXNames.cpp, BMP_Forecast_Desc
    -- Map the BMP_Forecase_Desc string to dzEvents Forecast value:
    -- domoticz.BARO_STABLE, BARO_SUNNY, BARO_CLOUDY, BARO_UNSTABLE, BARO_THUNDERSTORM

    forecast = -1
    if (device.forecastString == 'Stable') then forecast = domoticz.BARO_STABLE end
    if (device.forecastString == 'Sunny') then forecast = domoticz.BARO_SUNNY end
    if (device.forecastString == 'Cloudy') then forecast = domoticz.BARO_CLOUDY end
    if (device.forecastString == 'Unstable') then forecast = domoticz.BARO_UNSTABLE end
    if (device.forecastString == 'Thunderstorm') then forecast = domoticz.BARO_THUNDERSTORM
end
    if (device.forecastString == 'Unknown') then forecast = domoticz.BARO_STABLE end
    if (device.forecastString == 'Cloudy/Rain') then forecast = domoticz.BARO_CLOUDY end

    -- Update the virtual sensor Luftdruck
    domoticz.devices(IDX_LUFTDRUCK).updateBarometer(pressure, forecast)
  end
}
```

Domoticz Log Entry

```
2018-09-11 13:51:36.606 Status: dzVents: Info: Handling events for: "BMP280",
value: "19.9;0;2;1024.0;4"
2018-09-11 13:51:36.606 Status: dzVents: Info: ------ Start internal script:
luftdruck_update: Device: "BMP280 (BMP280)", Index: 115
2018-09-11 13:51:36.606 Status: dzVents: Info: Barometer: 1024/4/Cloudy
2018-09-11 13:51:36.607 Status: dzVents: Info: ------ Finished luftdruck_update
```

Hue

Purpose

To control devices connected to the Philips Hue Bridge.

Hue Bridge

There is one Hue Bridge (Hardware idx=12) connected to the Router.

Note
Initially Groups & Scenes are not used.

Hue Devices

The list of Hue Devices, which are automatically created by Domoticz (Setup > Settings > Hardware allow for 5 minutes), in order of their idx:

Idx	Hardware	ID	Unit	Name	Type	SubType
110	Philips Hue Bridge 1	0000001	1	Hue Haustür	Lighting 2	AC
111	Philips Hue Bridge 1	00000002	1	Hue WZ Ablage	Color Switch	WW
112	Philips Hue Bridge 1	0000003	1	Hue WZ TV	Lighting 2	AC
113	Philips Hue Bridge 1	0000004	1	Hue WZ Robs Ecke	Lighting 2	AC
118	Philips Hue Bridge 1	0000005	1	Hue MakeLab	Lighting 2	AC

Hue Lamp Add New

Example adding a Hue white lamp.

First the new Hue Lamp is added to the Hue Bridge. This is done by using the Hue App (Android).
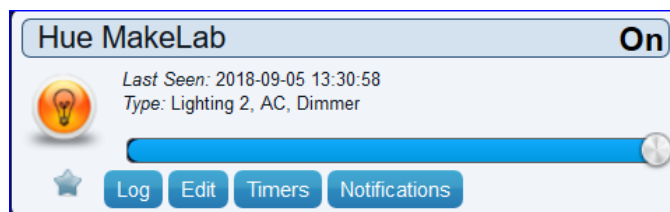After the Hue Lamp has been added to the Hue Bridge, Domoticz adds the device (idx=118) to the list (after Setup > Settings > Hardware allow for 5 minutes).

118	Philips Hue Bridge 1	0000005	1	Hue white lamp 4	Lighting 2	AC

Domoticz Log Entry – Listen to new devices and add the new Hue white lamp 4.

```
2018-09-05 13:22:14.995 Status: New sensors allowed for 5 minutes...
2018-09-05 13:25:41.749 (Philips Hue Bridge 1) Lighting 2 (Hue white lamp 4)
```

Add the device with name Hue MakeLab, set the level to 100%.

Hue MakeLab **On**

Last Seen: 2018-09-05 13:30:58
Type: Lighting 2, AC, Dimmer

Log Edit Timers Notifications

118	Philips Hue Bridge 1	0000005	1	Hue MakeLab	Lighting 2	AC	Set Level: 100 %

Domoticz Log Entry – Hue Lamp Level Change

If the level of the Hue Lamp is changed, the Domoticz Log Entry:

```
2018-09-05 13:37:55.133 Status: User: Admin initiated a switch command (118/Hue
MakeLab/Set Level)
```

Hue Control Tests

Lets start playing around with dzVents and a Hue Lamp Hue MakeLab (idx=118). The Hue Lamp can be handled like a dimmer switch, with methods to be used like device.dimTo(pct), switchOff(), switchOn().
See this documentation, chapter Switch.

To get an overview of the object properties, use device.dump() first.

```
secondsSinceMidnight: 49075        updateSoilMoisture()        unit: 1
ruleIsBeforeCivilTwilightStart()   bState: false               _state: Off
daysAgo: 0                         isTimer: false              hardwareType: Philips Hue
updateBarometer()                  usedByCamera: false         Bridge
setState()                         unit: 1                     maxDimLevel: 15
kodiSwitchOff()                    update()                    icon: dimmer
description:                       armAway()                   levelVal: 5
updateMode()                       setNightMode()              usedByCamera: false
hardwareType: Philips Hue Bridge   setDiscoMode()              protected: false
updateRadiation()                  setWhiteMode()              _nValue: 0
updateWind()                       baseType: device            hardwareName: Philips Hue
dump()                             setKelvin()                 Bridge 1
updateElectricity()                kodiSetVolume()             hardwareID: 12
updateWaterflow()                  updateP1()                  id: 118
updateYouless()                    hardwareID: 12              switchType: Dimmer
updateVoltage()                    updateAirQuality()          baseType: device
updateVisibility()                 isSecurity: false           signalLevel: 12
stop()                             play()                      name: Hue MakeLab
playFavorites()                    deviceSubType: AC           switchTypeValue: 7
updateUV()                         hardwareTypeValue: 38       deviceID: 0000005
updatePercentage()                 levelVal: 5                 lastLevel: 33
isGroup: false                     kodiStop()                  changed: true
updateCustomSensor()               lastLevel: 33               batteryLevel: 255
deviceId: 0000005                  updateCounter()             deviceType: Lighting 2
changed: true                      setVolume()                 rawData:
updateText()                       cancelQueuedCommands()      1: 15
updateLux()                        updatePressure()            subType: AC
deviceType: Lighting 2             active: false               timedOut: false
updateTempHumBaro()                _data:                      id: 118
isScene: false                     lastUpdate: 2018-09-05      rawData:
isHTTPResponse: false              13:37:55                    1: 15
updateSetPoint()                   description:                isVariable: false
updateTempHum()                    data:                       _nValue: 0
                                   hardwareTypeValue: 38
```

Script 1 – Device Trigger

If the Hue Lamp is switched ON, then switch it OFF again.

dzEvents Script

```
local IDX_HUE_MAKELAB = 118;

return {
  on = {
    devices = {
      IDX_HUE_MAKELAB
    }
  },
  execute = function(domoticz, device)
    domoticz.log('Device ' .. device.name .. ' changed to ' .. device.state,
domoticz.LOG_INFO)
    if (device.state == 'On') then
        domoticz.log('Switching OFF', domoticz.LOG_INFO)
        device.switchOff()
        domoticz.log('OK', domoticz.LOG_INFO)
    end
    -- domoticz.log(device.dump(), domoticz.LOG_INFO)
  end
}
```

Script 2 – Timer Trigger

Switch the Hue Lamp ON at 'HH:MM' and switch if OFF again one minute later.
This type of script could be used, to switch
- the Hue Lamps in the Living Room, ON 'at sunset' and OFF 'at 23:00'.
- The Hue Lamp Outside ON 'at sunset' and OFF 'at sunrise'.

dzEvents Script

```
-- Idx of the devices
local IDX_HUE_MAKELAB = 118;

return {
    active = true,
    on = {
        timer = {
          'at 15:49',
          'at 15:50'
        }
    },
    execute = function(domoticz, timer)
        if (timer.trigger == 'at 15:49') then
          domoticz.devices(IDX_HUE_MAKELAB).switchOn()
        end

        if (timer.trigger == 'at 15:50') then
            domoticz.devices(IDX_HUE_MAKELAB).switchOff()
        end
    end
}
```

Hue Timed Switch Lamps Living Room

Timed switch the Hue Lamps in the Living Room ('WZ'), ON 'at sunset' and OFF 'at 23:00'. Update the Info Message.

dzEvents Script

```lua
local IDX_HUE_WZABLAGE = 111
local IDX_HUE_WZTV = 112
local IDX_CONTROLMSG = 52

local function isnow(domoticz)
  return domoticz.time.rawDate .. ' ' .. domoticz.time.rawTime
end

local function isnowtime(domoticz)
  return domoticz.time.rawTime
end

return {
 active = true,
 on = {
       timer = {
         'at sunset',
         'at 22:30'
        }
 },
 execute = function(domoticz, timer)
   if (timer.trigger == 'at sunset') then
     domoticz.devices(IDX_HUE_WZABLAGE).switchOn()
     domoticz.devices(IDX_HUE_WZTV).switchOn()
     domoticz.devices(IDX_CONTROLMSG).updateText('Hue Lamps WZ turned ON ' ..
isnowtime(domoticz))
   end

   if (timer.trigger == 'at 22:30') then
     domoticz.devices(IDX_HUE_WZABLAGE).switchOff()
     domoticz.devices(IDX_HUE_WZTV).switchOff()
     domoticz.devices(IDX_CONTROLMSG).updateText('Hue Lamps WZ turned OFF ' ..
isnowtime(domoticz))
   end
 end
}
```

Domoticz Log Entry

```
2018-09-05 20:02:00.394 (Philips Hue Bridge 1) Color Switch (Hue WZ Ablage)
2018-09-05 20:02:00.417 (Philips Hue Bridge 1) Lighting 2 (Hue WZ TV)
2018-09-05 20:02:00.259 Status: dzVents: Info: ------ Start internal script:
hue_wz_timer:, trigger: at sunset
2018-09-05 20:02:00.276 Status: dzVents: Info: ------ Finished hue_wz_timer
2018-09-05 20:02:00.357 Status: EventSystem: Script event triggered:
/home/pi/domoticz/dzVents/runtime/dzVents.lua
```

Remote Switch

Purpose

To remote switch devices, i.e. lamps etc.

Device

The remote switch used is a Mumbai Wireless Switch Module.
The remote switch is connected to the RFXCOM RFXtrx433E.

Setup

1. Select Switches > Manual Light Switch
2. Add the Properties > See Picture right
3. Notes: On the Remote Control of the Switch the Type is displayed (AB440R which is compatible with AB400D)
4. To determine the House Code and the Unit Code, open the Switch and note the state of the dip switches and lookup the corresponding RFXCom values (see below)
5. Press Add Device
6. Select Setup > Devices
7. Select Switch Lab, click green V to add

Remote Switch Widget

Waste Calendar

Purpose

The Purpose 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.

A room plan "Termine" is defined (Tab Setup > More Options > Plans > Roomplan).

Example Room Termine

Devices Configuration

The devices are created as Virtual Sensors using the Dummy Hardware.

98	VirtualSensors	00082098	1	Biotonne Pi	General	Text	06-09-2018
99	VirtualSensors	00082099	1	Papier Pi	General	Text	21-09-2018
100	VirtualSensors	00082100	1	Restmüll Pi	General	Text	30-08-2018
101	VirtualSensors	00082101	1	Gelber Sack Pi	General	Text	30-08-2018
102	VirtualSensors	00082102	1	Schadstoff Pi	General	Text	08-09-2018
103	VirtualSensors	00082103	1	Biotonne Ho	General	Text	27-08-2018
104	VirtualSensors	00082104	1	Papier Ho	General	Text	17-09-2018
105	VirtualSensors	00082105	1	Restmüll Ho	General	Text	27-08-2018
106	VirtualSensors	00082106	1	Gelber Sack Ho	General	Text	28-08-2018

Concept

A dzVents Lua script, "waste_calendar_update" executed dzVents time trigger reads a CSV file, which contains per line the Device Idx, Interval, waste dates.
The device is updated based upon the difference in days between now and the waste date within a given interval (i.e. 16 or 32 days).
The script also notifies via email (receipients defined in the Domoticz Settings Email and Notification) if the waste target date is 1 days from actual date.

CSV Input File

The CSV file is read line by line, using per line, the Device Index idx (field 1) to update 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: idx, interval, date1, dateN

idx	NN 98, 99…
interval	Number of days between waste dates. Note that intervals might not be exact with 14 days or a month. To go for sure,use 16 instead of 14 16, 31
date	Format Day-Month-Year dd-mm-YYYY 09-08-2018

Example CSV file line entry
The example shows for idx 98 ("Biotonne Pi"), an interval every 16 days and number of dates for the remaining year 2018 and full year 2019.

```
98,16,13-12-2018,28-12-2018,10-01-2019,24-01-2019,07-02-2019,21-02-2019,07-03-
2019,21-03-2019,04-04-2019,18-04-2019,03-05-2019,16-05-2019,31-05-2019,14-06-
2019,27-06-2019,11-07-2019,25-07-2019,08-08-2019,22-08-2019,05-09-2019,19-09-
2019,04-10-2019,17-10-2019,01-11-2019,14-11-2019,28-11-2019,12-12-2019,27-12-
2019,09-01-2020
```

CSV File name & location

The CSV file wastecal.csv, is located in folder /home/pi/domoticz/scripts/dzVents/scripts.

The file is used by a dzVents Lua script waste_calendar_update created using the Domoticz Script editor.

dzVents Lua Script

A dzVents Lua script is time triggered once per day at 00:30. See source for details.

```
--[[
    waste_calendar_update.lua
    Read a csv file with wastecalendar dates per device.
    Update the device text with the date if the date is within interval specified.
    The wastecalender file "wastecal.csv" is located in the dzVents scripts folder.
    Line entry csv file: idx,interval,date1,dateN.
    Project: atHome
    Interpreter: dzVents, Timer
    See: athome.pdf
    Author: Robert W.B. Linn
    Version: 20180930
]]--

-- External modules: /home/pi/domoticz/scripts/dzVents/scripts
local utils = require('utils')
local msgbox = require('msgbox')

-- threshold in days to notify
-- if the difference of the current day to a wastecalendarday equals threshold
-- then notify
local TH_WASTECAL = 1

-- Helpers
function string:split( inSplitPattern, outResults )
  if not outResults then
    outResults = { }
  end
  local theStart = 1
  local theSplitStart, theSplitEnd = string.find( self, inSplitPattern, theStart )
  while theSplitStart do
    table.insert( outResults, string.sub( self, theStart, theSplitStart-1 ) )
    theStart = theSplitEnd + 1
    theSplitStart, theSplitEnd = string.find( self, inSplitPattern, theStart )
  end
  table.insert( outResults, string.sub( self, theStart ) )
  return outResults
end

-- the wastecalendar file is updated once per year
local wastecalfile = '/home/pi/domoticz/scripts/dzVents/scripts/wastecal.csv'
local sep = ","

-- read the csv file
function readandupdate(domoticz)
    -- read the file line by line.
    -- each line contains an array with device[1], wasteinterval[2], dates[2...]
    for line in io.lines(wastecalfile) do
      -- split the line by , and assign to a table
      local wastetable = line:split(sep)
      local idx = tonumber(wastetable[1])
      local interval = tonumber(wastetable[2])
      -- flag to check if a wastedate has been found
      local datefound = 0

      -- check the dates, start at 3 because 1=idx, 2=interval and number of table entries
      for i = 3, #wastetable do

            -- get the wastedate array entry
        local wastedate = wastetable[i]
```

```lua
        -- split the wastedate into an array with 3 entries day, month, year
        local wastedatesplit = wastedate:split("-")
        -- calculate the daysdiff between now and the wastedate
        local daysdiff = utils.datediffnow(wastedatesplit[1], wastedatesplit[2], wastedatesplit[3])
        --
        local month = tonumber(wastedatesplit[2])

        -- check if daysdiff within interval
        -- if daysdiff >= 0 and daysdiff < interval then
        if daysdiff >= 0 and datefound == 0 then
                datefound = 1
          print (domoticz.devices(idx).name .. '(' .. idx .. ')=' .. wastedate .. ', days=' .. daysdiff
.. ', interval=' .. interval)

          -- update the device text with the new date
                domoticz.devices(idx).updateText(wastedate)

          -- check number of days left compared to the threshold
          -- i.e. if 1 day left, then notify using device name & text.
          -- the device text contains the date
          -- NOT USED = handled by the dzVents script waste_calendar_notifier
          if daysdiff == TH_WASTECAL then
            -- print ('Device ' .. idx .. ' notifying...')
                  message = '[ACHTUNG] ' .. domoticz.devices(idx).name .. ' ' ..
domoticz.devices(idx).text
                      -- update alert message
                      -- msgbox.alertmsg(domoticz, domoticz.ALERTLEVEL_RED, message)
                  domoticz.log(message, domoticz.LOG_INFO)
                      domoticz.notify(message)
           end
        end

          -- check if for the last entry daysdiff less 0, then no new date available
          -- this is used f.e. for hohenfelde as last wastedate is sep or oct
        if i == #wastetable and daysdiff < 0 then
                datefound = 1
          print (idx .. '=' .. wastedate .. ', days=' .. daysdiff .. ', interval=' .. interval .. ',
KEIN TERMIN')

          -- update the device text with the info that there is no date (yet) available
                domoticz.devices(idx).updateText('Kein neuer Termin')
        end

         end

      -- check if a wastedate has been found, if not then set Kein termin message
      if datefound == 0 then
        print (idx .. '=' .. 'KEIN TERMIN')
     -- update the device text with the info that there is no date (yet) available
            domoticz.devices(idx).updateText('Kein neuer Termin')
        end

        -- JUST SOME LEARNERS
     -- print the content
     -- for i = 1, #wastetable do
     --         print( wastetable[i] )
     -- end
    end
end

return {
  on = {
        timer = {
            -- for tests use every minute
            -- 'every minute',
           'at 00:30'
        }
    },
  execute = function(domoticz)
        readandupdate(domoticz)
  end
}
```

Python CSV File Generation Script

A Python 2 script is used to create the wastecal.csv file.
The script reads a iCal file.

Script

```
# -*- coding: cp1252 -*-
## File:    wastecalicstocsv.py (Python 2)
## Project: atHome
## Purpuse: Create a CSV file with waste dates from an ICS iCal file Pinneberg and
empty entries for Hohenfelde
## Author:  Robert W.B. Linn
## Version: 20181209


## Notes:
## The generated csv file wastecal.csv must be placed in
/home/pi/domoticz/scripts/dzVents/scripts
## Domoticz Device Idx and Interval
## Biotonne Pi=98,16
## Papier Pi=99,32
## Restmuell Pi=100,16
## Gelber Sack Pi=101,16
## Schadstoff Pi=102,32
## Biotonne Ho=103,16
## Papier Ho =104,32
## Restmuell Ho=105,16
## Gelber Sack Ho=106,16


## The generated file has to be merged with the wastecal.csv file located on the
Domoticz server:
## /home/pi/domoticz/scripts/dzVents/scripts

import csv, io, sys
import datetime
from datetime import date

wastecalendarics = 'wastecal-pi-2019.ics'

# Define the wastecalender csv file located in the same folder as the python
script.
# CSV Format: wastetype, date1, date2 ...
# Date format dd-mm-YYYY, i.e. 09-08-2018
wastecalendarcsv = 'wastecal-2019.csv'

# Check the python version to set the file open arguments
if sys.version_info[0] == 2:  # Not named on 2.6
    icsaccess = 'rb'
    icskwargs = {}
    csvaccess = 'wb'
    csvkwargs = {}
else:
    icsaccess = 'rt'
    icskwargs = {'newline':''}
    csvaccess = 'a'
    csvkwargs = {}
    #csvkwargs = {'newline':''}
```

```
# Define the ics tokens
TOKEN_EVENT_BEGIN = "BEGIN:VEVENT"
TOKEN_EVENT_END = "END:VEVENT"
TOKEN_EVENT_DESCRIPTION = "DESCRIPTION:"
TOKEN_EVENT_START = "DTSTART:"
TOKEN_REST = "Restabfall 2"
TOKEN_BIO = "Bio"
TOKEN_SCHAD = "Schad"
TOKEN_GELB = "Gelb"
TOKEN_PAPIER = "Papier"

# Define the lists holding the dates per waste types. The lists are used to create
the csv file
pirestlist = []
pibiolist = []
pischadlist = []
pigelblist = []
pipapierlist = []
horestlist = []
hobiolist = []
hogelblist = []
hopapierlist = []

# Define helpers
eventfound = 0
eventdescription = ""
eventstart = 0
eventyear = 0
eventmonth = 0
eventday = 0

def csv2string(data):
    si = io.BytesIO()
    cw = csv.writer(si)
    cw.writerow(data)
    return si.getvalue().strip('\r\n')

# Get the actual year month day
s = datetime.date.today().strftime("%d-%m-%Y")
# Get the date now in format i.e. 09-08-2018
datenow = datetime.datetime.strptime(s, "%d-%m-%Y")

# Set the waste type as the first entry in the waste lists
## Biotonne Pi=98,16
## Papier Pi=99,32
## Restmuell Pi=100,16
## Gelber Sack Pi=101,16
## Schadstoff Pi=102,32
## Biotonne Ho=103,16
## Papier Ho =104,32
## Restmuell Ho=105,16
## Gelber Sack Ho=106,16

pibiolist.append('98,16')
pipapierlist.append('99,32')
pirestlist.append('100,16')
pigelblist.append('101,16')
pischadlist.append('102,32')
hobiolist.append('103,16')
hopapierlist.append('104,32')
```

```python
horestlist.append('105,16')
hogelblist.append('106,16')

# Open the wastecalendar ics file in read access
with open(wastecalendarics, icsaccess, **icskwargs) as icsfile:
    # lines = icsfile.readlines()
    # Loop thru the file line by line
    for line in icsfile:
        # Strip the CRLF from the line
        linet = line.strip()
        # Print the line for tests
        # print linet
        # Check the entries

        # If token event end is found and an event has been build then add a waste
entry date to the list.
        if linet.startswith(TOKEN_EVENT_END):
            if eventfound == 1 and len(eventdescription)> 0 and eventstart > 0:
                wastedate = "%s-%s-%s" % (eventday,eventmonth,eventyear)
                # print "Adding the line ", eventdescription, eventstart,
eventyear, eventmonth, eventday

                # Workout the waste types and add the date
                if eventdescription.startswith(TOKEN_REST):
                    pirestlist.append(wastedate)
                    # print "Adding ", eventdescription, "-", wastedate

                if eventdescription.startswith(TOKEN_BIO):
                    pibiolist.append(wastedate)

                if eventdescription.startswith(TOKEN_PAPIER):
                    pipapierlist.append(wastedate)

                if eventdescription.startswith(TOKEN_GELB):
                    pigelblist.append(wastedate)

                if eventdescription.startswith(TOKEN_SCHAD):
                    pischadlist.append(wastedate)

                # Reset the globals which are used for the next event
                eventdescription = ""
                eventstart = 0
                eventyear = 0
                eventmonth = 0
                eventday = 0
                eventfound = 0

        # Set the flag if an new event is found
        if linet.startswith(TOKEN_EVENT_BEGIN):
            eventfound = 1

        # Event was found and now the description
        if eventfound and linet.startswith(TOKEN_EVENT_DESCRIPTION):
            eventdescription = linet.replace(TOKEN_EVENT_DESCRIPTION, "")

        # Event was found and now the start date
        # Extract from the date, the year, month, day
        if eventfound and linet.startswith(TOKEN_EVENT_START):
            # Example = DTSTART:20180105T053000Z
            eventstartstr = linet.replace(TOKEN_EVENT_START, "")
```

```
  # Get the date in format 20180105
        index = eventstartstr.index('T')
        eventstartstr = eventstartstr[0:index]
        eventstart = int(eventstartstr)
 # Get the year, month, day
        eventyear = eventstartstr[0:4]
        eventmonth = eventstartstr[4:6]
        eventday = eventstartstr[6:8]
        # print eventyear, eventmonth, eventday

# Example to show the content of a list
#print len(biolist)
#for wastedate in biolist:
#    print wastedate
#print csv2string(biolist)

# Write the lists to the CSV file. Each row contains a waste type with dates.
# Sample: 98,16,12-01-2018,09-02-2018, ...
with open(wastecalendarcsv, csvaccess, **csvkwargs) as csvfile:
    # Pinneberg
    csvfile.write(",".join(pipapierlist))
    csvfile.write("\n")
    csvfile.write(",".join(pirestlist))
    csvfile.write("\n")
    csvfile.write(",".join(pibiolist))
    csvfile.write("\n")
    csvfile.write(",".join(pigelblist))
    csvfile.write("\n")
    csvfile.write(",".join(pischadlist))
    csvfile.write("\n")
    # Hohenfelde - the dates needs tobe added manually
    csvfile.write(",".join(hopapierlist))
    csvfile.write("\n")
    csvfile.write(",".join(horestlist))
    csvfile.write("\n")
    csvfile.write(",".join(hobiolist))
    csvfile.write("\n")
    csvfile.write(",".join(hogelblist))
    csvfile.write("\n")
    # Close with message
    csvfile.close()
    print "Created CSV file", wastecalendarcsv

# Exit the script
icsfile.close()
sys.exit()
```
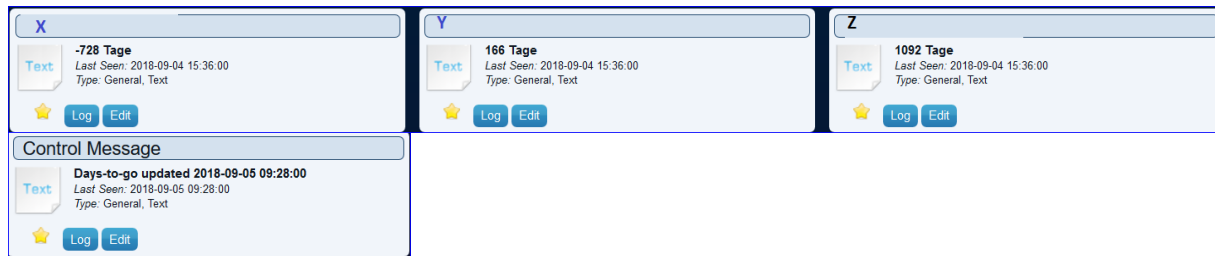
Days-To-Go

Purpose

To calculate, once a day around midnight, the days-to-go for several Virtual Devices.
Beside days-to-go, days-left are also calculated depending date given.
Update the Control Message after updating the devices.

Widget Sample

Note
Changed the name and the days calculated use different dates then the script below.

Solution

Run a **dzVents script**, once a day at 00:15, to calculate the date difference in days and update the devices.

The dzVents script is written using the Domoticz Event Editor (Setup > More Options > Events).

Event Properties

Event name: days_to_go_update.lua
Interpreter: dzVents
Event type: Timer
Event active

Event Script dzVents

(code stripped)

```lua
-- Idx of the devices
local IDXDAYSX = 29;
local IDXDAYSY = 57;
local IDXDAYSZ = 97;
local IDXCONTROLMSG = 52;

-- Calculate the date difference in days between now and the target date
-- Return days
local function datediffnow(d,m,y)
  local targetdate = os.time{day=d, month=m, year=y}
  local daysdiff = os.difftime(targetdate, os.time()) / (24 * 60 * 60)
  return math.floor(daysdiff)
end

-- Get the current date & time from the domoticz instance with time object
-- Return date & time now, i.e. 2018-09-05 09:09:00
local function isnow(domoticz)
  return domoticz.time.rawDate .. ' ' .. domoticz.time.rawTime
end

return {
    on = {
        timer = {
            -- 'every minute',
            'at 00:15'
        }
    },
    execute = function(domoticz, timer)
    domoticz.devices(IDXDAYSX).updateText(datediffnow(1,9,2021) .. ' Tage')
    domoticz.devices(IDXDAYSY).updateText(datediffnow(1,9,2016) .. ' Tage')
    domoticz.devices(IDXDAYSZ).updateText(datediffnow(1,9,2019) .. ' Tage')
    domoticz.devices(IDXCONTROLMSG).updateText('Days-to-go updated ' ..
isnow(domoticz))

end
}
```

Domoticz Log Entry

Script saved

```
2018-09-04 14:44:35.496 Status: EventSystem: reset all events...
2018-09-04 14:44:35.497 Status: dzVents: Write file:
/home/pi/domoticz/scripts/dzVents/generated_scripts/days-to-go-update.lua
```

Script running
(for tests running every minute instead of once per day)

```
2018-09-04 15:36:00.068 Status: dzVents: Info: ------ Start internal script: days-
to-go-update:, trigger: every minute
2018-09-04 15:36:00.085 Status: dzVents: Info: ------ Finished days-to-go-update
```

Addition

A script addition is made to calculate an age in xx Years yy Days.

```
ageyearsdays(7,9,2016)
```

Lua Functions

```
-- Calculate the date difference in days between now and the target date
-- Return days
local function datediffnow(d,m,y)
  -- Set the target date from the parameter
  local targetdate = os.time{day=d, month=m, year=y}

  -- Get the time diff between now and the target date in seconds in a day
  local daysdiff = os.difftime(targetdate, os.time()) / (24 * 60 * 60)

  -- Return the days
  return math.floor(daysdiff)
end

-- Calculate the age in years + days
-- Return age as string years J days T
local function ageyearsdays(dbirth,mbirth,ybirth)
  -- get the actual date
  t = os.date ("*t")

  -- get the year, month, day for now
  ynow = t.year
  mnow = t.month
  dnow = t.day

  -- year difference between now year and the target year
  ydiff = ynow - ybirth
  -- days difference between now and the target day+month for the now year
  -- i.e. dtarget+mtarget+ynow
  ddiff = math.abs(datediffnow(dbirth, mbirth, ynow))

  -- build the age string to return
  age = ydiff .. ' J ' .. ddiff .. ' T'
  -- print(age)
  return age
end
```

Raspberry Pi Monitoring

Purpose

To measure & show specific Raspberry Pi system information and trigger alarm if a value is above a certain threshold.
Monitored are: Memory Usage, Hard Disc Space, Temperature.

Solution

Use the Domoticz Hardware Motherboard, which will create several devices. The devices can be used to trigger events.

Hardware Motherboard

1	Motherboard	Yes	Motherboard sensors

Showing 1 to 8 of 8 entries

Update   Delete

Enabled: [ON]
Name: Motherboard
Type: Motherboard sensors
Data Timeout: Disabled
Specifying a Data Timeout will restart the hardware device if no data is received for the specified time.
Do not enable this option for devices that do not receive data!

Devices derived from Hardware Motherboard

		Idx ▲	Hardware ▲	ID ▲	Unit ▲	Name ▲	Type ▲	SubType ▲	
☐	🖼	1	Motherboard	0000044C	1	RPi Memory Usage	General	Percentage	20.93%
☐	🖼	2	Motherboard	0000044E	1	HDD /boot	General	Percentage	1.5%
☐	🖼	3	Motherboard	0000044F	1	RPi HDD	General	Percentage	32.32%
☐	🌡	4	Motherboard	0001	1	RPiTemp	Temp	LaCrosse TX3	44.0 C
☐	🖼	5	Motherboard	0000044D	1	CPU_Usage	General	Percentage	2.84%

Note
Not all devices are added, (only) RPi Memory Usage, RPi HDD, RPi Temp.

Widgets (Tabs Utility & Temperature)

RPi Memory Usage	20.84%
%	Last Seen: 2018-09-04 16:44:29 Type: General, Percentage
☆ Log Edit Notifications	

RPi HDD	32.32%
%	Last Seen: 2018-09-04 16:44:39 Type: General, Percentage
☆ Log Edit Notifications	

RPiTemp	43.5° C
	Last Seen: 2018-09-04 16:46:29
☆ Log Edit Notifications	

Monitoring

To monitor selective devices and notify if above threshold.
The notification is done using the Virtual Device Alert Message (idx=55).
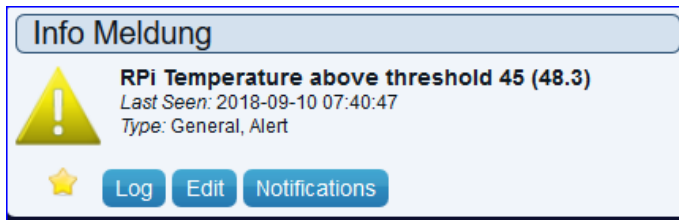
An email notification is send!

User Variables Thresholds

The thresholds are defined as user variables and used by the dzVents script.
(Setup > More Options > User variables).

Idx	Variable name	Variable type	Current value
5	TH_RPI_MEMORYUSAGE	Integer	80
4	TH_RPI_HDDUSAGE	Integer	70
3	TH_RPI_TEMPERATURE	Integer	41

dzVents Script

```
--[[
    rpi_monitor.lua
    Monitor the Raspberry Pi and notify, via alert message, in case threshold reached or
exceeded.
    Project: AtHome
    Interpreter: dzVents, Device
    See: AtHome.pdf
    Author: Robert W.B. Linn
    Version: 20180910
]]--

-- Module msgbox: /home/pi/domoticz/scripts/dzVents/scripts
local msgbox = require('msgbox')

-- Idx of the devices
local IDX_RPI_MEMORYUSAGE = 1
local IDX_RPI_HDDUSAGE = 3
local IDX_RPI_TEMPERATURE = 4

-- Thresholds are set via user variables, i.e. TH_RPI_MEMORYUSAGE

-- Check if the device state exceeds threshold and update control message
local function checkthreshold(domoticz, device, threshold)
    if (tonumber(device.state) > threshold) then
        local message = device.name .. ' above threshold ' .. threshold .. ' (' ..
device.state .. ')'
        msgbox.alertmsg(domoticz, domoticz.ALERTLEVEL_RED, message)
    end
end

return {
```

```
 on = {
        -- Devices idx to monitor
  devices = {
    IDX_RPI_MEMORYUSAGE,
    IDX_RPI_HDDUSAGE,
    IDX_RPI_TEMPERATURE
   }
 },
 execute = function(domoticz, device)
  -- RPi Memory Usage
  if (device.idx == IDX_RPI_MEMORYUSAGE) then
    checkthreshold(domoticz,device,domoticz.variables('TH_RPI_MEMORYUSAGE').value)
  end
  -- RPi HDD Usage
 if (device.idx == IDX_RPI_HDDUSAGE) then
    checkthreshold(domoticz,device,domoticz.variables('TH_RPI_HDDUSAGE').value)
  end
  -- RPi Temperature
 if (device.idx == IDX_RPI_TEMPERATURE) then
  checkthreshold(domoticz,device,domoticz.variables('TH_RPI_TEMPERATURE').value)
  end
 end
}
```

Electric Usage

Purpose

To measure the Electric Usage (Watt) obtained from a **volkszaehler** - Open Source Smart Meter - connected to a Raspberry Pi 2.

Hardware Setup
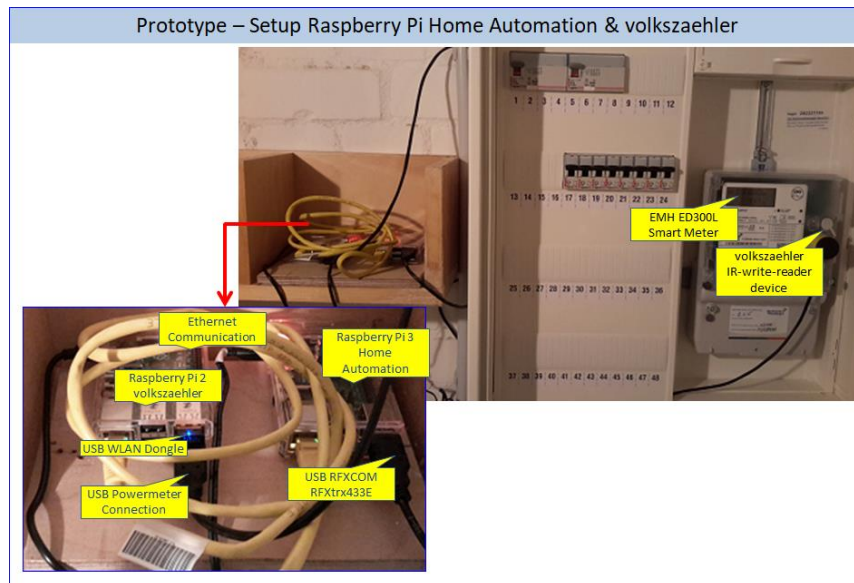

Volkzaehler Web Frontend

Solution

To get the electric usage from the volkszaehler and update the Domoticz device, a dzVents script is executed:

1. Send, every 5 minutes, a HTTP GET Request to the Raspberry Pi running the volkszaehler. The request response is handled using callback in the script.
2. The volkszaehler response is a JSON string.
3. Parse the JSON string to get the electric usage from key **data.average**.
   (which calculates the average electric usage over the last 5 minutes)
4. Update the Domoticz Electric Usage device:
   idx=44, VirtualSensors, Name=Stromverbrauch, Type=Usage, SubType=Electric.

Domoticz Device

| 44 | VirtualSensors | 82043 | 1 | Stromverbrauch | Usage | Electric | 140 Watt |
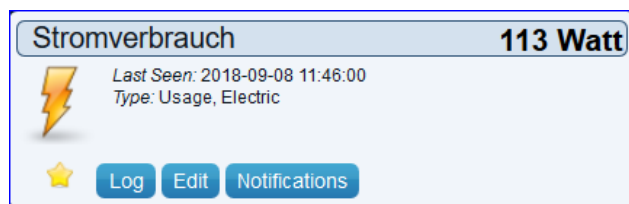
The HTTP GET request uses the channel (UUID: 958bce60-b342-11e8-b54f-bbec0573e1f4) and timestamp parameter:
- **from**=5+minutes+ago
- **to**=now

Request Examples
(see also the volkszaehler reference)
Data last 5 minutes (ensure to replace, in the URL request, the space by a +)

```
http://NNN.NNN.1.139/middleware.php/data/958bce60-b342-11e8-b54f-
bbec0573e1f4.json?from=5+minutes+ago&to=now
```

Data now

```
http://NNN.NNN.1.139/middleware.php/data/958bce60-b342-11e8-b54f-
bbec0573e1f4.json?from=now&to=now
```

Request Result JSON string

```
{"version":"0.3","data":{"tuples":[[1536243170351,926.045,1]],"uuid":" 958bce60-
b342-11e8-b54f-
bbec0573e1f4","from":1536243168796,"to":1536243170351,"min":[1536243170351,926.045
0147835],"max":[1536243170351,926.0450147835],"average":926.045,"consumption":0.4,
"rows":2}}
```

Obtain the power consumption from key

```
data.average
```

which is 926.045

dzEvents Script

The event uses a callback to handle the volkszaehler server response. The response is the JSON string from which the average electric usage is parsed.

```lua
-- HTTP GET request url with parameter from=5+minutes+ago, to=now
local requesturl = 'http://NNN.NNN.1.139/middleware.php/data/958bce60-b342-11e8-b54f-
bbec0573e1f4.json?from=5+minutes+ago&to=now'

-- Idx of the devices
local IDX_POWERCONSUMPTION = 44

return {
    -- active = true,
  on = {
   timer = {
     'every 5 minutes'
     -- 'every minute'
   },
   httpResponses = {
            'powerData'
        }
 },
 execute = function(domoticz, item)
   if (item.isTimer) then
     domoticz.openURL({
              url = requesturl,
              method = 'GET',
              callback = 'powerData'
    })
   end
   if (item.isHTTPResponse and item.ok) then
     -- parse the json string to get the average from key data.average
     local powerconsumption = tonumber(item.json.data.average)

     -- update the powerconsumption device
 domoticz.devices(IDX_POWERCONSUMPTION).updateEnergy(math.floor(powerconsumption))
     domoticz.log('Powerconsumption updated: ' .. powerconsumption)
   end
 end
}
```

Domoticz Log Entry

The log shows the HHTP request and the callback few seconds later

HTTP Request

```
2018-09-06 17:00:00.285 Status: dzVents: Info: ------ Start internal script:
powerconsumption_update:, trigger: every 5 minutes
2018-09-06 17:00:00.286 Status: dzVents: Info: ------ Finished
powerconsumption_update
2018-09-06 17:00:00.371 Status: EventSystem: Script event triggered:
/home/pi/domoticz/dzVents/runtime/dzVents.lua
```

Callback

```
2018-09-06 17:00:05.817 Status: dzVents: Info: Handling httpResponse-events for:
"powerData
```

```
2018-09-06 17:00:05.818 Status: dzVents: Info: ------ Start internal script:
powerconsumption_update: HTTPResponse: "powerData"
2018-09-06 17:00:05.844 Status: dzVents: Info: Powerconsumption updated: 150.206
2018-09-06 17:00:05.844 Status: dzVents: Info: ------ Finished
powerconsumption_update
```

volkszaehler

Overview

The volkszaehler (vz) Raspberry Pi (Model 2 with stretch) is connected to the electric meter reader EMH ED300L using the IR-Read-Write-USB-Device (provided by volkszaehler).
The data is polled on power consumption change and stored in a vz MySQL database.
Every 5 minutes Domoticz requests the average power consumption over the last 5 minutes.
The vz data is kept in the MySQL database for 24 hours (older data is deleted at midnight by a Python script – see Database Management).
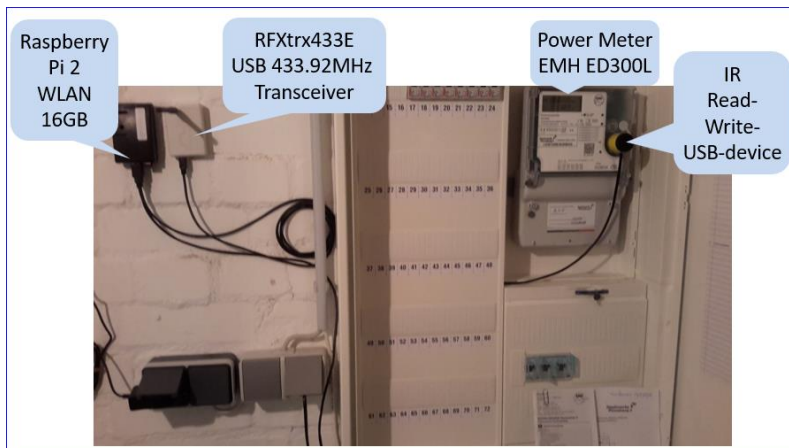
Note: See Appendix volkszaehler Setup for actual update and more details.

References: Raspberry-Pi,   API, Debug, EMH ED300L, Node-RED .

Setup Raspberry Pi

The Raspberry Pi is installed following the installation procedure described here.

Fix IP Address

The communication between the Domoticz Server and the vz Server via WLAN.
T3he vz Server has a fixed IP address NNN.NNN.1.NNN, which is defined in files:

/etc/dhcpcd.conf

```
# wlan interface
interface wlan0
static ip_address=rpi-vz-ip/24
static routers=NNN.NNN.1.1
static domain_name_servers=NNN.NNN.1.1
```

/etc/wpa_supplicant/wpa_supplicant.conf

```
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1
country=DE
network={
      ssid="o2-WLAN78_EXT"
      psk="******"
}
```

In the documentation, the volkzaehler IP is referred as **rpi-vz-ip**

USB Port Symlink

To ensure the right USB port is used, the Symlink ttyUSB-VZ is defined.

```
sudo nano /etc/udev/rules.d/99-usb-serial.rules
SUBSYSTEM=="tty", ATTRS{idVendor}=="10c4", ATTRS{idProduct}=="ea60",
ATTRS{serial}=="00F2E620", SYMLINK+="ttyUSB-VZ", MODE="0666"
```

Check
```
ls /dev/ttyU*
/dev/ttyUSB0   /dev/ttyUSB-VZ
```

Serial Link

The serial link must be setup with 9600 baud, 8n1. This is done by terminal command:

```
stty -F /dev/ttyUSB-VZ
1:0:8bd:0:3:1c:7f:15:4:5:1:0:11:13:1a:0:12:f:17:16:0:0:0:0:0:0:0:0:0:0:0:0:0:0:0:0
```

which must be added to rc.local to ensure correct settings for the USB port during setup.

```
sudo nano /etc/rc.local
# Set ttyUSB-VZ
stty -F /dev/ttyUSB-VZ
1:0:8bd:0:3:1c:7f:15:4:5:1:0:11:13:1a:0:12:f:17:16:0:0:0:0:0:0:0:0:0:0:0:0:0:0:0:0

exit 0
```

Check if data is received from the IR-Read-Write-USB-device:

```
xxd </dev/ttyUSB-VZ
00000000: 1b1b 1b1b 0101 0101 7607 000f 0016 bc87  ........v.......
00000010: 6200 6200 7263 0101 7601 0107 000f 0553  b.b.rc..v......S
00000020: 942d 0b09 0145 4d48 0000 532f 3801 0163  .-...EMH..S/8..c
00000030: 6a00 7607 000f 0016 bc88 6200 6200 7263  j.v.......b.b.rc
00000040: 0177 010b 0901 454d 4800 0053 2f38 0701  .w....EMH..S/8..
00000050: 0062 0aff ff72 6201 6505 530d fc7a 7707  .b...rb.e.S..zw.
00000060: 8181 c782 03ff 0101 0101 0445 4d48 0177  ...........EMH.w
00000070: 0701 0000 0009 ff01 0101 010b 0901 454d  ..............EM
```

Note
The tool minicom can also be used to setup the serial port.
```
sudo apt-get install minicom
sudo minicom -s
```

vz Setup Channels

Setting up the channels using the vz frontend (url=http://rpi-vz-ip) is crucial.
Must read: https://wiki.volkszaehler.org/howto/emh_pv-anlage?redirect=1
After the channels have been defined, the vzlogger.conf must be changed to add the channels.
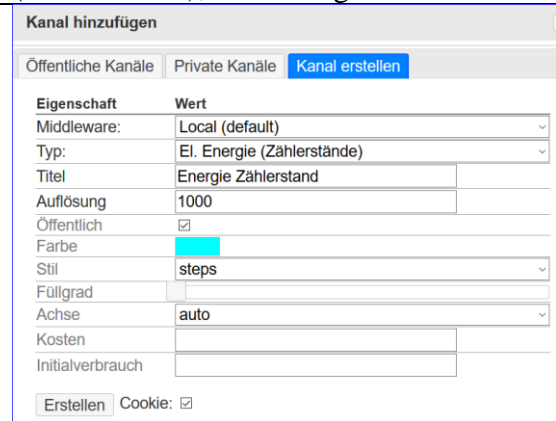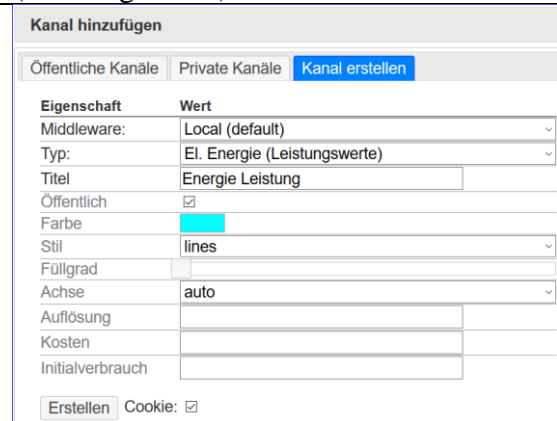
IMPORTANT
The **EMH ED300L** is sending in SML (Obis-Codes), which are 1.8.0 etc.
These are NOT S0-Impulses, but counters, i.e. „Zählerstände, Leistungswerte" depending Obis-Code.

There are two channels defined, from which channel 2 - Energy Consumption (16.7.0 Leistung) is used for Domoticz.

Channel 1: Haus 1.8.0 (Bezug +A), Typ El. Energie (Zählerstände), Auflösung 1000
UUID:5b8ea2a0-b342-11e8-bec6-15c040e6d041

Channel 2: Haus 16.7.0 (Leistung), Typ El. Energie (Leistungswerte)
UUID: 958bce60-b342-11e8-b54f-bbec0573e1f4

Channel: Zählerstand	**Channel: Leistung**
Haus 1.8.0 (Bezug +A), Typ El. Energie (Zählerstände), Auflösung 1000	Haus 16.7.0 (Leistung), Typ El. Energie (Leistungswerte)
UUID 5b8ea2a0-b342-11e8-bec6-15c040e6d041	**UUID** 958bce60-b342-11e8-b54f-bbec0573e1f4

// /etc/vzlogger.conf {
// Zählerstand
"uuid" : "5b8ea2a0-b342-11e8-bec6-15c040e6d041",
"middleware" : "http://localhost/middleware.php",
"identifier" : "1-0:1.8.0"
}
	/etc/vzlogger.conf {
	// Leistungswert
	"uuid" : "958bce60-b342-11e8-b54f-bbec0573e1f4",
	"middleware" : "http://localhost/middleware.php",
	"identifier" : "1-0:16.7.0"
	}

Web Frontend Settings	Web Frontend Settings

{
"version":"0.3",
"entity":
{"uuid":"5b8ea2a0-b342-11e8-bec6-15c040e6d041",
"type":"electric meter",
"color":"aqua",
"fillstyle":0,
"public":true,
"resolution":1000,
"style":"steps",
"title":"Energie Z\u00e4hlerstand",
"yaxis":"auto"}
}
	{
	"version":"0.3",
	"entity":
	{"uuid":"958bce60-b342-11e8-b54f-bbec0573e1f4",
	"type":"powersensor",
	"color":"aqua",
	"fillstyle":0,
	"public":true,
	"style":"lines",
	"title":"Energie Leistung",
	"yaxis":"auto"}
	}

vzlogger

The [vzlogger](#) is used to read the serial data and transfer the datato the middleware.

Configuration File

The configuration file is /etc/vzlogger.conf. Must [read](#) to understand the settings.
Ensure no empty lines are spaces after the last }.
IMPORTANT:
After making changes (sudo nano /etc/vzlogger.conf) restart the vzlogger and check status.

```
sudo systemctl restart vzlogger
sudo systemctl status vzlogger
```

/etc/vzlogger.conf

```
{
  "daemon": true,
  "verbosity": 0,
  "log": "/var/log/vzlogger.log",
  "retry": 30,
  "local": {
    "enabled": false,
    "port": 8080,
    "index": true,
    "timeout": 30,
    "buffer": 600
  },
  "push": [],
  "meters" : [
    {
      "enabled" : true,
      "protocol" : "sml",
      "device" : "/dev/ttyUSB-VZ",
      "baudrate": 9600,
      "parity": "8n1",
      "channels": [
        {
          // Leistungswert
          "uuid" : "958bce60-b342-11e8-b54f-bbec0573e1f4",
          "middleware" : "http://localhost/middleware.php",
          "identifier" : "1-0:16.7.0"
        },
        {
          // Zählerstand
          "uuid" : "5b8ea2a0-b342-11e8-bec6-15c040e6d041",
          "middleware" : "http://localhost/middleware.php",
          "identifier" : "1-0:1.8.0"
        }
      ]
    }
  ]
}
```

Start Stop Status

Terminal commands

Stop: sudo systemctl stop vzlogger
Start: sudo systemctl start vzlogger
Restart: sudo systemctl restart vzlogger
Status: sudo systemctl status vzlogger

Logfile

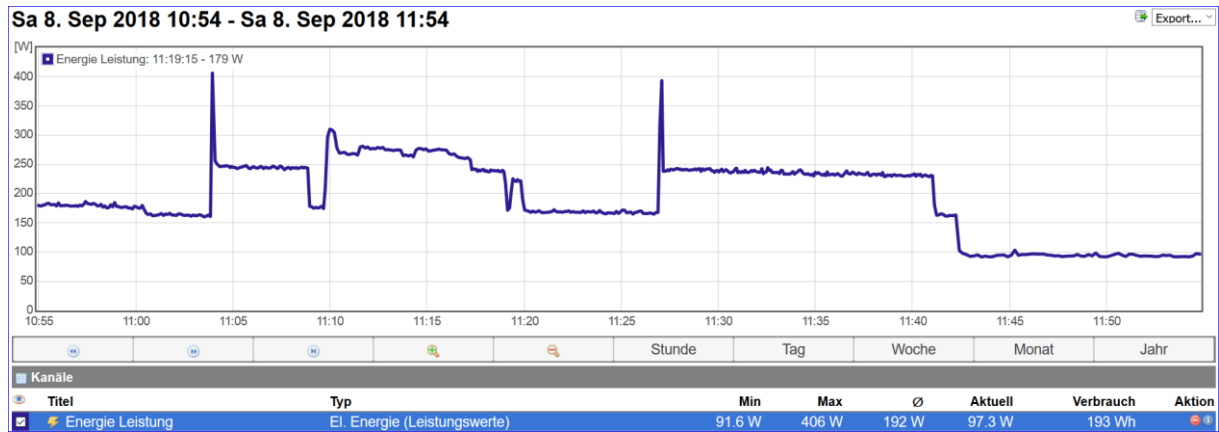**Check**

```
sudo cat /var/log/vzlogger.log
```

Delete

```
sudo systemctl stop vzlogger
sudo rm /var/log/vzlogger.log
sudo systemctl start vzlogger
sudo cat /var/log/vzlogger.log
```

vz Web Frontend

URL=http://rpi-vz-ip/?uuid=958bce60-b342-11e8-b54f-bbec0573e1f4
(Leistungswerte)

Read more [here](#).

Database Management

To manage the data stored in a MySQL database, the tool phpMyAdmin is used.
Browser URL = http://rpi-vz-ip/phpmyadmin/

User vz-admin

Prior using, create a user vz-admin and grant access.

```
sudo mysql -uroot -praspberry
CREATE USER 'vz-admin'@'%' IDENTIFIED BY 'secure';
GRANT USAGE ON *.* TO 'vz-admin'@'%';
GRANT ALL PRIVILEGES ON `volkszaehler`.* TO 'vz-admin'@'%' WITH GRANT OPTION;
exit
```

Login phpMyAdmin

Username: vz-admin, Password: *****, Database: volkszaehler

Check Database Size

URL

```
http://rpi-vz-ip/middleware.php/capabilities/database.json?
```
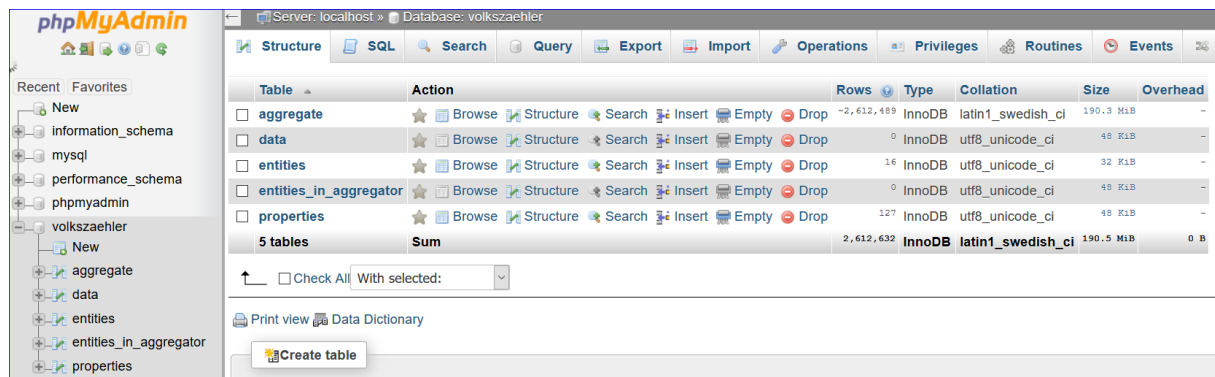
Result

```
{"version":"0.3", "capabilities": {"database":
 {"data_rows":67235725,"data_size":5935382528,"aggregation_enabled":1,"aggregation
_rows":2611868,"aggregation_ratio":25.742}
 }}
```

Size of database and access_log

Limit logging
```
sudo nano /etc/mysql/my.cnf
[mysqld]
expire_log_days = 2
max_binlog_size = 5M
```

Delete Data

There are several ways to delete data from the vz MySQL database.

Option 1

This ist the hard way of deleting data via MySQL commands.
Example deleting all data older then 24 hours from database volkszaehler, table data.

```
mysql --user=root -praspberry

mysql> grant select, update, insert, delete on volkszaehler.* to vz@localhost;
Query OK, 0 rows affected (0.01 sec)

mysql> use volkszaehler

mysql> delete FROM data where timestamp< (unix_timestamp(current_timestamp)-
24*60*60)*1000 ;
```
Note
This operation might take long…

Check the database size after completion

Option 2

Use terminal command

```
wget -O - -q "http://rpi-vz-ip/middleware.php/data/27e28d00-b2dc-11e8-903d-
c3b75ae4ef20.json?operation=delete&from=now
```


Option 3

Use URL

Prepare mysql
```
mysql --user=vz-admin -psecure

mysql> grant select, update, insert, delete on volkszaehler.* to vz@localhost;
Query OK, 0 rows affected (0.01 sec)
```

URL deleting data
```
http://rpi-vz-ip/middleware.php/data/958bce60-b342-11e8-b54f-
bbec0573e1f4.json?operation=delete&from=1+day+ago
```

Result JSON
```
{"version":"0.3","rows":37466}
```

URL Check databse size
```
http://rpi-vz-ip/middleware.php/capabilities/database.json?
```

Result JSON
```
{"version":"0.3","capabilities":{"database":{"data":{"rows":72047,"size":12632064}
,"aggregation":{"rows":0,"size":49152,"ratio":0}}}}
```

vzclean

Purpose

To delete, every day at midnight, data which is older then 24 hours from the database volkszaehler, table data.

Solution

A Python script (vzclean.py) logs into the database as user vz-admin and deletes the data from table data. The script is executed, via bash script (vzclean.sh) once a day at 00:05.

Script location:

```
/home/pi/scripts
```

The script uses the config file vzclean.ini to obtain MySQL database and user login information.

Hint

After running vzclean or to check adhoc, use URL

```
http://rpi-vz-ip/middleware.php/capabilities/database.json?
```

Python

vzcleanconfig.py

```python
## See https://pypi.org/project/configparser/
import configparser

def read_db_config(filename='vzclean.ini', section='mysql'):
    """ Read database configuration file and return a dictionary object
    :param filename: name of the configuration file
    :param section: section of database configuration
    :return: a dictionary of database parameters
    """
    # create parser and read ini configuration file
    parser = configparser.ConfigParser()
    parser.read(filename)

    # get section, default to mysql
    db = {}
    if parser.has_section(section):
        items = parser.items(section)
        for item in items:
            db[item[0]] = item[1]
            # print(db[item[0]])
    else:
        raise Exception('{0} not found in the {1} file'.format(section, filename))

    # print(db)
    return db
```

vzclean.ini

```ini
[mysql]
host = localhost
database = volkszaehler
user = vz-admin
password = *****
```

vzclean.py

```python
#!/usr/bin/env python
"""
File: vzclean.py
Author: Robert W.B. Linn, 20180909
Project: AtHome
Purpose: Delete all channel data prior last 24 h from database volkszaehler, table data.
Run once per day via crontab.
The scripts logs to a file. If the script is executed with parameter l, then the log is
printed to the console.
Run the script: python3 vzclean.py  or python3 vzclean.py l
The script is executed via crontab "5 0 * * * /home/pi/scripts/vzclean.sh l" using the bash
script vzclean.sh.
        cd /home/pi/scripts
        python3 vzclean.py $1
    Example Log
        vzclean 20180909
        Connection: OK
        Rows befor delete: 31
        Rows after delete: 1
        Rows last 24h: 25067
        Connection closed

    Install mysql-connector
```

```
    sudo apt-get update
    sudo apt-get upgrade
    sudo apt-get -y install python-mysql.connector
    sudo apt-get -y install python3-mysql.connector

    References
    http://www.mysqltutorial.org/python-mysql-query/

    volkszaehler database structure
    https://wiki.volkszaehler.org/development/schema
    volkszaehler API ref:
    https://wiki.volkszaehler.org/development/api/reference
    Table data
    id, int(11); channel_id, int(11); timestamp, bigint(20); value, double
"""
# Import
from mysql.connector import MySQLConnection, Error
from vzcleanconfig import read_db_config
import sys

vzclean = 'vzclean 20180909'
logfilename = 'vzclean.log'

# Define the queries including test queries
query_select_entities = "SELECT * FROM entities ;"
query_select_data = "SELECT * FROM data ;"
query_select_data_by_channel_ts = "SELECT * FROM data WHERE timestamp >
(unix_timestamp(current_timestamp)-24*60*60)*1000 ;"
query_select_data_prior_last24h = "SELECT * FROM data WHERE timestamp <
(unix_timestamp(current_timestamp)-24*60*60)*1000 ;"
query_select_data_prior_last24h_count = "SELECT COUNT(channel_id) FROM data WHERE timestamp
< (unix_timestamp(current_timestamp)-24*60*60)*1000 ;"
# Ref: http://www.mysqltutorial.org/python-mysql-delete-data/
query_delete_data_prior_last24h = "delete FROM data where timestamp<
(unix_timestamp(current_timestamp)-24*60*60)*1000 ;"

def log(logfile, text):
    """Write a line to the logfile
    :logfile: Logfile opened for writing (w+) or append (a+)
    :text: Line to write
    """
    logfile.write('{}\r\n'.format(text))

def logprint(logfilename):
    """Print the content of the logfile
    :logfile: Logfile opened for reading (r)
    """
    f = open(logfilename, "r")
    content = f.read()
    print(content)

def show_tables(conn):
    """Show the table for database volkszaehler
    :conn: Database connection
    """
    cursor = conn.cursor()
    cursor.execute("SHOW TABLES")
    for x in cursor:
        print(x)

def select_fetchone(conn, qry):
    """Query a table
    :conn: Database connection
    :qry: Query to execute
    """
    cursor = conn.cursor()
    cursor.execute(qry)
```

```
    row = cursor.fetchone()

    while row is not None:
        print(row)
        row = cursor.fetchone()

def select_count(conn, qry):
    """Get the number of records
    :conn: Database connection
    :qry: Select query with COUN()
    :return: Number of records
    """
    cursor = conn.cursor()
    cursor.execute(qry)
    result = cursor.fetchone()
    return result[0]

def delete_data(conn, qry):
    """Delete data from a
    :conn: Database connection
    :qry: Delete statement
    """
    cursor = conn.cursor()
    cursor.execute(qry)
    # accept the change
    conn.commit()

# Connect and run queries
def cleandata(showlog):
    logfile = open(logfilename,"w+")
    log(logfile, vzclean)

    """ Connect to MySQL database """
    db_config = read_db_config()

    try:
        conn = MySQLConnection(**db_config)

        if conn.is_connected():
            log(logfile, 'Connection: OK')
            #show_tables(conn)
            #select_fetchone(conn, query_select_entities)
            #select_fetchone(conn, query_select_data_by_channel_ts)

            # delete data prior last 24h
            log(logfile, 'Rows befor delete: {}'.format(select_count(conn,
query_select_data_prior_last24h_count)))
            delete_data(conn, query_delete_data_prior_last24h)
            log(logfile, 'Rows after delete: {}'.format(select_count(conn,
query_select_data_prior_last24h_count)))
            log(logfile, 'Rows last 24h: {}'.format(select_count(conn,
query_select_data_by_channel_ts)))
        else:
            log(logfile, '[ERROR] Database connection: failed')

    except Error as error:
        log(logfile, '[ERROR] Database connection:{}'.format(error))

    finally:
        conn.close()
        log(logfile, 'Connection closed')
        logfile.close()

        # If showlog argument is given, print the log to the console
        if showlog == 'l':
            logprint(logfilename)
```

```python
if __name__ == '__main__':
    if (len(sys.argv) == 2):
        cleandata(sys.argv[1])
    else:
        cleandata('')
```

Bash

vzclean.sh

```bash
#!/bin/bash
# Run vzclean Python script
# Optional Parameter: l to show the log
# Ensure to make the script executable: sudo chmod +x vzclean.sh
# Robert W.B. Linn, 20180909

# Set the scripts folder
cd /home/pi/scripts

# Run the script
python3 vzclean.py $1
```

Crontab

Add to crontab the following lines to execute the script daily at 5 minutes past midnight.

```
crontab -e
# Run vzclean daily at 00:05
5 0 * * * /home/pi/scripts/vzclean.sh l
```

Ethernet Link volkszaehler Raspberry Pi

This is an example incase the vz Raspberry Pi and the Domoticz Raspberry Pi are direct connected via an Ethernet cable.
For the wired connection, 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

vz Raspberry Pi	Domoticz Raspberry Pi
[Raspbian GNU/Linux 8 (stretch)]	[Raspbian GNU/Linux 8 (stretch)]
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/dhcpcd.conf	sudo nano /etc/dhcpcd.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 vz Raspberry Pi to the Domoticz Raspberry Pi:
- Login to vz Raspberry Pi as user pi.
- Check if the Raspberry Pi Domoticz is reachable: $ping 169.254.87.85
- Check if the sshd service is running on the vz Raspberry Pi: $ps ax | grep sshd
- Connect to the Domoticz Raspberry Pi via ssh: $ssh pi@169.254.87.85
  pi@169.254.87.85's password: *****
- The pi@DoPro prompt is shown

Coffee Machine Monitor

Purpose

To monitor the Coffee Machine, start & end time.
The Coffee Machine swiches OFF, 2 hours after being switched ON.

Solution

Monitor the Electric Usage Device (idx=44) value if above certain threshold (> 1500 Watt) between 07:00 – 08:00. If above threshold, set the Alert Message.

Monitoring is handled by a dzVents Script for device changes and timer.
The script uses a script level persistent variable "notified" to handle the Alert Message notifications (see dzvents script).

dzVents Script

```
local msgbox = require('msgbox')
local IDX_STROMVERBRAUCH = 44
local TH_STROMVERBRAUCH = 1500
local MONITOR_START = 'at 07:00'
local MONITOR_START_END = 'at 07:01-07:59'
local MONITOR_END = 'at 08:00'

return {
  on = {
     devices = {
        IDX_STROMVERBRAUCH
      },
      timer = {
        MONITOR_START,
        MONITOR_START_END,
        MONITOR_END,
      },
  },
  data = {
       notified = { initial = 0 }
     },
  execute = function(domoticz, item)
  if (item.isTimer) then
    if (item.trigger == MONITOR_START) then
     domoticz.data.notified = 0
    end
    if (item.trigger == MONITOR_END) then
     timeoff = os.date("*t")
     message = ('Kaffee Machine AUS %02d:%02d'):format(timeoff.hour, timeoff.min)
     msgbox.alertmsg(domoticz, domoticz.ALERTLEVEL_GREY, message)
    end
  end

  if (item.isDevice) then
    device = item
    if (device.WhActual > TH_STROMVERBRAUCH) and (domoticz.data.notified == 0) then
     timeon = os.date("*t")
     message = ('Kaffee Machine AN  %02d:%02d'):format(timeon.hour, timeon.min)
     -- calculate the time off = timeon + 2 hours
     timeoff = os.date("*t", os.time() + 2*60*60)
     -- define message to notify the time coffee machine switches off
```

```
    message = ('Kaffee Machine aus um %02d:%02d'):format(timeoff.hour,timeoff.min)
     -- update alert message
    msgbox.alertmsg(domoticz, domoticz.ALERTLEVEL_ORANGE, message)
    -- set notified flag
    domoticz.data.notified = 1
   end
  end
end
}
```

Alert Message

Purpose

To display an Alert Message on the UI tabs Dashboard and Utility.

The Alert Message is populated by various dzEvents, i.e.
- Raspberry Pi monitor (rpi_moonitor.lua)
- Hue lamps living room timed switch (hue_wz_timer.lua)
- More see the dedicated functions

Solution

Add a Virtual Devices named "Info Meldung", Type General, SubType Alert, Idx=55.

⚠	55	VirtualSensors	82054	1	Alert Message	General	Alert	Hue MakeLab switched OFF 18:56:43

Add a User Variable IDX_ALERTMSG, Integer, Value=55 (see previous).

6	IDX_ALERTMSG	Integer	55

Dashboard

⏺ Info Meldung	RPi Temperature above threshold 45 (48.3) ⚠

Widget

Info Meldung

RPi Temperature above threshold 45 (48.3)
Last Seen: 2018-09-10 07:40:47
Type: General, Alert

⭐ Log Edit Notifications

Log Entries

Info Meldung	
Date ▾	**Data** ▴
2018-09-10 07:40:47	RPi Temperature above threshold 45 (48.3)
2018-09-09 22:30:00	WZ Hue Lampen AUS 22:30:00
2018-09-09 19:53:00	Hue Lampen WZ AN19:53:00
2018-09-09 17:20:09	Keine

Showing 1 to 4 of 4 entries — First Previous 1 Next Last

Clear

dzEvents Script

Simple example: Switch the Hue MakeLab ON, then the event will turn the lamp OFF and updates the alert sensor with a message.

```
local IDX_HUE_MAKELAB = 118;

local function isnowtime(domoticz)
    return domoticz.time.rawTime
end

local function setalertmsg(domoticz, level, msg)
  domoticz.devices(domoticz.variables('IDX_ALERTMSG').value).updateAlertSensor(level, msg)
end

return {
    -- active = true,
  on = {
   devices = {
     IDX_HUE_MAKELAB
   }
  },
  execute = function(domoticz, device)
    if (device.state == 'On') then
     device.switchOff()
     local msg = device.name .. ' switched OFF ' .. isnowtime(domoticz)
     setalertmsg(domoticz, domoticz.ALERTLEVEL_GREEN, msg)
    end
  end
}
```

For all events use the function:

```
-- Update the alert message with level.
local function setalertmsg(domoticz, level, msg)
  domoticz.devices(domoticz.variables('IDX_ALERTMSG').value).updateAlertSensor(level, msg)
end
```

Example

```
local msg = device.name .. ' > ' .. threshold .. ' (' .. device.state .. ')'
setalertmsg(domoticz, domoticz.ALERTLEVEL_YELLOW, msg)
```

Alert Message Reset

Purpose

To reset the Alert Message using a Switch.

Solution

Create a virtual device type: Light/Switch, SubType: Switch.

If the switch is turned ON, the text of the alert sensor (Type=General, SubType=Alert, idx=55) is set to the value of the User Variable DEF_ALERTMSG is set and the switch is turned OFF again. This is handled using a dzVents script.

Device List

119	VirtualSensors	000140C7	1	Info Meldung Reset	Light/Switch	Switch	Off

Widget
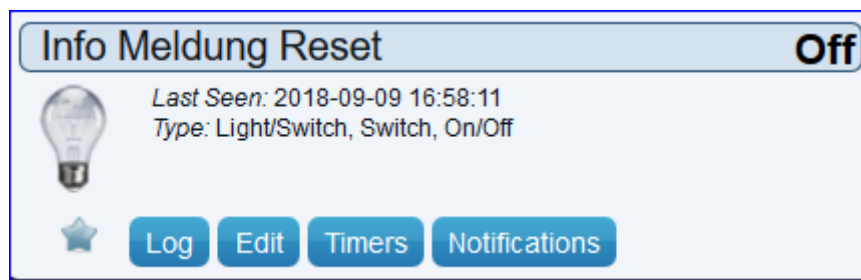
User Variable

Idx	Variable name	Variable type	Current value
7	DEF_ALERTMSG	String	Keine

dzVents Script

```
--[[
    alertmsg_reset.lua
    If the switch is turned ON,
    the text of the alert sensor (idx=55) is set to the value of the User Variable
DEF_ALERTMSG and
    the switch is turned OFF again.
]]--

-- External modules
local msgbox = require('msgbox')

-- Idx of the devices
local IDX_ALERTMSG_RESET = 119

return {
  on = {
    devices = {
      IDX_ALERTMSG_RESET
    }
  },
  execute = function(domoticz, device)
    domoticz.log('Device ' .. device.name .. ' was changed ' ..
domoticz.devices(domoticz.variables('IDX_ALERTMSG').value).text, domoticz.LOG_INFO)
    if (device.state == 'On') then
      -- only change if the current text differs from the default text
      if (domoticz.devices(domoticz.variables('IDX_ALERTMSG').value).text ~=
domoticz.variables('DEF_ALERTMSG').value) then
        local message = domoticz.variables('DEF_ALERTMSG').value
       msgbox.alertmsg(domoticz, domoticz.ALERTLEVEL_GREY, message)
       domoticz.log(message)
      end
      device.switchOff()
    end
  end
}
```

Database Tables

The Alert Message device (idx=55) is stored in the domoticz.db, table DevicesStatus.

```
SELECT * FROM DeviceStatus WHERE ID=55;
```

ID	Har...	Devi...	Unit	Name	Used	Type	Sub...	Swit...	Fav...	Sign...	Batt...	nVal...	sValue	LastUpdate	Order	Addj...	Addj...	Addj...	Addj...	StrP...	StrP...	Last...	Prot...	Cust...	Des...	Opt
55	6	82054	1	Alert Meldung	1	243	22	0	1	12	255	0	Reset	2018-09-13 13:57:03	55	0.0	1.0	0.0	1.0			0	0	0		

The log entries are stored in table LightingLog.

```
SELECT * FROM LightingLog WHERE DeviceRowID=55;
```

Devi...	nVal...	sValue	Date	User
55	2	[ERROR] Powerconsumption: 7 2018-09-13 11:...	2018-09-13 11:50:00	
55	2	[ERROR] Powerconsumption: 7 2018-09-13 11:...	2018-09-13 11:55:00	
55	2	[ERROR] Electric usage: 7 2018-09-13 12:00:00	2018-09-13 12:00:00	
55	2	[ERROR] Powerconsumption: 7 2018-09-13 12:...	2018-09-13 12:00:00	
55	2	[ERROR] Electric usage: 7 2018-09-13 12:05:00	2018-09-13 12:05:00	
55	2	[ERROR] Powerconsumption: 7 2018-09-13 12:...	2018-09-13 12:05:00	
55	2	[ERROR] Powerconsumption: 7 2018-09-13 12:...	2018-09-13 12:10:00	
55	0	Keine	2018-09-13 13:50:36	
55	0	Keine	2018-09-13 13:50:42	
55	0	Reset	2018-09-13 13:57:03	

In case entries should be deleted, stop domoticz, delete the rows using SQLite tool and start domoticz.

Info Message (Not Used)

Purpose

NOT USED – Replaced by the Alert Message.

To display an Info Message on the Dashboard.
The Info Message is populated by various dzEvents, i.e.

- Raspberry Pi monitor (rpi_moonitor.lua)
- Hue lamps living room timed switch (hue_wz_timer.lua)
- More see the dedicated functions

Solution

Add a Virtual Sensor named "Control Message", Type General, SubType Text.

52	VirtualSensors	00082051	1	Control Message	General	Text	Hue MakeLab Switching OFF 16:55:34

In the various dzEvents, the device is updated.

Control Message	Hue MakeLab switched OFF 17:18:05

Control Message

Text — **Hue MakeLab switched OFF 17:18:05**
Last Seen: 2018-09-05 17:18:05
Type: General, Text

Log Edit
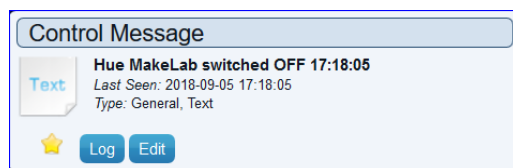
dzEvents Script

```
local IDX_HUE_MAKELAB = 118;
local IDX_CONTROLMSG = 52

local function isnowtime(domoticz)
    return domoticz.time.rawTime
end

return {
    active = true,
  on = {
    devices = {
      IDX_HUE_MAKELAB
    }
  },
  execute = function(domoticz, device)
    if (device.state == 'On') then
      device.switchOff()
      domoticz.devices(IDX_CONTROLMSG).updateText(device.name .. ' switched OFF ' ..
isnowtime(domoticz))
    end
  end
}
```

Web Frontend Customized

Purpose

To build a Custom Web Frontend running on a dedicated web server.

Web Front End Sample

Display weather data from 3 devices and update every 5 seconds. A clock is running every second.
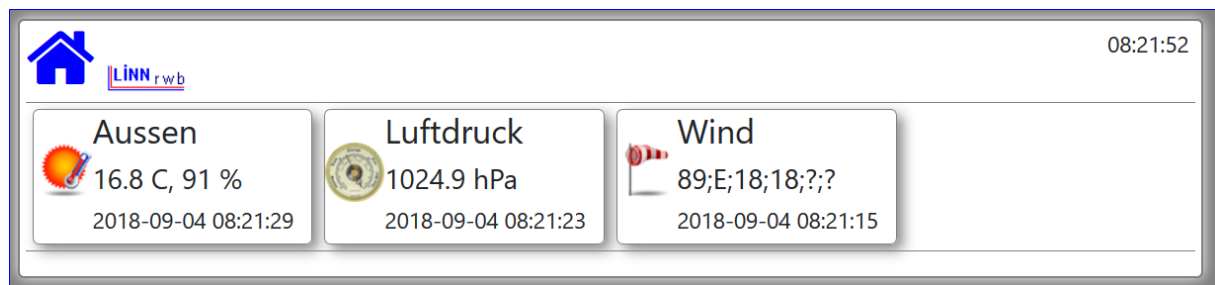
Devices Used

```
Temperature & Humidity
(Name="Aussen", idx=14, Hardware=RFXtrx433e, Type=Temp + Humidity, SubType=Cresta,
TFA TS34C)

Luftdruck
(Name="Luftdruck", idx=116, Hardware=VirtualSensors, Type=General,
SubType=Barometer)

Wind
(Name="Wind", idx=117, Hardware=VirtualSensors, Type=Wind, SubType=WTGR800)
```


Concept

The device data is requested (URL) via the Domoticz engine using JSON.
The Domoticz engine allows to interact with all devices using JSON (nice solution).

The UI uses jQuery, Bootstrap and Domoticz icons.

The data for the favorite (=Dashboard) devices is requested from the Domoticz server via URL.

```
URL: domoticzurl:8080/json.htm?type=devices&used=true&filter=all&favorite=1
```

From the request result (JSON string), selective data for the devices with idx=14,116,117 is obtained.

JSON Result Snippet

The URL returns the data in JSON format.

This snippet shows some of the properties for the devices.
The key result[] is an array with the device information.

In this example the results array (made bold) has a length of 3 for the devices with idx,14,116,117. For each of the devices, the Keys "Data" and "Idx" are used.

Using JavaScript the result array is parsed (see below).

```
{
    "ActTime" : 1536051617,
        ...
    "app_version" : "4.9980",
    "result" : [
        {
            "Data" : "20.9 C, 79 %",
            "idx" : "14"
        },
        {
            "Data" : "1025 hPa",
            "idx" : "116"
        },
        {
            "Data" : "89;E;14;21;?;?",
            "idx" : "117"
        }
    ],
    "status" : "OK",
    "title" : "Devices"
}
```

JavaScript Snippet

```
// Define the idx to of the devices display
var arrayIdx = [14,116,117];
url = domoticzURL + "json.htm?type=devices&used=true&filter=all&favorite=1"
$.getJSON(url, function(result){
  var idx = 0;
  var sensorInfo = "";
  for (i = 0; i < result.result.length; i++) {
    idx = parseInt(result.result[i].idx);
    if (arrayIdx.indexOf(idx) > -1) {
      sensorInfo += setSensorInfo(result,i);
      console.log("Found ixd=" + idx);
    }
  …
}
```

HTML & JavaScript

```
<!DOCTYPE html>
<html>
<head>
  <title>AtHome Weather Frontend</title>
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <meta charset="utf-8"/>
  <!-- Reference to the external CSS -->
  <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css">
  <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/4.7.0/css/font-awesome.min.css">
  <!-- Reference to the external JS -->
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
  <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/4.1.3/js/bootstrap.min.js"></script>
  <!-- Styling -->
  <style>
    body{background-color: grey; padding: 0px 1% 0% 1%;}
    .container{padding: 0px 20px 20px 20px;border-radius: 5px; background-color: white;
border-style: solid; border-color: grey; border-width: 2px;box-shadow: 0px 0px 10px
white;margin-top: 2%}
    .row {border-bottom-style: solid; border-color: grey; border-width: 1px;}
    #linn_logo {max-width: 100px; margin: 5px; }
    #top-bar{text-align: left; margin-bottom: 1px; padding: 0px;}
    #clock{float: right; margin: 5px;}
    .v-align {float: none; display: inline-block;vertical-align: middle;} /*class for
vertically aligning elements in a row*/
    #house-image{font-size: 9em; vertical-align: middle; display: inline-block; margin-top:
10px; margin-bottom: 10px;}
    #house-status-text{text-align: left;}
    #outside-temperature{text-align: left;}
    #outside-humidity{text-align: left;}
    #outside-humiditystatus{text-align: left;}
    #outside-lastupdate{text-align: left;}
    .sensor-outer-box {padding: 0px;}
    .sensor-box{padding: 0px;margin: 5px; background-color: white; box-shadow: 5px 5px 10px
#888888; border-style: solid; border-color: grey; border-width: 1px; border-radius: 5px;}
    .sensor-icon{vertical-align: 80%; margin: 0px;}
    .sensor-data {display: inline-block; margin: 0px;}
    </style>
  </head>

  <body>
    <div class="container">
      <div class="row">
        <div class="col-sm-12" id="top-bar">
          <i class="fa fa-home" style="font-size:60px;color:blue;"></i>
          <img id="linn_logo" src="linn_logo.png">
          <div id="clock">
          </div>
        </div>
      </div>
      <div class="row" id="sensor-container">
      </div>
    </div>

    <script>
      $(document).ready(function(){
        var domoticzURL = "http://rpi-vz-ip:8080/";
        //var domoticzURL = "http://localhost:8080/";
        // Define the idx to of the devices display
        var arrayIdx = [14,116,117];
        // Timer interval polling data (ms)
        var intervalTimer = 5000;
```

```
    // Update the sensorinfo
    $(function() {
      updateClock();
      updateSensors();
    });

    // Returns the image of the sensor that is stored in the image folder on the Domoticz
server
    // /home/pi/domoticz/www/images
    function getSensorImage(TypeImg){
      var sensorURL = "";
      switch(TypeImg) {
        case "dimmer":
          return domoticzURL + "images/Light48_On.png";
          break;
        case "temperature":
          return domoticzURL + "images/temp-20-25.png";
          break;
        case "wind":
          return domoticzURL + "images/wind48.png";
          break;
        case "gauge":
          return domoticzURL + "images/baro48.png";
          break;
        default:
          return domoticzURL + "images/Light48on.png";
      }
    }

    function setSensorInfo(result,i){
      var sensorInfo = "";
      sensorInfo += ''+
        '<div class="col-sm-6 col-md-4 col-lg-3 sensor-outer-box">'+
          '<div class="sensor-box">'+
            '<img class="sensor-icon" src="' + getSensorImage(result.result[i].TypeImg)
+'"></img>'+
            '<div class="sensor-data">'+
              '<h4>' + result.result[i].Name + '</h4>'+
              '<h5>' + result.result[i].Data +'</h5>'+
              '<h6>'  + result.result[i].LastUpdate+'</h6>'+
            '</div>'+
          '</div>'+
        '</div>';
      return sensorInfo;
    }

    function updateClock() {
      var d = new Date();
      var hours = d.getHours();
      if (hours < 10) { hours = '0' + hours;}
      var minutes = d.getMinutes();
      if (minutes < 10) { minutes = '0' + minutes;}
      var seconds = d.getSeconds();
      if (seconds < 10) { seconds = '0' + seconds;}
      document.getElementById("clock").innerHTML = hours + ":" + minutes + ":" + seconds;
      setTimeout(updateClock, 1000);
    }


    function updateSensors() {
      // Get data for the favorite (=Dashboard) devices and select the idx=14,116,117
      // URL: domoticzurl:8080/json.htm?type=devices&used=true&filter=all&favorite=1
      url = domoticzURL + "json.htm?type=devices&used=true&filter=all&favorite=1"
      $.getJSON(url, function(result){
        var idx = 0;
        var sensorInfo = "";
```

```
        for (i = 0; i < result.result.length; i++) {
         idx = parseInt(result.result[i].idx);
         if (arrayIdx.indexOf(idx) > -1) {
          sensorInfo += setSensorInfo(result,i);
         }
        }
        document.getElementById("sensor-container").innerHTML = sensorInfo;
        //Finally set a timer to repeatly poll the status of the sensors
        setTimeout(updateSensors,intervalTimer);
      });
    }
   });
  </script>
 </body>
</html>
```

App AtHome

Purpose

To develop an Android & Windows Desktop App displaying & controlling the Domoticz favorite devices, i.e. the devices shown in the Domoticz Web UI Tab Dashboard.

Solution

Android app developed with B4A v8.5.

Read the B4J Forum Share My Creations post here.

Windows Desktop application developed with B4J v6.8.

Read the B4J Forum Share My Creations post here.
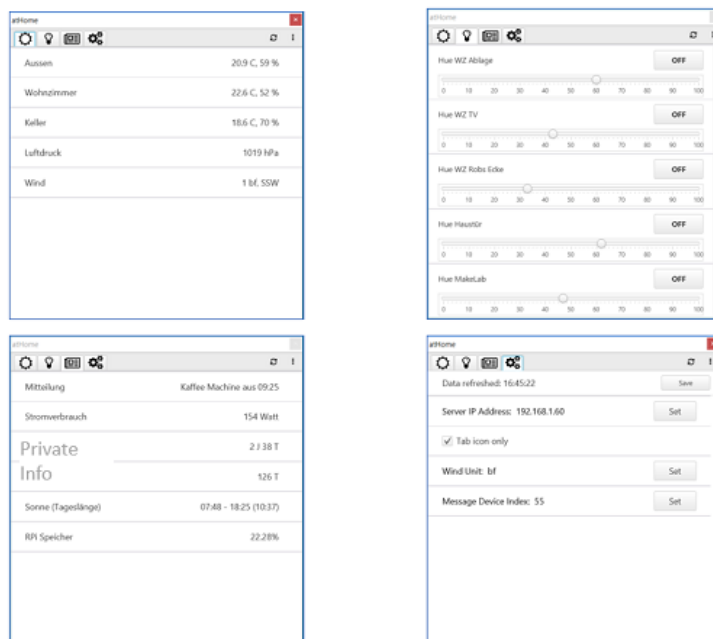
Note: current version in use has many more devices.

Windows Desktop application developed with [Lazarus](#) v1.8.4.

Not published yet.
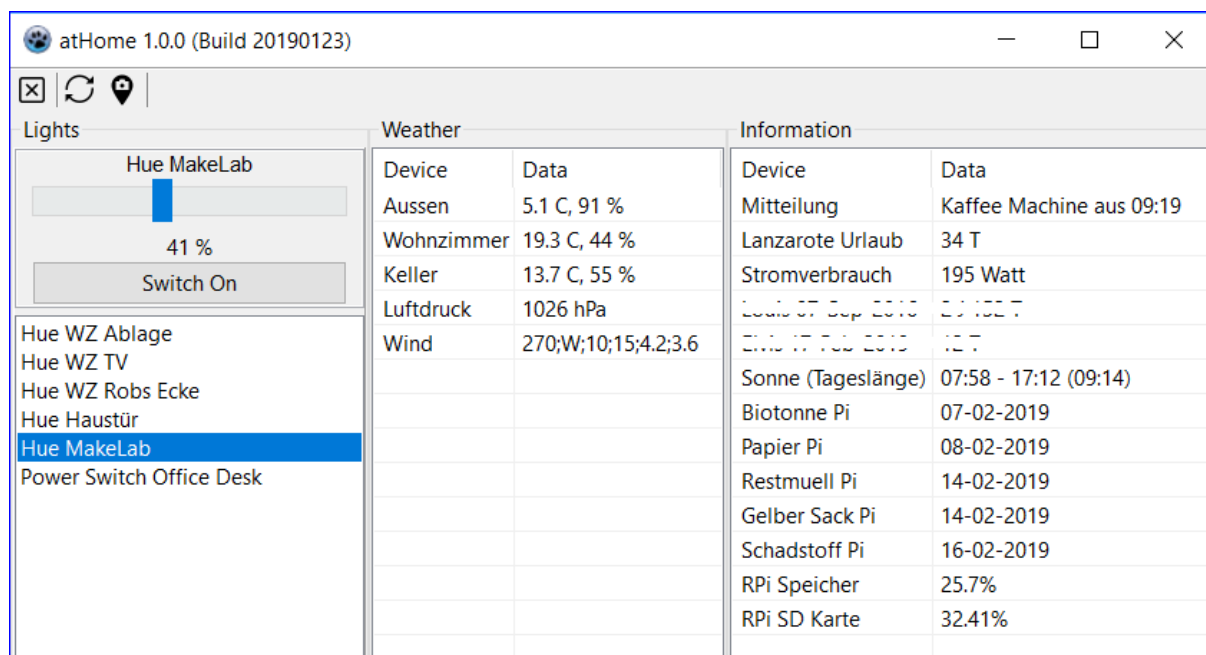
User Variables

User variables are used mainly for the events.

Syntax

PREFIX_DEVICE

In UPPERCASE!

Prefix	Meaning	Example
DEF	Default	DEF_ALERTMSG
IDX	Device Index	IDX_ALERTMSG
TH	Threshold	TH_RPI_HDDUSAGE

Example
User Variables with Variable Type and Value

Variable name	Variable type	Current value
DEF_ALERTMSG	String	Keine
IDX_ALERTMSG	Integer	55
TH_RPI_MEMORYUSAGE	Integer	80
TH_RPI_HDDUSAGE	Integer	70
TH_RPI_TEMPERATURE	Integer	45

Script Usage Examples

Lua, dzVents

Read Value

```
-- Update the alert message with level.
function msgbox.alertmsg(domoticz, level, msg)
  domoticz.devices(domoticz.variables('IDX_ALERTMSG').value).updateAlertSensor(level, msg)
end
```

```
local message = domoticz.variables('DEF_ALERTMSG').value
```

Set Value

```
domoticz.variables('DEF_ALERTMSG').set('Hello World')
```

Read here how to use with dzVents.

Explore: Domoticz Functions

Domoticz offers a variety of functionality. The next chapters explore some of them, prior using in the AtHome Functions.

Events System

Purpose

To explore the Domoticz Events System and the Event Editor (see Setup > More Options > Events) supporting the interpreter:
Blockly, Lua, Python, dzVents (next generation Lua Scripting)

Some Notes regarding the various Domoticz Interpreters.

Blockly
NOT USED – all events via dzVents. For learning & testing developed a script.

Lua
NOT USED – all events via dzVents. For learning & testing developed a script.

Lua dzVents
Target is using the next generation Lua Scripting dzVents - Domoticz Easy Events – with the Domoticz build-in Editor.
Important to read this first.
Note
The Lua editor is context sensitive, prompt with auto-complete options and show common errors to help with debugging. The editor theme can be changed by pressing the control and comma keys at the same time. Domoticz will save the scheme when events are saved.

Python
In favour to program using Python instead Lua, **BUT** the new **dzVents** is so great, that for event scripting will use dzVents.
For building Domoticz PlugIns will use Python.
Note
The Python Event System uses Python3 (same as Python Plugin System). Events are enabled along with plugins, it's no longer possible to enable one without the other.

Event Execution Order

The order of events being executed, is:
1. file scripts - stored in the folder /home/pi/domoticz/scripts/…
2. database scripts - stored in tables [EventMaster] and [EventRules]
3. on/off action script - explore usage & storage

Important (taken from Domoticz Wiki)

Scripts that written on the file system and inside Domoticz using the internal web-editor share the same namespace. If there are two scripts with the same name, only the file system script will be used – this is logged.

Event Scheduling & Trigger

The scheduling is fixed in domoticz.
- ALL "device" scripts (scripts with "device" in their name) run everytime ANY device status changes.
- ALL "time" scripts (scripts with "time" in their name) run every MINUTE.

So far, have been using file scripts but considering moving to database solution only for device events.

For event triggerd once per day, target to use crontab with Python scripts.

Event Database Tables

The events which are created using the Domoticz Event Editor, are stored in tables [EventMaster] and [EventRules]

```
CREATE TABLE [EventMaster] ([ID] INTEGER PRIMARY KEY,  [Name] VARCHAR(200) NOT
NULL, [XMLStatement] TEXT NOT NULL, [Status] INTEGER DEFAULT 0, [Interpreter]
VARCHAR(10) DEFAULT 'Blockly', [Type] VARCHAR(10) DEFAULT 'All');
```

```
CREATE TABLE [EventRules] ([ID] INTEGER PRIMARY KEY, [EMID] INTEGER, [Conditions]
TEXT NOT NULL, [Actions] TEXT NOT NULL, [SequenceNo] INTEGER NOT NULL, FOREIGN KEY
(EMID) REFERENCES EventMaster(ID));
```

Event Script Viewer

Developed, with Lazarus, an application to view the dzVents scripts.
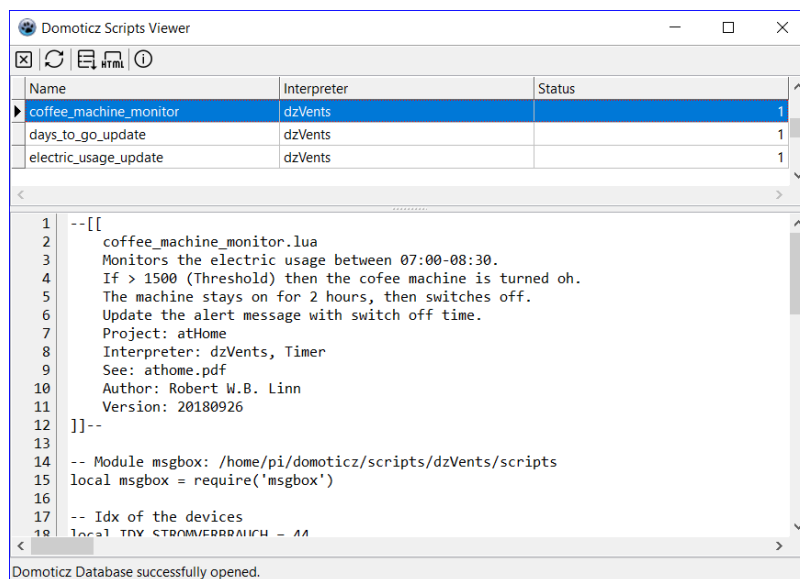
Event Development

Whilst creating scripts using the Event Editor, two Browser Tabs are open:

- Domoticz Event Editor – to create/modify scripts using Ace Editor, to activate/deactivate a script,
- Domoticz Log (Setup > Log > Tab All) – to check script OK after save.
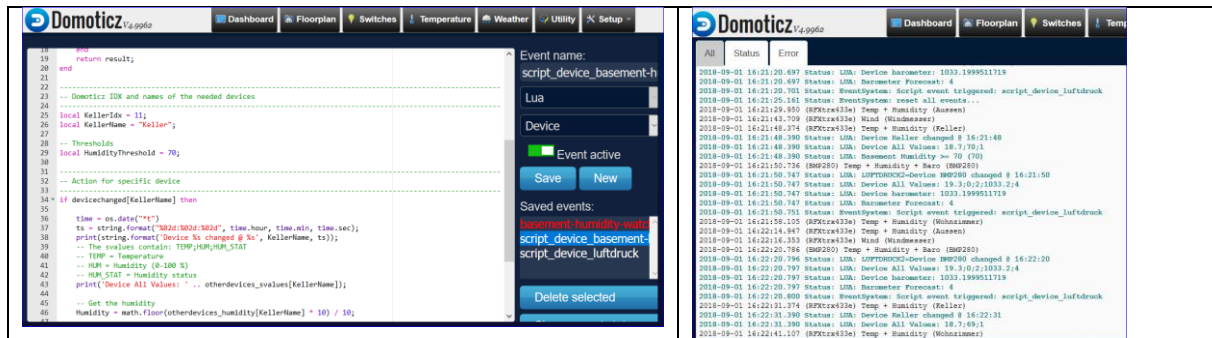
Event Sample Basement Humidity Monitor

Purpose

To explore the various interpreter using the same event to monitor a device:
Device: Keller (idx=11)
Property: humidity
Rule:
If the Basement Humidity is >= 70 (threshold), write a message to the Domoticz Log.

Note
For the sample scripts, added to the Event Names, a suffix as an interpreter indicator:
_d (Blockly), _l (Lua), _d (dzVents), _p (Python).

Blockly

Event Properties: basement_humidity_monitor_b, Blockly, All.
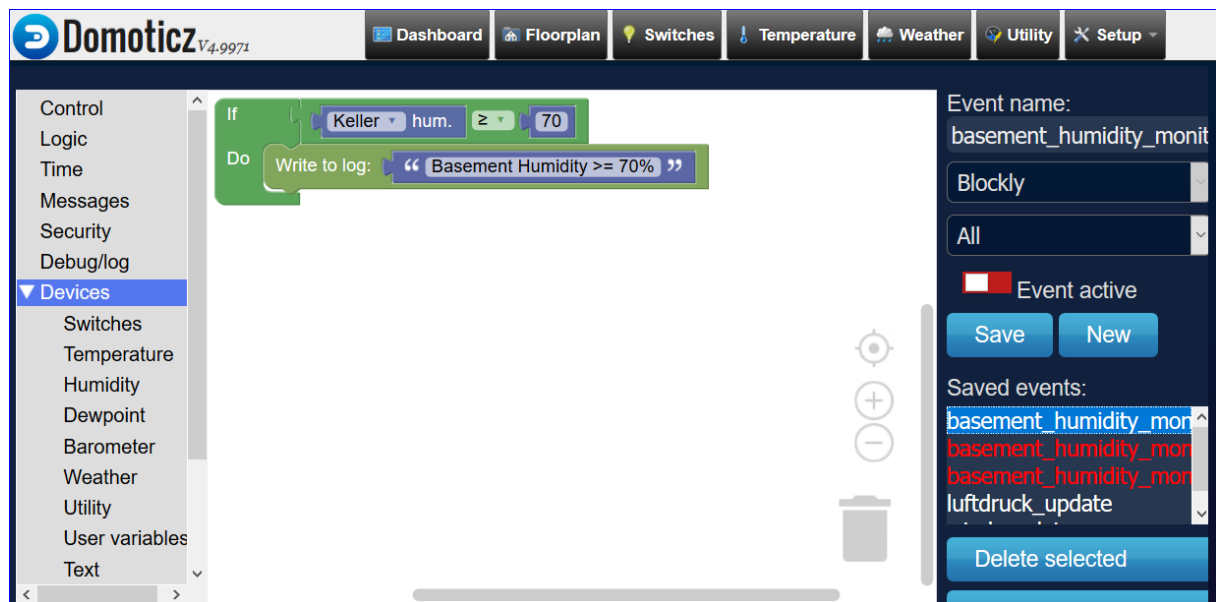
Domoticz Log Entry

```
2018-09-01 10:32:04.478 EventSystem: Event triggered: basement_humidity_monitor_b
2018-09-01 10:32:04.478 Status: Basement Humidity >= 70%
```

Lua

Event Properties: basement_humidity_monitor_l, Lua, Device.

Script

```lua
commandArray = {}

-- Domoticz IDX and names of the needed devices
local KellerIdx = 11;
local KellerName = "Keller";

local HumidityThreshold = 70;

if devicechanged[KellerName] then
        time = os.date("*t")
        ts = string.format("%02d:%02d:%02d", time.hour, time.min, time.sec);
        print(string.format('Device %s changed @ %s', KellerName, ts));
        print('Device All Values: ' .. otherdevices_svalues[KellerName]);
        -- The svalues contain: TEMP;HUM;HUM_STAT
        -- TEMP = Temperature
        -- HUM = Humidity (0-100 %)
        -- HUM_STAT = Humidity status
        -- Get the humidity
        Humidity = math.floor(otherdevices_humidity[KellerName] * 10) / 10;

    -- Check if the humidity is equal or above threshold - notify
    if Humidity >= HumidityThreshold then
        print('Basement Humidity >= ' .. HumidityThreshold .. ' (' .. Humidity .. ')');
    end
end

return commandArray
```

Domoticz Log Entry

Sample Script – Error

```
2018-09-01 13:20:29.442 Status: LUA: Device Keller changed @ 13:20:29
2018-09-01 13:20:29.442 Status: LUA: Device All Values: 18.7;74;3
2018-09-01 13:20:29.442 Error: EventSystem: in basement-humiditymonitor_l:
[string "-- script_device_basement-humidity..."]:41: bad argument #2 to 'format'
(number expected, got nil)
```

Caused by using wrong variable name, i.e. humidity instead of Humidity.

Sample Script – OK

```
2018-09-01 13:43:25.433 Status: LUA: Device Keller changed @ 13:43:25
2018-09-01 13:43:25.433 Status: LUA: Device All Values: 18.6;75;3
2018-09-01 13:43:25.433 Status: LUA: Basement Humidity >= 70 (75)
```

Lua dzVents

Event Properties: basement_humidity_monitor_d, dzVents, Device.

Script

```
local HumidityThreshold = 70;
return {
        on = {
                devices = {
                        'Keller'
                }
        },
        execute = function(domoticz, device)
            if (device.name == 'Keller' and device.humidity >= HumidityThreshold) then
                domoticz.log('Basement Humidity >= ' .. HumidityThreshold .. ' (' ..
device.humidity .. ')', domoticz.LOG_INFO)
            end
end
}
```

The script is stored in:

```
/home/pi/domoticz/scripts/dzVents/generated_scripts/basement_humidity_monitor_d.lua
```

With owner root.

Domoticz Log Entry

Sample Script – Error

```
2018-09-01 19:05:55.584 Status: dzVents: Error (2.4.7): error loading module '
basement_humidity_monitor_d' from file
'/home/pi/domoticz/scripts/dzVents/generated_scripts/
basement_humidity_monitor_d.lua':
```

```
2018-09-01 19:05:55.584 .../generated_scripts/ basement_humidity_monitor_d.lua:12:
'then' expected near ')'
```

Caused by using double)) prior then.

Sample Script – OK

```
2018-09-01 19:09:42.474 Status: dzVents: Write file:
/home/pi/domoticz/scripts/dzVents/generated_scripts/
basement_humidity_monitor_d.lua
2018-09-01 19:11:39.449 Status: dzVents: Info: Handling events for: "Keller",
value: "18.6;70;1"
2018-09-01 19:11:39.449 Status: dzVents: Info: ------ Start internal script:
script_device_basement_humidity_2: Device: "Keller (RFXtrx433e)", Index: 11
2018-09-01 19:11:39.449 Status: dzVents: Info: Basement Humidity >= 70 (70)
2018-09-01 19:11:39.449 Status: dzVents: Info: ------ Finished
basement_humidity_monitor_d
```

Python

Event Properties: basement_humidity_monitor_p, Python, Device.

Script

```
"""
Script: basement_humidity_monitor_p
project: AtHome
Monitor the basement humidity if exceeds threshold and write to Domoticz log.
@interpreter: Python, Device
@see: readme.txt
@author: Robert W.B. Linn
@version: 20180902
-------------------------------------------------------------------------------
"""

# Imports
import domoticz
import DomoticzEvents as DE

# Devices
KellerIdx = 11
KellerName = 'Keller'

# Set the threshold to monitor the humidity (%RH)
HumidityThreshold = 70

if DE.changed_device_name == KellerName:
    # Test Log
    # 2018-09-02 11:32:46.085 Python: Changed: ID: 11 Name: Keller, Type: 82, subType: 7,
switchType: 0, s_value: 18.7;75;3, n_value: 0, n_value_string: 18.7;75;3,
last_update_string: 2018-09-02 11:32:46
    DE.Log("Python: Changed: " + DE.changed_device.Describe())
    # Test s_value
    # 2018-09-02 11:32:46.085 18.7;75;3
    svalue = DE.Devices[KellerName].s_value
    DE.Log(svalue)
    # Get the humidity, which is at the 2nd entry of the svalue, i.e. 75
    # Split the svalue by ;. The result is a list with length 3,
data[0]=T,data[1]=H,data[2]=Comfort
    data = svalue.split(";")
```

```
    # Log all 3 entries
    # for temp in data:
    #     DE.Log(temp)
    # Get the :
    if len(data) == 3:
        Humidity = int(data[1])
        if Humidity >= HumidityThreshold:
            # Log that the humidity is above threshold. The log string uses new string
formatting
            DE.Log('Basement Humidity >= {}%RH ({}%RH)'.format(HumidityThreshold,
Humidity))
```

Domoticz Log Entry

Sample Script – Error

```
…
2018-09-02 11:58:34.884 Status: EventSystem: Script event triggered:
luftdruck_update
2018-09-02 11:58:34.107 Error: EventSystem: Failed to execute python event script
"basement_humidity_monitor_p"
2018-09-02 11:58:34.108 Error: EventSystem: Traceback (most recent call last):
2018-09-02 11:58:34.108 Error: EventSystem: File "<string>", line 41, in <module>
2018-09-02 11:58:34.108 Error: EventSystem: IndexError: tuple index out of range
```

Caused by using wrong string format tuples, i.e. {1} ({2}) instead of 0,1 or leave blank {} ({}).

Sample Script – OK

```
2018-09-02 12:01:26.253 Python: Changed: ID: 11 Name: Keller, Type: 82, subType:
7, switchType: 0, s_value: 18.7;74;3, n_value: 0, n_value_string: 18.7;74;3,
last_update_string: 2018-09-02 12:01:26
2018-09-02 12:01:26.253 18.7;74;3
2018-09-02 12:01:26.253 Basement Humidity >= 70%RH (74%RH)
```

Python with Device Update

Extending the previous Python script

Update the Virtual Sensor "Info Message" (idx=52) (Type: General, Text) with the message logged, if the humidity exceeds the threshold.

| 52 | VirtualSensors | 00082051 | 1 | Info Message | General | Text | Basement Humidity >= 70%RH (78%RH) |

Result

```
Info Message
         Basement Humidity >= 70%RH (77%RH)
Text     Last Seen: 2018-09-02 19:47:15
         Type: General, Text

   ⭐  Log  Edit
```
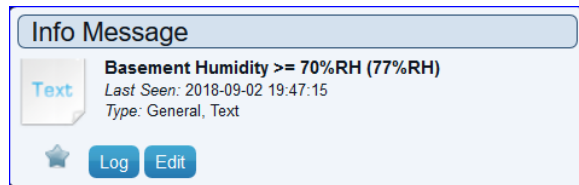
Event Properties: basement_humidity_monitor2_p, Python, Device.

Script

```python
"""
script: basement_humidity_monitor2_p
project: AtHome
Monitor the basement humidity (device=Keller) if exceeds threshold, write to Domoticz log
and update the virtual sensor text (device=Info Message)
@interpreter: Python, Device
@see: readme.txt
@author: Robert W.B. Linn
@version: 20180902
---------------------------------------------------------------------------
"""

# Imports
import domoticz
import DomoticzEvents as DE
from urllib.request import Request, urlopen
from urllib.error import URLError, HTTPError

# Define the domoticz server ip with port!
domoticzserver="localhost:8080"

# Devices
KellerIdx = 11
KellerName = 'Keller'
TextInfoIdx = 52
TextInfoName = 'Info Message'

# Set the threshold to monitor the humidity (%RH)
HumidityThreshold = 70

def domoticzrequest (url):
    """Send update request to domoticz
    :url: Domoticz JSON/API url
    :result: Result is a json object
    """
    req = Request(url)
    try:
        response = urlopen(req)
    except HTTPError as e:
```

```
        # print('Server couldn\'t fulfill the request.')
        # print('Error code: ', e.code)
        return "Error fulfilling the request." + e.code
    except URLError as e:
        # print('Failed to reach a server.')
        # print('Reason: ', e.reason)
        return "Error reaching the server." + e.code
    else:
        # everything is fine
        return response.read().decode('utf-8')


if DE.changed_device_name == KellerName:
    # Test Log
    # 2018-09-02 11:32:46.085 Python: Changed: ID: 11 Name: Keller, Type: 82, subType: 7,
switchType: 0, s_value: 18.7;75;3, n_value: 0, n_value_string: 18.7;75;3,
last_update_string: 2018-09-02 11:32:46
    DE.Log("Python: Changed: " + DE.changed_device.Describe())
    # Test s_value
    # 2018-09-02 11:32:46.085 18.7;75;3
    svalue = DE.Devices[KellerName].s_value
    DE.Log(svalue)
    # Get the humidity, which is at the 2nd entry of the svalue, i.e. 75
    # Split the svalue by ;. The result is a list with length 3,
data[0]=T,data[1]=H,data[2]=Comfort
    data = svalue.split(";")
    # Log all 3 entries
    # for temp in data:
    #     DE.Log(temp)
    # Get the humidity at pos 2 which is index 1 :
    if len(data) == 3:
        Humidity = int(data[1])
        if Humidity >= HumidityThreshold:
            # Log that the humidity is above threshold. The log string uses new string
formatting
            svalueinfo = 'Basement Humidity >= {}%RH ({}%RH)'.format(HumidityThreshold,
Humidity)
            DE.Log(svalueinfo)

            # Update the Virtual Sensor Info_Message
            # This is not working - not sure why, might be only in plugins
            # DE.Command(TextInfoName, svalueinfo)

            # Require to escape the message string including spaces
            svalueinfo = urllib.parse.quote_plus(svalueinfo)
            # Build the url. Example:
            #
http://localhost:8080/json.htm?type=command&param=udevice&idx=52&nvalue=0&svalue=Basement+H
umidity+%3E%3D+70%25RH+%2877%25RH%29
            url = "http://" + domoticzserver + "/json.htm?type=command&param=udevice&idx="
+ str(TextInfoIdx) + "&nvalue=0&svalue=" + svalueinfo
            # DE.Log(url)
            ret = domoticzrequest(url)
            # Log the return . which is a JSON string {"status" : "OK","title" : "Update
Device"}
            # DE.Log(ret)
```

Domoticz Log Entry

Sample Script – OK

```
2018-09-02 19:53:42.963 Python: Changed: ID: 11 Name: Keller, Type: 82, subType:
7, switchType: 0, s_value: 18.8;78;3, n_value: 0, n_value_string: 18.8;78;3,
last_update_string: 2018-09-02 19:53:42
2018-09-02 19:53:42.963 18.8;78;3
```

```
2018-09-02 19:53:42.963 Basement Humidity >= 70%RH (78%RH)
```

dzVents

The event system dzVents is the next generation of Domoticz Events.

As a general point, be very secure in writing the scripts, also when making changes, i.e.
ensure all variables have the same name, do not mix local vars with the same name etc.

Device Properties

To analyse the properties of a device, use device.dump(), to then select from the list.

Script

This example checks if the defined devices (see idx of the devices) have changed, to then
dump the properties but also show the device state property.

```
-- Idx of the devices
local IDXMEMORYUSAGE = 1;
local IDXHDDUSAGE = 3;
local IDXTEMPERATURE = 4;
local IDXCONTROLMSG = 52;

return {
  on = {
    devices = {
      IDXMEMORYUSAGE,
      IDXHDDUSAGE,
      IDXIDXTEMPERATURE
    }
  },
  execute = function(domoticz, device)
    domoticz.log('RPi Monitor Device ' .. device.name .. ',' .. device.idx .. ' was
changed:' .. device.state .. '/' .. device.rawData[1], domoticz.LOG_INFO)
  end
}
```

Domoticz Log Entry

```
2018-09-05 10:01:10.305 Status: dzVents: Info: Handling events for: "RPi Memory
Usage", value: "24.20"
2018-09-05 10:01:10.305 Status: dzVents: Info: ------ Start internal script:
rpi_monitor: Device: "RPi Memory Usage (Motherboard)", Index: 1
2018-09-05 10:01:10.305 Status: dzVents: Info: RPi Monitor Device RPi Memory
Usage,1 was changed:24.20/24.20
2018-09-05 10:01:10.305 Status: dzVents: Info: ------ Finished rpi_monitor
```

Property List

Example properties for device RPi Memory Usage:

```
ruleIsAtSunrise()                 utils:                           hardwareName: Motherboard
wday: 4                           fileExists()                     protected: false
isUTC: false                      LOG_MODULE_EXEC_INFO: 2          hardwareType: Motherboard
ruleIsAfterCivilTwilightEnd()     LOG_ERROR: 1                     sensors
hour: 9                           toJSON()                         lastUpdate: 2018-09-05 09:47:49
ruleIsAfterCivilTwilightStart()   log()                            subType: Percentage
ruleIsBeforeCivilTwilightStart()  rgbToHSB()                       baseType: device
current:                          LOG_INFO: 3                      changed: true
isdst: true                       osExecute()                      batteryLevel: 255
min: 49                           urlEncode()                      switchType: On/Off
year: 2018                        print()                          switchTypeValue: 0
day: 5                            fromJSON()                       deviceType: General
yday: 248                         LOG_FORCE: 0.5                    rawData:
sec: 9                            LOG_DEBUG: 4                      1: 24.15
month: 9                          updatePressure()                 id: 1
wday: 4                           deviceId: 0000044C               signalLevel: 12
hour: 9                           updateRain()                     description:
secondsAgo: 80                    close()                          lastLevel: 0
ruleIsOnDate()                    updateP1()                       name: RPi Memory Usage
milliSeconds: 0                   _adapters:                       timedOut: false
millisecondsAgo: 80896            1: Percentage device adapter     switchType: On/Off
ruleIsInWeek()                    disarm()                         setWhiteMode()
dDate: 1536133669                 startPlaylist()                  deviceType: General
ruleIsAtDayTime()                 changed: true                    switchTypeValue: 0
ruleIsAfterSunrise()              setVolume()                      protected: false
day: 5                            stop()                           isScene: false
isdst: true                       _data:                           state: 24.15
min: 47                           deviceID: 0000044C               hardwareTypeValue: 23
minutesAgo: 1                     data:                            idx: 1
                                  unit: 1                          isTimer: false
                                  icon: hardware                   signalLevel: 12
                                  hardwareTypeValue: 23            bState: false
                                  _state: 24.15                    isVariable: false
                                  _nValue: 0                       hardwareType: Motherboard
                                  hardwareID: 1                    sensors
```

External Modules

Purpose

To use external modules in dzVents scripts, which are created with the Events editor.

Example

Update the virtual device named "Alert Message" (Type General, SubType Alert, Idx=55) by using a function defined in external module **msgbox.lua**.

```
local message = device.name .. ' switched OFF ' -- .. msg.isnowdatetime(domoticz)
msgbox.alertmsg(domoticz, domoticz.ALERTLEVEL_GREEN, message)
```
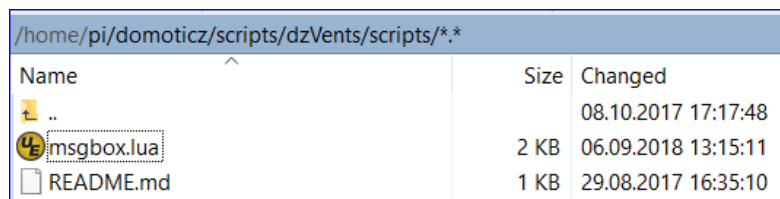
Module Access

The module "msgbox.lua" MUST be in folder:

```
/home/pi/domoticz/scripts/dzVents/scripts
```

To access functions from the module, use the syntax **msgbox.functionname**, when the module is defined like:

```
local msgbox = require('msgbox')
```

Example (screenshot from WinSCP)

Module Script

The script defines date & time functions and functions to update a text device & alert device. The idx of the text device is set in the external module, for the alert device as a user variable. This is to show both ways – best would be to have both device idx as user variables to be flexible in assigning devices.
A domoticz object is required to access properties & methods.

```
msgbox.lua: /home/pi/domoticz/scripts/dzVents/scripts
```

```lua
-- msgbox.lua
-- Update the virtual devices to display a control (Text Sensor) or alert message
(Alert Sensor).
-- Requires a user variable IDX_ALERTMSG for the Alert Sensor
-- Robert W.B. Linn
-- 20180906

local msgbox = {};

local IDX_CONTROLMSG = 52

-- Get the current date & time from the domoticz instance with time object
-- Return datetime now, i.e. 2018-09-06 09:09:00
function msgbox.isnowdatetime(domoticz)
    return domoticz.time.rawDate .. ' ' .. domoticz.time.rawTime
end

-- Get the current date from the domoticz instance with time object
-- Return date now, i.e. 2018-09-06
function msgbox.isnowdate(domoticz)
    return domoticz.time.rawDate
end

-- Get the current time from the domoticz instance with time object
-- Return time now, i.e. 09:09:00
function msgbox.isnowtime(domoticz)
    return domoticz.time.rawTime
end

-- Update the alert message with level.
function msgbox.alertmsg(domoticz, level, msg)
 domoticz.devices(domoticz.variables('IDX_ALERTMSG').value).updateAlertSensor(leve
l, msg)
end

-- Update the control message.
function msgbox.controlmsg(domoticz, msg)
 domoticz.devices(IDX_CONTROLMSG).updateText(msg)
end

return msgbox
```

dzVents Script

The script hue_control2.lua, created with the Event editor, references the external module msgbox.lua as msgbox.
If the script is set to enabled, the scripts is stored in folder

```
/home/pi/domoticz/scripts/dzVents/generated_scripts
```

```
-- Module msgbox.lua: /home/pi/domoticz/scripts/dzVents/scripts
-- Access function using msgbox.functionname
local msgbox = require('msgbox')

-- Idx of the devices
local IDX_HUE_MAKELAB = 118

return {
    -- active = true,
  on = {
    devices = {
      IDX_HUE_MAKELAB
    }
  },
  execute = function(domoticz, device)
    if (device.state == 'On') then
      device.switchOff()
      local message = device.name .. ' = OFF ' -- .. msg.isnowdatetime(domoticz)
      msgbox.alertmsg(domoticz, domoticz.ALERTLEVEL_GREEN, message)
      msgbox.controlmsg(domoticz, message)
      domoticz.log('OK', domoticz.LOG_INFO)
    end
  end
}
```

Folder Scripts

Scripts located in folder

```
/home/pi/domoticz/scripts/dzVents/scripts
```

are executed every minute, if these are not modules.

```
-- Lua script executed every minute in the dzVents scripts folder
print(os.date("today is %A, at %x"))
```

Domoticz Log Entry

```
-- 2018-09-10 12:15:00.750 Status: dzVents: today is Monday, at 09/10/18
```

MQTT

Purpose

To enable subscribing and publishing MQTT messages (see Wiki).
Ensure also to lookup the reference Domoticz API/JSON Url's.

Install

Raspberry Pi

Install mosquitto on the Raspberry Pi.

Open Terminal and run

```
sudo apt-get install mosquitto mosquitto-clients
```

Check if mosquitto is running, by invoking

```
mosquitto -h
```

Output:

```
mosquitto version 1.4.10 (build date Fri, 22 Dec 2017 08:19:25 +0000)
mosquitto is an MQTT v3.1 broker.
```

Domoticz

Domoticz Add Hardware
Setup > Hardware

Property	Value
Type	MQTT Client Gateway with LAN Interface
Name	MQTT
Remote Address	rpi-domoticz-ip
Port	1883
Publish Topic	Out
Username	Empty
Password	Empty

Check the Domoticz Log Setup > Log

```
2018-08-27 12:21:12.126 Status: MQTT: Connecting to rpi-domoticz-ip:1883
2018-08-27 12:21:12.228 Status: MQTT: connected to: rpi-domoticz-ip:1883
2018-08-27 12:21:12.328 Status: MQTT: Subscribed
```

Domoticz Topics

Subscribe to Topics from Domoticz	domoticz/out
Publish a Topic to Domoticz	domoticz/in

Mosquitto

Test MQTT messaging using mosquitto commands from a terminal.

Subscribe

Command:

```
mosquitto_sub -t 'domoticz/out'
```

Some sample messages, in JSON format, published by Domoticz in JSON format.
Source from the devices provided by the Hardware Device Motherboard:

For each of the messages the topic is domoticz/out and the payload is a JSON string.
To get a value from a key, the JSON string needs to be parsed.

CPU Usage

```
{
   "Battery" : 255,"RSSI" : 12,"description" : "","dtype" : "General","id" : "0000044D",
   "idx" : 1, "name" : "CPU_Usage", "nvalue" : 0, "stype" : "Percentage", "svalue1" :
"0.19",
   "unit" : 1
}
```

CPU Temperature

```
{
    "Battery" : 255, "RSSI" : 12, "description" : "", "dtype" : "Temp", "id" : "1",
    "idx" : 2, "name" : "Internal Temperature", "nvalue" : 0, "stype" : "LaCrosse TX3",
    "svalue1" : "42.4", "unit" : 1
}
```

Publish

Let's change the text for a Virtual Device Type General, Text with Idx 9, Name "Biotonne Pi" from "Hello World" to "Moin Moin".

		Idx ^	Hardware ^	ID ^	Unit ^	Name ^	Type ^	SubType ^	Data
☐		9	Virtual Devices	00082009	1	Biotonne Pi	General	Text	Hello World

Command:

```
mosquitto_pub -t 'domoticz/in' -m '{"idx":9,"nvalue":0,"svalue":"Moin Moin"}'
```

There is no terminal output from this command.
Possible checks if the command was successful:

Check Domoticz Widget has changed (Tab Utility)	**Biotonne Pi** Text **Moin Moin** *Last Seen: 2018-08-27 13:28:57* *Type: General, Text* ☆ Log Edit
Check Domoticz Log entry (Tab Setup > Log)	`2018-08-27 15:57:26.792 MQTT: Topic: domoticz/in, Message: {"idx":9,"nvalue":0,"svalue":"Moin Moin"}`

MQTT Hints

Add the host to the mosquitto command:

```
-h localhost
```

Python

To be able to use MQTT from python, the paho client library is required.

Paho Client Library

To install, download the client and run the Python setup script to install the client.
There are two ways:
- cloning from github, which will install the client with source code and examples
- using pip and pip3

GitHub

Run from a terminal

```
mkdir /home/pi/python
git clone https://github.com/eclipse/paho.mqtt.python.git
cd paho.mqtt.python
sudo python setup.py install
```

Output

```
pi@DomoticzDev:~ $ mkdir python
pi@DomoticzDev:~ $ git clone https://github.com/eclipse/paho.mqtt.python.git
Cloning into 'paho.mqtt.python'...
remote: Counting objects: 2932, done.
remote: Compressing objects: 100% (66/66), done.
remote: Total 2932 (delta 27), reused 85 (delta 24), pack-reused 2834
Receiving objects: 100% (2932/2932), 835.76 KiB | 447.00 KiB/s, done.
Resolving deltas: 100% (1487/1487), done.
pi@DomoticzDev:~ $ cd paho.mqtt.python
pi@DomoticzDev:~/paho.mqtt.python $ sudo python setup.py install
running install
…
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
```

Pip (Python2) & Pip3 (Python3)

Run from a terminal with sample output

```
$ pip install paho-mqtt
Collecting paho-mqtt
  Downloading
https://files.pythonhosted.org/packages/2a/5f/cf14b8f9f8ed1891cda893a2a7d1d6fa23de
2a9fb4832f05cef02b79d01f/paho-mqtt-1.3.1.tar.gz (80kB)
    100% |81kB 989kB/s
Building wheels for collected packages: paho-mqtt
  Running setup.py bdist_wheel for paho-mqtt ... done
  Stored in directory:
/home/pi/.cache/pip/wheels/38/ca/67/86c7e4acc659ce5ab74cbb8cc38de50c90ed4f827133e3
6994
Successfully built paho-mqtt
Installing collected packages: paho-mqtt
Successfully installed paho-mqtt-1.3.1
```

```
$ pip3 install paho-mqtt
Collecting paho-mqtt
  Downloading https://www.piwheels.org/simple/paho-mqtt/paho_mqtt-1.3.1-py3-none-
any.whl (57kB)
    100% |61kB 437kB/s
Installing collected packages: paho-mqtt
Successfully installed paho-mqtt-1.3.1
```

Subscribe Sample Script

This sample script connects to the MQTT broker running on the Raspberry Pi, subscribes to the Domoticz Out topic ('domoticz/out) and filters out the value for the device with idx 1, which is the Raspberry Pi motherboard CPU usage in percentage.

		Idx ^	Hardware ^	ID ^	Unit ^	Name	^	Type ^	SubType ^	
☐		1	Motherboard	0000044D	1	CPU_Usage		General	Percentage	0.18%

```python
#!/usr/bin/env python

import time
import paho.mqtt.client as mqtt
import json

broker= "localhost"
topic = "domoticz/out"
idx = 1

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

def on_message(client, userdata, message):
    print("message received " ,str(message.payload.decode("utf-8")))
    print("message topic=",message.topic)
    print("message qos=",message.qos)
    print("message retain flag=",message.retain)
    json_string = str(message.payload.decode("utf-8"))
    getidxvalue(json_string, idx)

def on_log(client, userdata, level, buf):
    print("log: ",buf)

# JSON Parsing
def getidxvalue(json_string, idx):
    parsed_json = json.loads(json_string)
    parsed_idx = int(parsed_json['idx'])
    if parsed_idx == idx:
        parsed_name = str(parsed_json['name'])
        parsed_value = str(parsed_json['svalue1'])
        print(parsed_idx, parsed_name, parsed_value)

print("Creating new instance")
client = mqtt.Client("P1")
client.on_connect = on_connect
client.on_message = on_message
client.on_log = on_log
print("Connecting to the broker", broker)
client.connect(broker, 1883, 60)
time.sleep(4)
client.loop_forever()
```

Output Sample with lots of prints to show the steps & actions

```
$ python3 mqtttest.py
creating new instance
connecting to broker
log:  Sending CONNECT (u0, p0, wr0, wq0, wf0, c1, k60) client_id=b'P1'
log:  Received CONNACK (0, 0)
Connected with result code 0
log:  Sending SUBSCRIBE (d0) [(b'domoticz/out', 0)]
log:  Received SUBACK
log:  Received PUBLISH (d0, q0, r0, m0), 'domoticz/out', ...  (229 bytes)
message received  {
   "Battery" : 255,
   "RSSI" : 12,
   "description" : "",
   "dtype" : "General",
   "id" : "0000044D",
   "idx" : 1,
   "name" : "CPU_Usage",
   "nvalue" : 0,
   "stype" : "Percentage",
   "svalue1" : "0.77",
   "unit" : 1
}

message topic= domoticz/out
message qos= 0
message retain flag= 0
1 CPU_Usage 0.77

message topic= domoticz/out
message qos= 0
message retain flag= 0
1 CPU_Usage 0.33
^CTraceback (most recent call last):
  File "mqtttest.py", line 54, in <module>
    client.loop_forever()
  File "/home/pi/.local/lib/python3.5/site-packages/paho/mqtt/client.py", line
1481, in loop_forever
    rc = self.loop(timeout, max_packets)
  File "/home/pi/.local/lib/python3.5/site-packages/paho/mqtt/client.py", line
988, in loop
    socklist = select.select(rlist, wlist, [], timeout)
KeyboardInterrupt
```

Publish Sample Script

This sample script connects to the MQTT broker running on the Raspberry Pi and publishes text to a Virtual Device with idx 9, name Biotonne Pi, Type General, SubType Text

9	Virtual Devices	00082009	1	Biotonne Pi	General	Text	Moin Moin

The script sets the text from "Moin Moin" to "Hello Again".

Python3 Script

```python
#!/usr/bin/env python

import time
import paho.mqtt.client as mqtt

version="MQTT Test v20180827"
broker= "localhost"
topic = "domoticz/in"
idx = 9
payload='{"command": "udevice", "idx": %s, "svalue": "Hello Again"}' % (idx)

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

def on_message(client, userdata, message):
    print("message received " ,str(message.payload.decode("utf-8")))
    print("message topic=",message.topic)
    print("message qos=",message.qos)
    print("message retain flag=",message.retain)

def on_log(client, userdata, level, buf):
    print("log: ",buf)

print(version)
print("Creating new instance")
client = mqtt.Client("P1")
client.on_connect = on_connect
client.on_message = on_message
client.on_log = on_log
print("Connecting to the broker", broker)
client.connect(broker, 1883, 60)
time.sleep(4)
print("Publishing message to topic",topic, "with payload",payload)
domoticz/in, Message: {"command": "udevice", "idx": 9, "svalue": "Hello Again"}
client.publish(topic,payload)
time.sleep(4)
client.loop_stop()
```
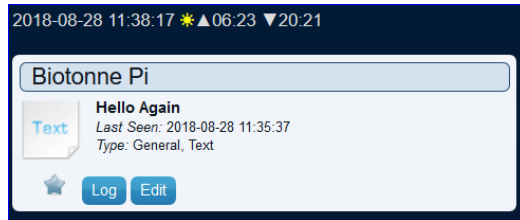
Output

```
pi@DomoticzDev:~/python $ python3 mqtt-pub-test.py
MQTT Test v20180827
Creating new instance
Connecting to the broker localhost
log:  Sending CONNECT (u0, p0, wr0, wq0, wf0, c1, k60) client_id=b'P1'
```

```
Publishing message to topic domoticz/in with payload {"command": "udevice", "idx":
9, "svalue": "Hello Again"}
log:  Sending PUBLISH (d0, q0, r0, m1), 'b'domoticz/in'', ... (57 bytes)
```

Widget with updated text

SQL

Purpose

To execute CRUD operations on the Domoticz SQLite Database located /home/pi/Domoticz/domoticz.db.

<TODO> Identify and build solution to manage entries, i.e. select or delete data esp. from the Measurement Tables.

Thoughts:
- using Python3 scripts triggered by a Switch or Node-RED with Dashboard UI (and input field)
- using Python3 with guizero – simple UI

HINTS
- To analyse the database, external tools like the SQLiteSpy are useful helpers.
- For tests, the Domoticz database has been exported from the Raspberry Pi to the Development Device (Setup > Settings > Backup/Restore > Backup Database).
- For final solution, need to ensure Domoticz is stopped first, then perform CRIUD operations on the Domoticz database and start Domoticz afterwards.

SQL Commands

For testing of SQL statements on the Domoticz database, used

- the Domoticz database "Pinneberg.db" which has been exported via Setup > Settings > Backup/Restore > Backup Database
  The database "Pinneberg.db" has been downloaded from the Raspberry Pi to the Downloads folder of the Development Device.
- the table tables [DeviceStatus] and [Temperature] of the Domoticz database "Pinneberg.db".
- the tool SQLiteSpy.

Tables

The tables holding measurement data are
'MultiMeter','MultiMeter_Calendar','Percentage','Percentage_Calendar','Rain','Rain_Calendar', 'Temperature','UV','UV_Calendar','Wind','Wind_Calendar'

<TODO> need to check if this changes depending Domoticz version.

SQL Create Statements for the tables [DeviceStatus] and [Temperature]

```
CREATE TABLE [DeviceStatus] ([ID] INTEGER PRIMARY KEY, [HardwareID] INTEGER NOT
NULL, [DeviceID] VARCHAR(25) NOT NULL, [Unit] INTEGER DEFAULT 0, [Name]
VARCHAR(100) DEFAULT Unknown, [Used] INTEGER DEFAULT 0, [Type] INTEGER NOT NULL,
[SubType] INTEGER NOT NULL, [SwitchType] INTEGER DEFAULT 0, [Favorite] INTEGER
DEFAULT 0, [SignalLevel] INTEGER DEFAULT 0, [BatteryLevel] INTEGER DEFAULT 0,
```

```
[nValue] INTEGER DEFAULT 0, [sValue] VARCHAR(200) DEFAULT null, [LastUpdate]
DATETIME DEFAULT (datetime('now','localtime')),[Order] INTEGER BIGINT(10) default
0, [AddjValue] FLOAT DEFAULT 0, [AddjMulti] FLOAT DEFAULT 1, [AddjValue2] FLOAT
DEFAULT 0, [AddjMulti2] FLOAT DEFAULT 1, [StrParam1] VARCHAR(200) DEFAULT '',
[StrParam2] VARCHAR(200) DEFAULT '', [LastLevel] INTEGER DEFAULT 0, [Protected]
INTEGER DEFAULT 0, [CustomImage] INTEGER DEFAULT 0, [Description] VARCHAR(200)
DEFAULT '', [Options] TEXT DEFAULT null, [Color] TEXT DEFAULT NULL);

CREATE TABLE [Temperature] ([DeviceRowID] BIGINT(10) NOT NULL, [Temperature] FLOAT
NOT NULL, [Chill] FLOAT DEFAULT 0, [Humidity] INTEGER DEFAULT 0, [Barometer]
INTEGER DEFAULT 0, [DewPoint] FLOAT DEFAULT 0, [SetPoint] FLOAT DEFAULT 0, [Date]
DATETIME DEFAULT (datetime('now','localtime')));
```

Select

Purpose

To select device status and temperature data from the Domoticz database tables [DeviceStatus], [Temperature].

Python

```
#!/usr/bin/env python
import sqlite3
from sqlite3 import Error
import time
import os

version="SQL Test v20180828"
database="/home/pi/domoticz/domoticz.db"

def DatabaseExists(database):
    if not os.path.exists(database):
        print("Database %s not found." % (database))
        return False
    else:
        print("Database %s found." % (database))
        return True

def create_connection(db_file):
    """ create a database connection to the SQLite database
        specified by the db_file
    :param db_file: Database file
    :return: Connection object or None
    """
    try:
        conn = sqlite3.connect(db_file)
        return conn
    except Error as e:
        print(e)

    return None

def select_all_device_status(conn):
    """
    Query all rows in the tasks DeviceStatus
    :param conn: the Connection object
    :return:
    """
    print("Select Device Status")
    cur = conn.cursor()
    cur.execute("SELECT * FROM DeviceStatus")
    rows = cur.fetchall()
    for row in rows:
        print(row)

def select_all_temperatures(conn):
    """
    Query all rows in the tasks Temperature
    :param conn: the Connection object
    :return:
    """
    print("Select all Temperatures")
    cur = conn.cursor()
    cur.execute("SELECT * FROM Temperature")
    rows = cur.fetchall()
```

```python
    for row in rows:
        print(row)

def select_temperature_by_threshold(conn, threshold):
    """
    Query temperatures by threshold
    :param conn: the Connection object
    :param threshold:
    :return:
    """
    print("Select Temperature by Threshold %s" % (threshold))
    cur = conn.cursor()
    cur.execute("SELECT * FROM Temperature WHERE Temperature>?", (threshold,))
    rows = cur.fetchall()
    for row in rows:
        print(row)

def main():
    print(version)
    if DatabaseExists(database):
        print("Connecting to the database...",database)
        # create a database connection
        conn = create_connection(database)
        with conn:
            select_all_device_status(conn)
            select_all_temperatures(conn)
            select_temperature_by_threshold(conn, 41.0)

        conn.close()

if __name__ == '__main__':
    main()
```

Output data from a Domoticz Test Database

```
python3 sql-select-test.py
SQL Test v20180828
Database /home/pi/domoticz/domoticz.db found.
Connecting to the database... /home/pi/domoticz/domoticz.db
Select Device Status
(1, 4, '0000044D', 1, 'CPU_Usage', 0, 243, 6, 0, 0, 12, 255, 0, '0.33', '2018-08-
28 19:04:15', 1, 0.0, 1.0, 0.0, 1.0, '', '', 0, 0, 0, '', None, None)
(2, 4, '1', 1, 'Internal Temperature', 0, 80, 5, 0, 0, 12, 255, 0, '41.3', '2018-
08-28 19:03:35', 2, 0.0, 1.0, 0.0, 1.0, '', '', 0, 0, 0, '', None, None)
(3, 4, '0000044C', 1, 'Memory Usage', 0, 243, 6, 0, 0, 12, 255, 0, '16.82', '2018-
08-28 19:03:25', 3, 0.0, 1.0, 0.0, 1.0, '', '', 0, 0, 0, '', None, None)
(4, 4, '000005DC', 1, 'Process Usage', 0, 243, 31, 0, 0, 12, 255, 0, '20.5800',
'2018-08-28 19:03:25', 4, 0.0, 1.0, 0.0, 1.0, '', '', 0, 0, 0, '', '1;MB', None)
(5, 4, '0000044E', 1, 'HDD /media/pi/SETTINGS', 0, 243, 6, 0, 0, 12, 255, 0,
'1.50', '2018-08-28 19:04:35', 5, 0.0, 1.0, 0.0, 1.0, '', '', 0, 0, 0, '', None,
None)
(6, 4, '0000044F', 1, 'HDD /boot', 0, 243, 6, 0, 0, 12, 255, 0, '32.32', '2018-08-
28 19:04:35', 6, 0.0, 1.0, 0.0, 1.0, '', '', 0, 0, 0, '', None, None)
(7, 4, '00000450', 1, 'HDD /media/pi/data', 0, 243, 6, 0, 0, 12, 255, 0, '0.08',
'2018-08-28 19:04:35', 7, 0.0, 1.0, 0.0, 1.0, '', '', 0, 0, 0, '', None, None)
(8, 4, '00000451', 1, 'HDD /', 0, 243, 6, 0, 0, 12, 255, 0, '17.52', '2018-08-28
19:04:35', 8, 0.0, 1.0, 0.0, 1.0, '', '', 0, 0, 0, '', None, None)
(9, 2, '00082009', 1, 'Biotonne Pi', 1, 243, 19, 0, 0, 12, 255, 0, 'Hello Again',
'2018-08-28 11:40:52', 9, 0.0, 1.0, 0.0, 1.0, '', '', 0, 0, 0, '', None, None)
Select all Temperatures
(2, 40.8, 0.0, 0, 0, 0.0, 0.0, '2018-08-28 17:05:00')
(2, 41.9, 0.0, 0, 0, 0.0, 0.0, '2018-08-28 17:10:00')
(2, 41.3, 0.0, 0, 0, 0.0, 0.0, '2018-08-28 17:15:00')
(2, 40.8, 0.0, 0, 0, 0.0, 0.0, '2018-08-28 17:20:00')
Select Temperature by Threshold 41.0
(2, 41.9, 0.0, 0, 0, 0.0, 0.0, '2018-08-28 17:10:00')
(2, 41.3, 0.0, 0, 0, 0.0, 0.0, '2018-08-28 17:15:00')
```

Delete

Purpose

To delete data from the Domoticz database devices tables using where clause on field Date.

Sample SQL create statement for table [Temperature]

```
CREATE TABLE [Temperature] ([DeviceRowID] BIGINT(10) NOT NULL, [Temperature] FLOAT
NOT NULL, [Chill] FLOAT DEFAULT 0, [Humidity] INTEGER DEFAULT 0, [Barometer]
INTEGER DEFAULT 0, [DewPoint] FLOAT DEFAULT 0, [SetPoint] FLOAT DEFAULT 0, [Date]
DATETIME DEFAULT (datetime('now','localtime')));
```

The date is stored in field [Date] with format DATETIME DEFAULT
(datetime('now','localtime')).

Sample Entries for table [Temperature] for device with idx=28

```
DeviceRowID Temperature Chill Humidity Barometer DewPoint SetPoint Date
28     17.5  17.5  0     0       0.0   0.0    2018-08-27 14:30:00
28     18.9  18.9  0     0       0.0   0.0    2018-08-27 14:45:00
28     19.0  19.0  0     0       0.0   0.0    2018-08-27 14:50:00
…
```

SELECT statement with WHERE clause

```
SELECT * FROM [Temperature] WHERE [DeviceRowID] = 28 AND [Date] < date('2018-08-
28');
```
This works and returns all entries with a date < 2018-08-28.

DELETE statement with WHERE clause

Delete all entries from table [Temperature] with Date < 2018-08-28

```
DELETE FROM [Temperature] WHERE [DeviceRowID] = 28 AND [Date] < date('2018-08-
28');
```

Checked result:

```
SELECT * FROM [Temperature] WHERE [DeviceRowID] = 28;
```

The result is OK, i.e. all entries with a date < 2018-08-28 have been deleted.

Delete all data related to device
with idx=44 (Stromverbrauch)

Delete all data from Table Meter for the device with idx=44 (Stromverbrauch) with Select statement to check.

```
DELETE FROM Meter WHERE DeviceRowID=44

SELECT * FROM Meter WHERE DeviceRowID=44

DELETE FROM MultiMeter_Calendar WHERE DeviceRowID=44

SELECT * FROM MultiMeter_Calendar WHERE DeviceRowID=44
```

Python

Task: Delete temperature data.
Table: [Temperature]

<TODO> Add function, i.e. delete_temperature_by_date(conn, date).

```python
#!/usr/bin/env python

import sqlite3
from sqlite3 import Error
import time
import os

# Define globals
version="SQL Test v20180828"
# Domoticz database path
database="/home/pi/domoticz/domoticz.db"

def DatabaseExists(database):
    if not os.path.exists(database):
        print("Database %s not found." % (database))
        return False
    else:
        print("Database %s found." % (database))
        return True

def create_connection(db_file):
    """ create a database connection to the SQLite database
        specified by the db_file
    :param db_file: database file
    :return: Connection object or None
    """
    try:
        conn = sqlite3.connect(db_file)
        return conn
    except Error as e:
        print(e)

    return None

def select_all_temperatures(conn):
    """
    Query all rows in the tasks Temperature
    :param conn: the Connection object
```

```python
    :return:
    """
    print("Select all Temperatures")
    cur = conn.cursor()
    cur.execute("SELECT * FROM Temperature")

    rows = cur.fetchall()

    for row in rows:
        print(row)


def select_temperature_by_threshold(conn, threshold):
    """
    Query temperatures by threshold
    :param conn: the Connection object
    :param threshold:
    :return:
    """
    print("Select Temperature by Threshold %s" % (threshold))
    cur = conn.cursor()
    cur.execute("SELECT * FROM Temperature WHERE Temperature>?", (threshold,))

    rows = cur.fetchall()

    for row in rows:
        print(row)

def delete_temperature_by_threshold(conn, threshold):
    """
    Delete all temperatures by threshold
    :param conn: the Connection object
    :param threshold:
    :return:
    """
    print("Delete Temperature by Threshold %s" % (threshold))
    cur = conn.cursor()
    cur.execute("DELETE FROM Temperature WHERE Temperature>?", (threshold,))
    conn.commit()

def main():
    print(version)

    if DatabaseExists(database):
        print("Connecting to the database...",database)
        # create a database connection
        conn = create_connection(database)
        with conn:
            threshold = 40.0

            select_temperature_by_threshold(conn, threshold)

            delete_temperature_by_threshold(conn, threshold)

            # Check if zero
            select_temperature_by_threshold(conn, threshold)

        conn.close()

if __name__ == '__main__':
    main()
```

Output

```
python3 sql-delete-test.py
```

```
SQL Test v20180828
Database /home/pi/domoticz/domoticz.db found.
Connecting to the database... /home/pi/domoticz/domoticz.db
Select Temperature by Threshold 40.0
(2, 40.8, 0.0, 0, 0, 0.0, 0.0, '2018-08-28 19:20:00')
Delete Temperature by Threshold 40.0
Select Temperature by Threshold 40.0
```

Node-RED

Purpose

To explore Node-RED (NR) functionality for this Home Automation Solution:

- NR alternative script engine, i.e. perform tasks, i.e. MQTT subscribe or publish
- NR UI Dashboard Add-On as a simple monitor using Gauges & Charts
- Trigger notifications or as watchdog for Domoticz process

Node-RED enables Flow-based programming for the Internet of Things.

If not familiar with the Node-RED concept, strongly recommend reading the Node-RED get started documentation.

Installation

Desktop > Preferences > Recommended Software > All Programms > Click on Node-RED. Press OK. The package will be installed.
After installation, a new Desktop Menu Entry: Programming > Node-RED.

To enable autostart of Node-RED at every boot, run terminal command (including output):

```
sudo systemctl enable nodered.service
Created symlink /etc/systemd/system/multi-user.target.wants/nodered.service →
/lib/systemd/system/nodered.service.
pi@DomoticzDev:~ $
```

To go for sure if working, reboot the Raspberry Pi (sudo shutdown -r now) and access Node-RED (http://rpi-domoticz-ip:1880).

Use disable, to disable autostart.

Access Flows & Dashboard UI

To access Node-RED, point a browser at
http://localhost:1880 (when using Raspberry Pi connected monitor) or
http:// rpi-domoticz-ip:1880 (remote)

This enables to manage Node-RED, define & run flows etc.

If the Dashboard add-on is installed: http://rpi-domoticz-ip:1880/ui

Start, Stop, Log

Task	Terminal Command
Node-RED Start	node-red-start
Node-RED Stop	node-red-stop
Node-RED view recent log output	node-red-log

Update

<TODO> Check if possible via Desktop Check Updates.

Important:
Backup the flows file, i.e. /home/pi/.node-red/flows_DomoticzDev.json, prior updating.
After updating, reboot the Raspberry Pi.

```
cd /home/pi/.node-red
update-nodejs-and-nodered

or if not working

bash <(curl -sL https://raw.githubusercontent.com/node-red/raspbian-deb-
package/master/resources/update-nodejs-and-nodered)
```

Note on using command update-nodejs-and-nodered:
If this message occurs, then use the bash command:
This Node-RED install doesn't support using apt. Please see the online docs at
https://nodered.org/docs/hardware/raspberrypi for the best way to update and upgrade.

Update Log Example

```
This script will remove versions of Node.js prior to version 6.x, 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 pi at /home/pi
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 v8.11.4   Npm 6.4.1
  Clean npm cache                      ✔
  Install Node-RED core                ✔      0.19.3
  Move global nodes to local           -
  Install extra Pi nodes               -
  Npm rebuild existing nodes           ✔
  Add menu shortcut                    ✔
  Update systemd script                ✔
      id=Node-RED.desktop

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  Thu  6 Sep 15:10:56 CEST 2018  -  Finished  Thu  6 Sep 15:13:05 CEST 2018
```

Manage Node Packages

npm

To manage node packages in Node-RED, the npm package is required.
Check if installed:

```
npm -v
```

If package not found, then install via Desktop > Preferences > Add / Remove Software and restart Node-RED (terminal commands node-red-stop, node-red-start)

Check npm version:

```
npm -v
Output
6.4.1
```

Install/Updates

To install or update any node package, do this via Node-RED Menu Settings > Manage Palette > Install.
After install/update, restart Node-RED (terminal commands node-red-stop, node-red-start).
It is not recommended to install using terminal commands.
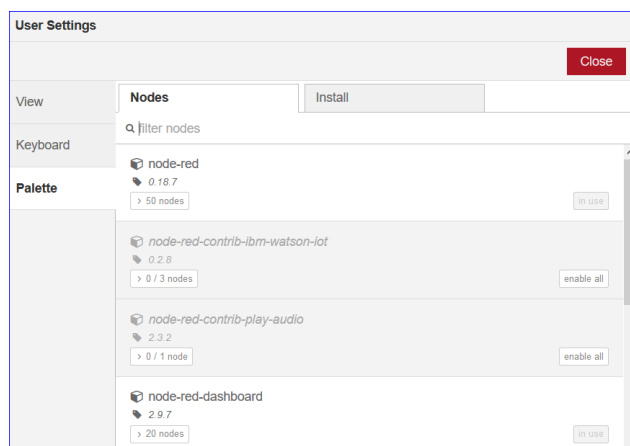
Folder Locations

Data	Folder
Modules	/home/pi/.node-red
Flows	/home/pi/.node-red/flows_DomoticzDev.json

Additional Node Packages	/home/pi/.node-red/node_modules/<package name>
Additional Node Packages Sample Dashboard	/home/pi/.node-red/node_modules/node-red-dashboard

Flow Domoticz MQTT Messages

A simple example flow to test Node-RED.
Subscribe to MQTT topic "domoticz/in" to listen to all Domoticz MQTT messages.

Node-RED Flow

The MQTT Broker is localhost with default port 1883.
The JSON node is used to convert JSON message to JavaScript message.

Node-RED Flow Source

```
[{"id":"45e7c703.b9b9f","type":"mqtt
in","z":"76cb1798.6a823","name":"","topic":"domoticz/out","qos":"2","broker":"d58d713d.0e0eb8","x":110,
"y":340,"wires":[["cc4fd39c.684108"]]},{"id":"cc4fd39c.684108","type":"json","z":"76cb1798.6a823","name
":"","property":"payload","action":"","pretty":false,"x":270,"y":340,"wires":[["552cdad.37816a4"]]},{"i
d":"552cdad.37816a4","type":"debug","z":"76cb1798.6a823","name":"DEBUG
idx=ALL","active":true,"tosidebar":true,"console":false,"tostatus":false,"complete":"payload","x":810,"
y":340,"wires":[]},{"id":"d58d713d.0e0eb8","type":"mqtt-broker","z":"","name":"Domoticz MQTT
Broker","broker":"localhost","port":"1883","clientid":"","usetls":false,"compatmode":true,"keepalive":"
60","cleansession":true,"birthTopic":"","birthQos":"0","birthPayload":"","closeTopic":"","closeQos":"0"
,"closePayload":"","willTopic":"","willQos":"0","willPayload":""}]
```

Output debug
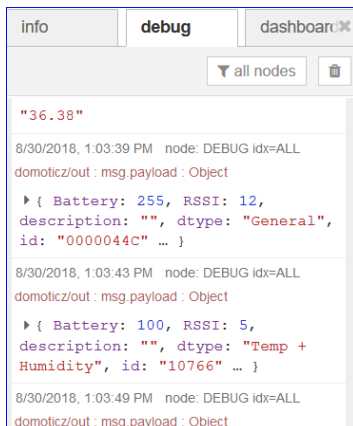
Flow Raspberry Pi CPU Usage

Subscribe to MQTT topic "Domoticz/in" to listen to all Domoticz MQTT messages.
From the message payload (after conversion to JSON), select the property idx with value 1.
Then select the property svalue1 to get the CPU Usage.

Domoticz Topic "Domoticz/in" Message Payload for idx = 1

```
{"Battery":255,"RSSI":12,"description":"","dtype":"General","id":"0000044D","idx":
1,"name":"CPU_Usage","nvalue":0,"stype":"Percentage","svalue1":"0.28","unit":1}
```
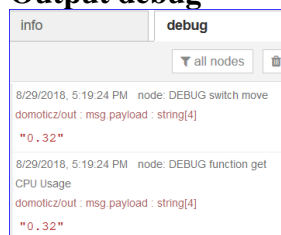
Node-RED Flow
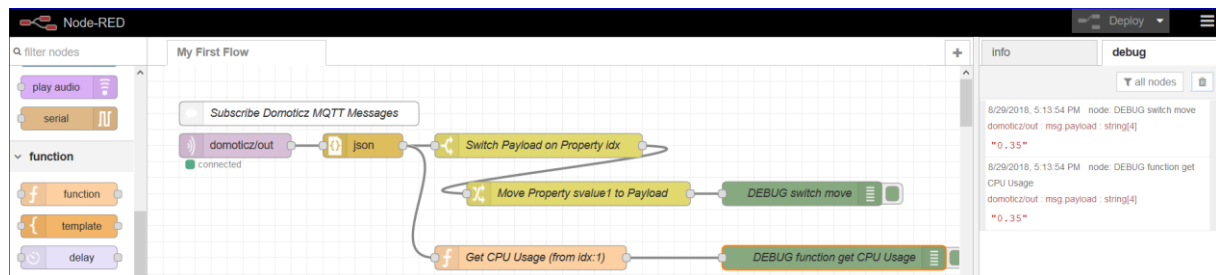
Node-RED Flow Source

```
[{"id":"f67403bf.ebdc88","type":"mqtt
in","z":"76cb1798.6a823","name":"","topic":"domoticz/out","qos":"2","broker":"d58d713d.0e0eb8","x":110,
"y":100,"wires":[["db5fff92.295628"]]},{"id":"35214a2f.f89656","type":"debug","z":"76cb1798.6a823","nam
e":"DEBUG switch
move","active":true,"tosidebar":true,"console":false,"tostatus":false,"complete":"payload","x":820,"y":
160,"wires":[]},{"id":"db5fff92.295628","type":"json","z":"76cb1798.6a823","name":"","property":"payloa
d","action":"","pretty":false,"x":270,"y":100,"wires":[["5f7a4a09.abb57c","13e0d06f.2b15d"]]},{"id":"5f
7a4a09.abb57c","type":"switch","z":"76cb1798.6a823","name":"Switch Payload on Property
idx","property":"payload.idx","propertyType":"msg","rules":[{"t":"eq","v":"1","vt":"num"}],"checkall":"
true","repair":false,"outputs":1,"x":490,"y":100,"wires":[["912289d2.e8c1"]]},{"id":"13e0d06f.2b15d","t
ype":"function","z":"76cb1798.6a823","name":"Get CPU Usage (from idx:5)","func":"// Get the property
idx from the payload\nidx = msg.payload.idx;\ncpu_usage=-1;\n// Check for idx 1\nif (idx == 5) {\n
// Get the cpu usage from property svalue1\n    cpu_usage = msg.payload.svalue1;\n    //
node.warn(\"Idx=1, CPU Usage=\" + cpu_usage);\n    // Assign the cpu usage to the message payload\n
msg.payload = cpu_usage;\n    // Return the message\n    return
msg;\n}\n","outputs":1,"noerr":0,"x":480,"y":240,"wires":[["c030082a.286338","61fef7cc.4fc02","d5bc4ce8
.ae1f28"]]},{"id":"912289d2.e8c1","type":"change","z":"76cb1798.6a823","name":"Move Property svalue1 to
Payload","rules":[{"t":"move","p":"payload.svalue1","pt":"msg","to":"payload","tot":"msg"}],"action":""
,"property":"","from":"","to":"","reg":false,"x":540,"y":160,"wires":[["35214a2f.f89656"]]},{"id":"c030
082a.286338","type":"debug","z":"76cb1798.6a823","name":"DEBUG function get CPU
Usage","active":true,"tosidebar":true,"console":false,"tostatus":false,"complete":"payload","x":860,"y"
:240,"wires":[]},{"id":"acdab92.b0151c8","type":"debug","z":"76cb1798.6a823","name":"DEBUG
idx=1","active":true,"tosidebar":true,"console":false,"tostatus":false,"complete":"payload","x":800,"y"
:100,"wires":[]},{"id":"d58d713d.0e0eb8","type":"mqtt-broker","z":"","name":"Domoticz MQTT
Broker","broker":"localhost","port":"1883","clientid":"","usetls":false,"compatmode":true,"keepalive":"
60","cleansession":true,"birthTopic":"","birthQos":"0","birthPayload":"","closeTopic":"","closeQos":"0"
,"closePayload":"","willTopic":"","willQos":"0","willPayload":""}]
```

Output debug

Node-RED as MQTT Publisher

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

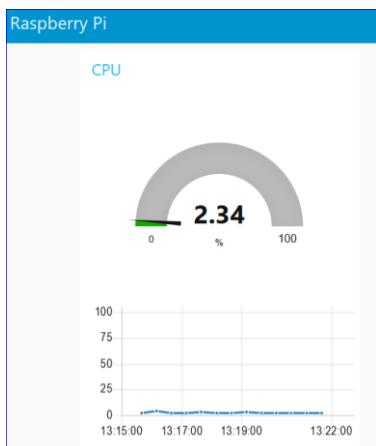For an example, see Dashboard UI > Flow Virtual Sensor.

Dashboard UI

The Node-RED Dashboard module offers UI nodes to build a Dashboard.
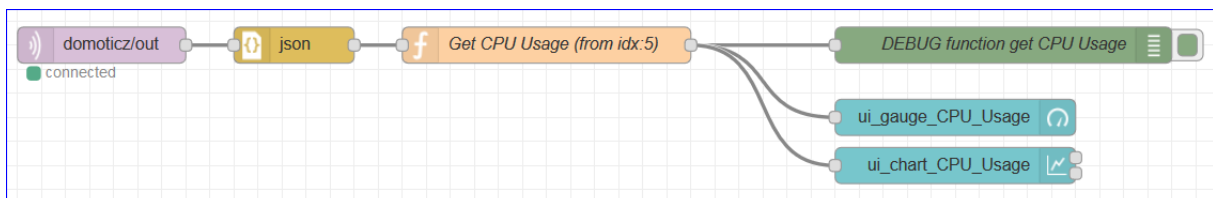
Flow Monitor CPU Usage

Build a simple example of a dashboard using the Raspberry Pi CPU Usage information, which is displayed in a Gauge and Line Chart.

Dashboard

Node-RED Flow

The Function Node selects idx 5 and builds message payload holding the CPU Usage in %.

```
// Get the property idx from the payload
idx = msg.payload.idx;
cpu_usage=-1;
// Check for idx 1
if (idx == 5) {
    // Get the cpu usage from property svalue1
    cpu_usage = msg.payload.svalue1;
    // node.warn("Idx=1, CPU Usage=" + cpu_usage);
    // Assign the cpu usage to the message payload
    msg.payload = cpu_usage;
    // Return the message
    return msg;
}
```

Node-RED Flow Source

```
[{"id":"f67403bf.ebdc88","type":"mqtt
in","z":"76cb1798.6a823","name":"","topic":"domoticz/out","qos":"2","broker":"d58d713d.0e0eb8","x":110,
"y":240,"wires":[["db5fff92.295628"]]},{"id":"db5fff92.295628","type":"json","z":"76cb1798.6a823","name
":"","property":"payload","action":"","pretty":false,"x":270,"y":240,"wires":[["13e0d06f.2b15d"]]},{"id
":"13e0d06f.2b15d","type":"function","z":"76cb1798.6a823","name":"Get CPU Usage (from
idx:5)","func":"// Get the property idx from the payload\nidx = msg.payload.idx;\ncpu_usage=-1;\n//
Check for idx 1\nif (idx == 5) {\n    // Get the cpu usage from property svalue1\n    cpu_usage =
msg.payload.svalue1;\n    // node.warn(\"Idx=1, CPU Usage=\" + cpu_usage);\n    // Assign the cpu usage
to the message payload\n    msg.payload = cpu_usage;\n    // Return the message\n    return
msg;\n}\n","outputs":1,"noerr":0,"x":480,"y":240,"wires":[["c030082a.286338","61fef7cc.4fc02","d5bc4ce8
.ae1f28"]]},{"id":"c030082a.286338","type":"debug","z":"76cb1798.6a823","name":"DEBUG function get CPU
Usage","active":true,"tosidebar":true,"console":false,"tostatus":false,"complete":"payload","x":860,"y"
:240,"wires":[]},{"id":"61fef7cc.4fc02","type":"ui_gauge","z":"76cb1798.6a823","name":"ui_gauge_CPU_Usa
ge","group":"c3b6865e.2b93e","order":0,"width":0,"height":0,"gtype":"gage","title":"","label":"%","form
at":"{{value}}","min":0,"max":"100","colors":["#00b500","#e6e600","#ca3838"],"seg1":"","seg2":"","x":82
0,"y":300,"wires":[]},{"id":"d5bc4ce8.ae1f28","type":"ui_chart","z":"76cb1798.6a823","name":"ui_chart_C
PU_Usage","group":"c3b6865e.2b93e","order":0,"width":0,"height":0,"label":"","chartType":"line","legend
":"false","xformat":"HH:mm:ss","interpolate":"linear","nodata":"","dot":true,"ymin":"0","ymax":"100","r
emoveOlder":1,"removeOlderPoints":"","removeOlderUnit":"3600","cutout":0,"useOneColor":false,"colors":[
"#1f77b4","#aec7e8","#ff7f0e","#2ca02c","#98df8a","#d62728","#ff9896","#9467bd","#c5b0d5"],"useOldStyle
":false,"x":820,"y":340,"wires":[[],[]]},{"id":"d58d713d.0e0eb8","type":"mqtt-
broker","z":"","name":"Domoticz MQTT
Broker","broker":"localhost","port":"1883","clientid":"","usetls":false,"compatmode":true,"keepalive":"
60","cleansession":true,"birthTopic":"","birthQos":"0","birthPayload":"","closeTopic":"","closeQos":"0"
,"closePayload":"","willTopic":"","willQos":"0","willPayload":""},{"id":"c3b6865e.2b93e","type":"ui_gro
up","z":"","name":"CPU","tab":"8a1743ab.9d931","disp":true,"width":"6","collapse":false},{"id":"8a1743a
b.9d931","type":"ui_tab","z":"","name":"Raspberry Pi","icon":"dashboard"}]
```

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").

Switch Node and Change Node

Instead of using a Function Node, Switch Node and Change Node can be used.
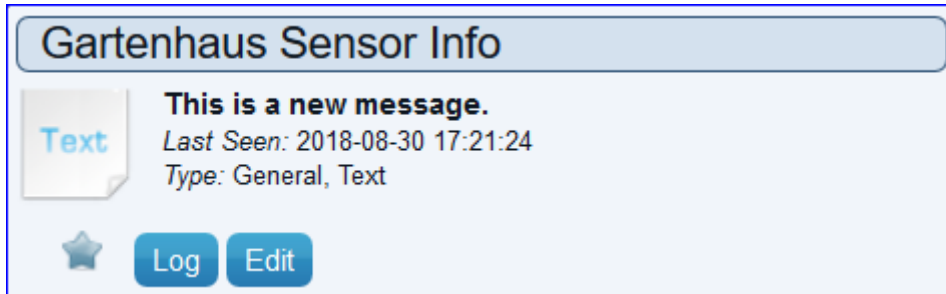
Node-RED Flow
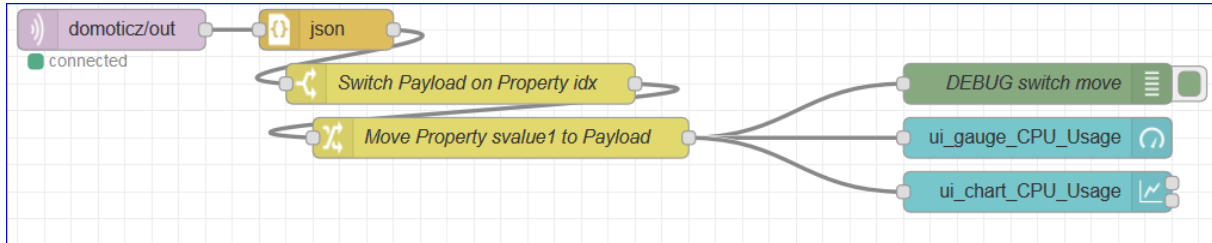
Node-RED Flow Source

```
…
[{"id":"f67403bf.ebdc88","type":"mqtt
in","z":"76cb1798.6a823","name":"","topic":"domoticz/out","qos":"2","broker":"d58d
713d.0e0eb8","x":110,"y":100,"wires":[["db5fff92.295628"]]},{"id":"35214a2f.f89656
","type":"debug","z":"76cb1798.6a823","name":"DEBUG switch
move","active":true,"tosidebar":true,"console":false,"tostatus":false,"complete":"
payload","x":800,"y":140,"wires":[]},{"id":"db5fff92.295628","type":"json","z":"76
cb1798.6a823","name":"","property":"payload","action":"","pretty":false,"x":270,"y
":100,"wires":[["5f7a4a09.abb57c"]]},{"id":"5f7a4a09.abb57c","type":"switch","z":"
76cb1798.6a823","name":"Switch Payload on Property
idx","property":"payload.idx","propertyType":"msg","rules":[{"t":"eq","v":"5","vt"
:"num"}],"checkall":"true","repair":false,"outputs":1,"x":370,"y":140,"wires":[["9
12289d2.e8c1"]]},{"id":"912289d2.e8c1","type":"change","z":"76cb1798.6a823","name"
:"Move Property svalue1 to
Payload","rules":[{"t":"move","p":"payload.svalue1","pt":"msg","to":"payload","tot
":"msg"}],"action":"","property":"","from":"","to":"","reg":false,"x":400,"y":180,
"wires":[["35214a2f.f89656","61fef7cc.4fc02","d5bc4ce8.ae1f28"]]},{"id":"61fef7cc.
4fc02","type":"ui_gauge","z":"76cb1798.6a823","name":"ui_gauge_CPU_Usage","group":
"c3b6865e.2b93e","order":0,"width":0,"height":0,"gtype":"gage","title":"","label":
"%","format":"{{value}}","min":0,"max":"100","colors":["#00b500","#e6e600","#ca383
8"],"seg1":"","seg2":"","x":800,"y":180,"wires":[]},{"id":"d5bc4ce8.ae1f28","type"
:"ui_chart","z":"76cb1798.6a823","name":"ui_chart_CPU_Usage","group":"c3b6865e.2b9
3e","order":0,"width":0,"height":0,"label":"","chartType":"line","legend":"false",
"xformat":"HH:mm:ss","interpolate":"linear","nodata":"","dot":true,"ymin":"0","yma
x":"100","removeOlder":1,"removeOlderPoints":"","removeOlderUnit":"3600","cutout":
0,"useOneColor":false,"colors":["#1f77b4","#aec7e8","#ff7f0e","#2ca02c","#98df8a",
"#d62728","#ff9896","#9467bd","#c5b0d5"],"useOldStyle":false,"x":800,"y":220,"wire
s":[[],[]]},{"id":"d58d713d.0e0eb8","type":"mqtt-broker","z":"","name":"Domoticz
MQTT
Broker","broker":"localhost","port":"1883","clientid":"","usetls":false,"compatmod
e":true,"keepalive":"60","cleansession":true,"birthTopic":"","birthQos":"0","birth
Payload":"","closeTopic":"","closeQos":"0","closePayload":"","willTopic":"","willQ
os":"0","willPayload":""},{"id":"c3b6865e.2b93e","type":"ui_group","z":"","name":"
CPU","tab":"8a1743ab.9d931","disp":true,"width":"6","collapse":false},{"id":"8a174
3ab.9d931","type":"ui_tab","z":"","name":"Raspberry Pi","icon":"dashboard"}]
```
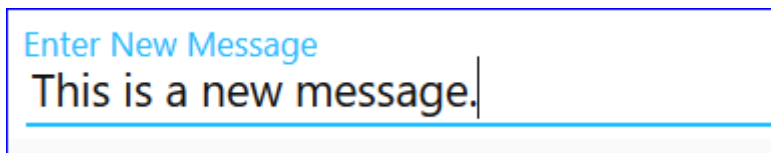
Flow Virtual Sensor Text Update

Update the Virtual Sensor "Gartenhaus Sensor Info" with new text.
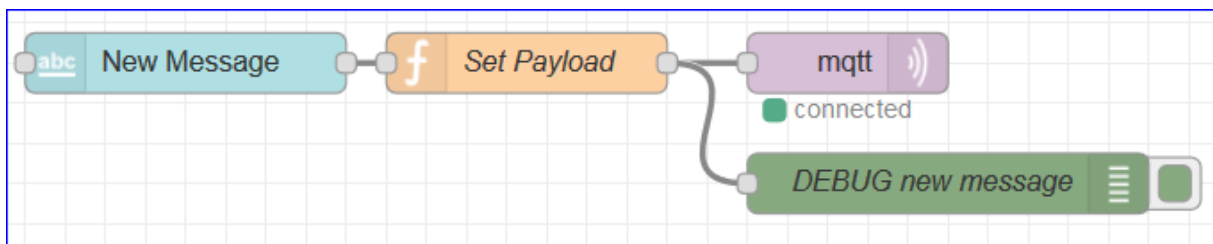The device has idx 52, Type General, Text.

Widget showing the updated text

Node-RED Dashboard UI with ui_text_input Node

Node-RED Flow with Nodes ui_text_input > function > mqtt out

The text input node has a delay of 0, means the content is sent after pressing enter.
The function node created the new message which is published via MQTT using the localhost broker.

```
// Build a new message
var msginfo = {};
msginfo.topic = "domoticz/in";

// Get the text from the incoming payload from the text input node.
text = msg.payload;

// Build the new payload
msginfo.payload = { "idx" : 52, "nvalue": 0, "svalue": text};
node.warn(msginfo.payload);

// Return the message
return msginfo;
```

The information for building the MQTT payload is taken from the Domoticz API/JSON URL's documentation (here).

Node-RED Flow Source

```
[{"id":"38d6f42f.539a94","type":"ui_text_input","z":"76cb1798.6a823","name":"New
Message","label":"Enter New
Message","group":"27ca36a9.4c3cba","order":0,"width":0,"height":0,"passthru":false,"mode":"text","delay
":"0","topic":"","x":120,"y":820,"wires":[["896e27c.ed14dd8"]]},{"id":"d4be5444.33e23","type":"mqtt
out","z":"76cb1798.6a823","name":"","topic":"","qos":"","retain":"","broker":"d58d713d.0e0eb8","x":450,
"y":820,"wires":[]},{"id":"896e27c.ed14dd8","type":"function","z":"76cb1798.6a823","name":"Set
Payload","func":"// Build a new message\nvar msginfo = {};\nmsginfo.topic = \"domoticz/in\";\n\n// Get
the text from the incoming payload\ntext = msg.payload;\n\n// Build the new payload\nmsginfo.payload =
{ \"idx\" : 52, \"nvalue\": 0, \"svalue\": text};\nnode.warn(msginfo.payload);\n\n// Return the
message\nreturn
msginfo;\n","outputs":1,"noerr":0,"x":290,"y":820,"wires":[["d4be5444.33e23","f167bf33.2066e8"]]},{"id"
:"f167bf33.2066e8","type":"debug","z":"76cb1798.6a823","name":"DEBUG new
message","active":true,"tosidebar":true,"console":false,"tostatus":false,"complete":true,"x":500,"y":
880,"wires":[]},{"id":"27ca36a9.4c3cba","type":"ui_group","z":"","name":"Hue","tab":"8a1743ab.9d931","d
isp":true,"width":"6","collapse":false},{"id":"d58d713d.0e0eb8","type":"mqtt-
broker","z":"","name":"Domoticz MQTT
Broker","broker":"localhost","port":"1883","clientid":"","usetls":false,"compatmode":true,"keepalive":"
60","cleansession":true,"birthTopic":"","birthQos":"0","birthPayload":"","closeTopic":"","closeQos":"0"
,"closePayload":"","willTopic":"","willQos":"0","willPayload":""},{"id":"8a1743ab.9d931","type":"ui_tab
","z":"","name":"Raspberry Pi","icon":"dashboard"}]
```

Python Plug In

Purpose

The Python Plugin Framework allows to create an interface between Hardware (or Virtual Hardware) and Domoticz.

Mandatory to read this manual first and use as reference.

My first Python Plugin, Soil Moisture Monitor, can be found here.
Plugin Purpose: Measure soil moisture of a plant, display the value in Domoticz Home Automation System, in a TinkerForge Segment Display 4x7 and TinkerForge RGB LED indicator.

Must state, it makes real fun to see how new hardware devices with according devices are added and used.
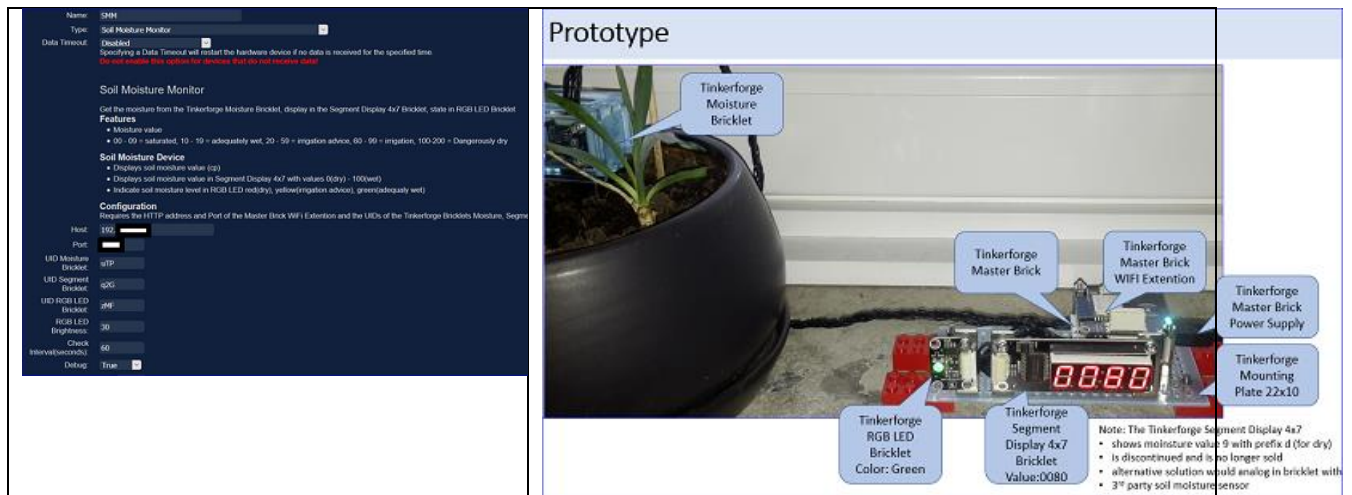
Appendix Domoticz Hints

Start, Stop, Status

Terminal commands

```
sudo service domoticz.sh start
sudo service domoticz.sh stop
sudo service domoticz.sh status
```

Output Status Sample

```
domoticz.service - LSB: Home Automation System
   Loaded: loaded (/etc/init.d/domoticz.sh)
   Active: active (running) since Mon 2018-08-27 09:32:39 CEST; 16min ago
  Process: 31803 ExecStop=/etc/init.d/domoticz.sh stop (code=exited,
status=0/SUCCESS)
  Process: 32093 ExecStart=/etc/init.d/domoticz.sh start (code=exited,
status=0/SUCCESS)
   CGroup: /system.slice/domoticz.service
           └─32100 /home/pi/domoticz/domoticz -daemon -www 8080 -sslww 443

Aug 27 09:32:39 139 domoticz.sh[32093]: 2018-08-27 09:32:39.694  Status: Dom...z
Aug 27 09:32:39 139 domoticz[32098]: Domoticz is starting up....
Aug 27 09:32:39 139 domoticz[32100]: Domoticz running...
Aug 27 09:32:39 139 domoticz.sh[32093]: 2018-08-27 09:32:39.695  Status: Bui...6
Aug 27 09:32:39 139 domoticz.sh[32093]: 2018-08-27 09:32:39.695  Status: Sys...i
Aug 27 09:32:39 139 domoticz.sh[32093]: 2018-08-27 09:32:39.695  Status: Sta.../
Aug 27 09:32:39 139 domoticz.sh[32093]: domoticz: Domoticz is starting up....
Aug 27 09:32:39 139 systemd[1]: Started LSB: Home Automation System.
Hint: Some lines were ellipsized, use -l to show in full.
```

Lost Username and Password

Open terminal session and login as User Pi

Stop Domoticz and wait 15 seconds to complete

```
sudo service domoticz.sh stop
```

Check Domoticz Status

```
sudo service domoticz.sh stop
```

Change to Domoticz Folder

```
cd domoticz
```

Note: /home/pi/Domoticz
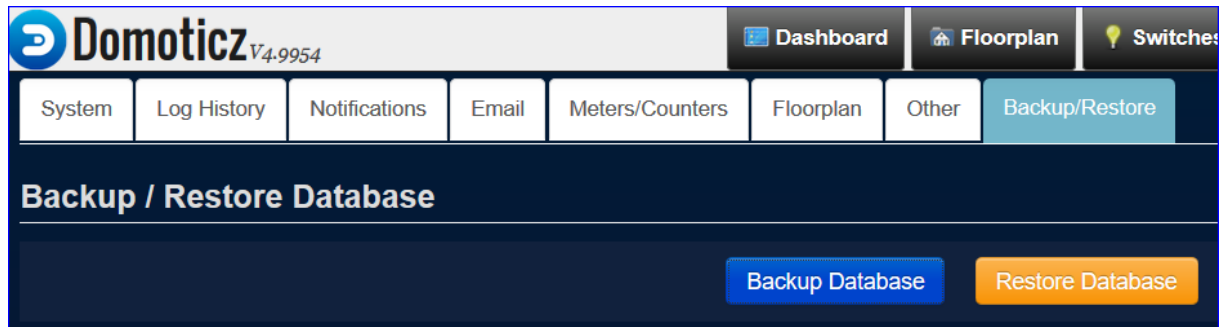
Start Domoticz with parameter no www password

```
sudo ./domoticz -nowwwpwd
```

Create Backup

From the Domoticz UI go to:
Setup > Settings > Backup/Restore : Click Backup Database

This will create a file in the downloads folder on the development device.

Troubleshooting

Device Widget turns yellow/red

Check the device status using URL with parameter type=devices and rid=idx.

Example
Checking device named Windmesser (TFA Dostmann 30.3168 Windmeter) because the weather widget turned yellow.

From browser run:

http://NNN.NNN.1.60:8080/json.htm?type=devices&rid=28
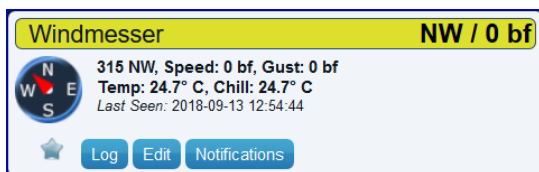
Result JSON string (snippet)

```
{
    "app_version" : "4.9999",
    "result" : [
        {
            "BatteryLevel" : 0,
            "Data" : "22.00;NNE;5;0;26.5;26.5",
            "HardwareName" : "RFXtrx433e",
            "HardwareType" : "RFXCOM - RFXtrx433 USB 433.92MHz Transceiver",
            "SignalLevel" : 6,
            "Used" : 1,
            "idx" : "28"
        }
    ],
    "status" : "OK",
    "title" : "Devices"
}
```

From the returned result, the BatteryLevel is 0.
It seems that the rechargeable battery is defect for this device.
The battery is recharged by a small solar panel on top of the device.

Appendix Domoticz Build Source

Sample commands for building Domoticz on a Raspberry Pi from the Domoticz Source. Followed the instructions from [here](#).

```
Swap File 100 to 500MB
sudo service dphys-swapfile status
sudo swapoff -a
free
sudo nano /etc/dphys-swapfile
  change 100 to 500
sudo swapon -a
free

Domoticz Build (took about 2 hours)
sudo apt-get install cmake make gcc g++ libssl-dev git libcurl4-gnutls-dev libusb-
dev python3-dev zlib1g-dev

mkdir boost

cd boost

wget https://dl.bintray.com/boostorg/release/1.68.0/source/boost_1_68_0.tar.gz

tar xfvz boost_1_68_0.tar.gz

cd boost_1_68_0/

./bootstrap.sh

./b2 stage threading=multi link=static --with-thread --with-system

sudo ./b2 install threading=multi link=static --with-thread --with-system

cd ../../

rm -Rf boost/

cd /home/pi

free

git clone https://github.com/domoticz/domoticz.git dev-domoticz
  Cloning into 'dev-domoticz'...
  remote: Counting objects: 59677, done.
  remote: Compressing objects: 100% (16/16), done.
  remote: Total 59677 (delta 9), reused 13 (delta 6), pack-reused 59655
  Receiving objects: 100% (59677/59677), 206.16 MiB | 3.82 MiB/s, done.
  Resolving deltas: 100% (43178/43178), done.
  Checking out files: 100% (2784/2784), done.

cd dev-domoticz

git pull
  Already up-to-date.
```

```
cmake -DCMAKE_BUILD_TYPE=Release CMakeLists.txt
  -- The C compiler identification is GNU 6.3.0
  -- The CXX compiler identification is GNU 6.3.0
  ...
  -- Compiling Revision #9700
  -- Configuring done
  -- Generating done
  -- Build files have been written to: /home/pi/dev-domoticz

make
  Scanning dependencies of target minizip
  ...
  [100%] Linking CXX executable domoticz
  [100%] Built target domoticz

ls
…
domoticz.sh
…

sudo ./domoticz
  2018-09-12 16:18:10.882  Status: Domoticz V4.9999 (c)2012-2018 GizMoCuz
  2018-09-12 16:18:10.882  Status: Build Hash: 9e8ea729, Date: 2018-09-09 18:27:57
  2018-09-12 16:18:10.883  Status: Startup Path: /home/pi/dev-domoticz/
  2018-09-12 16:18:11.292  Status: PluginSystem: Started, Python version '3.5.3'.
  2018-09-12 16:18:11.299  Active notification Subsystems: gcm, http (2/13)
  2018-09-12 16:18:11.304  Status: WebServer(HTTP) started on address: :: with port
8080
  2018-09-12 16:18:11.318  Status: WebServer(SSL) started on address: :: with port
443
  2018-09-12 16:18:11.319  Status: Proxymanager started.
  2018-09-12 16:18:11.321  Starting shared server on: :::6144
  2018-09-12 16:18:11.321  Status: TCPServer: shared server started...
  2018-09-12 16:18:11.322  Status: RxQueue: queue worker started...
  2018-09-12 16:18:13.323  Status: EventSystem: reset all events...
  2018-09-12 16:18:13.323  Status: EventSystem: reset all device statuses...
  2018-09-12 16:18:13.472  Status: PluginSystem: Entering work loop.
  2018-09-12 16:18:13.480  Status: Python EventSystem: Initalizing event module.
  2018-09-12 16:18:13.481  Status: EventSystem: Started
  2018-09-12 16:18:13.481  Status: EventSystem: Queue thread started...
  2018-09-12 16:18:42.447  Status: Incoming connection from: NNN.NNN.1.3
```

Updating Domoticz
Binary install update:
Use the Web Interface to update Domoticz (Setup->Check for Updates)

Manual update:
Stable: cd domoticz, ./updaterelease
Beta: cd domoticz, ./updatebeta
Source Update: cd domoticz, git pull, make -j 4 OR make

Appendix volkszaehler Setup

Description of the volkszaehler with the IR device, connected to a Raspberry Pi.
volkszaehler.org is an Open Source (GPL license) Smart Metering Hard- and Software
solution. Important to read the "howto get started" first to get an understanding of the concept.
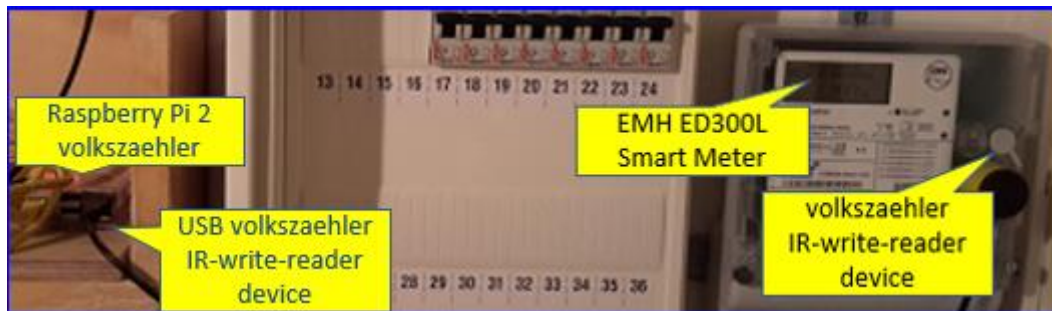
Volkszaehler Hardware setup

Hardware

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

Software

- Raspberry Pi Linux vz 4.14.62-v7+ #1134 (armv7l)
- volkszaehler Raspberry Pi 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 WLAN static 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 rpi-domoticz-ip
netmask 255.255.255.0
broadcast NNN.NNN.0.255
gateway rpi-domoticz-ip
```

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 Ethernet static 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.

volkszahler Raspberry Pi [Raspbian GNU/Linux 8 (jessie)] Linux 39 4.1.19-v7+ #858	Home Automation Raspberry Pi [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/dhcpcd.conf	sudo nano /etc/dhcpcd.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
```

Entry:
```
SUBSYSTEM=="tty", ATTRS{idVendor}=="10c4", ATTRS{idProduct}=="ea60",
ATTRS{serial}=="00F2E620", SYMLINK+="ttyUSB-VZ", MODE="0666"
```

Check the USB device information with:
```
sudo lsusb -v
```

To check detailed USB port information, use the command:
```
udevadm info --query=all --name=/dev/ttyUSB-VZ
```

Output Example
```
P: /devices/platform/soc/3f980000.usb/usb1/1-1/1-1.5/1-1.5:1.0/ttyUSB0/tty/ttyUSB0
N: ttyUSB0
S: serial/by-id/usb-Silicon_Labs_CP2104_USB_to_UART_Bridge_Controller_00F2E620-if00-port0
S: serial/by-path/platform-3f980000.usb-usb-0:1.5:1.0-port0
S: ttyUSB-VZ
E: DEVLINKS=/dev/serial/by-path/platform-3f980000.usb-usb-0:1.5:1.0-port0 /dev/serial/by-id/usb-
Silicon_Labs_CP2104_USB_to_UART_Bridge_Controller_00F2E620-if00-port0 /dev/ttyUSB-VZ
E: DEVNAME=/dev/ttyUSB0
E: DEVPATH=/devices/platform/soc/3f980000.usb/usb1/1-1/1-1.5/1-1.5:1.0/ttyUSB0/tty/ttyUSB0
E: ID_BUS=usb
E: ID_MODEL=CP2104_USB_to_UART_Bridge_Controller
E: ID_MODEL_ENC=CP2104\x20USB\x20to\x20UART\x20Bridge\x20Controller
E: ID_MODEL_FROM_DATABASE=CP210x UART Bridge / myAVR mySmartUSB light
E: ID_MODEL_ID=ea60
E: ID_PATH=platform-3f980000.usb-usb-0:1.5:1.0
E: ID_PATH_TAG=platform-3f980000_usb-usb-0_1_5_1_0
E: ID_REVISION=0100
E: ID_SERIAL=Silicon_Labs_CP2104_USB_to_UART_Bridge_Controller_00F2E620
E: ID_SERIAL_SHORT=00F2E620
E: ID_TYPE=generic
E: ID_USB_DRIVER=cp210x
E: ID_USB_INTERFACES=:ff0000:
E: ID_USB_INTERFACE_NUM=00
E: ID_VENDOR=Silicon_Labs
E: ID_VENDOR_ENC=Silicon\x20Labs
E: ID_VENDOR_FROM_DATABASE=Cygnal Integrated Products, Inc.
E: ID_VENDOR_ID=10c4
E: MAJOR=188
E: MINOR=0
E: SUBSYSTEM=tty
E: TAGS=:systemd:
E: USEC_INITIALIZED=5005222
```

USB Configuration

The USB port settings are 9600 Baud 8 n.

Set via command stty:

```
stty -F /dev/ttyUSB1
1:0:8bd:0:3:1c:7f:15:4:5:1:0:11:13:1a:0:12:f:17:16:0:0:0:0:0:0:0:0:0:0:0:0:0:0
```

Minicom
The setup can also be done via minicom

```
sudo apt-get install minicom
sudo minicom -s
```

Ensure port is set during boot
Add this line into /etc/rc.local (by sudo nano /etc/rc.local)

```
stty -F /dev/ttyUSB-VZ
1:0:8bd:0:3:1c:7f:15:4:5:1:0:11:13:1a:0:12:f:17:16:0:0:0:0:0:0:0:0:0:0:0:0:0:0
```

Check Data coming from the USB device (option 1):

```
cat /dev/ttyUSB-VZ | od -tx1
```

Output Example:

```
    0000000 2f 38 01 01 63 e4 c7 00 76 07 00 0a 00 07 db 75
    0000020 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/ttyUSB-VZ
```

Define Channels

To define the channels, reboot the RPi and access via Webbrowser the volkszaehler Frontend using its ip address: http://rpi-vz-ip

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","t
itle":"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","t
itle":"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":"Wirklei
sting","fillstyle":0,"public":true,"resolution":10000,"style":"steps","title":"Wir
kleistung","yaxis":"auto"}}
```

vzlogger Setup

The defined channels must be defined in the file /etc/vzlogger.conf :

```
sudo nano /etc/vzlogger.conf
```

Two channels are defined **"Energie Leisting"** and **"Energie Zaehlerstand"**:

```
/**
 * vzlogger configuration
 * Use properly encoded JSON with javascript comments
 * http://wiki.volkszaehler.org/software/controller/vzlogger#configuration
 * Online configuration editor: http://volkszaehler.github.io/vzlogger/
 */

{
// General settings
"daemon": true, // run periodically
"verbosity": 0, // log (0=log_alert,1=log_error,3=log_warning,5=log_info,10=log_debug,15=log_finest)
"log": "/var/log/vzlogger.log",   // log file, optional
"retry": 30,                       // http retry delay in seconds

// Build-in HTTP server
"local": {
  "enabled": false,      // enable local HTTPd for serving live readings
  "port": 8080,          // TCP port for local HTTPd
  "index": true,         // provide index listing of available channels if no UUID was requested
  "timeout": 30,         // timeout for long polling comet requests in seconds (0 disables comet)
  "buffer": 600          // HTTPd buffer configuration for serving readings, default -1
                         //   >0: number of seconds of readings to serve
                         //   <0: number of tuples to server per channel (e.g. -3 will serve 3 tuples)
},

"push": [],
"meters" : [
  {
    "enabled" : true,
    "protocol" : "sml",
    "device" : "/dev/ttyUSB-VZ",
    "baudrate": 9600,
    "parity": "8n1",
    "channels":
    [
      {
        // Leistungswert
        "uuid" : "958bce60-b342-11e8-b54f-bbec0573e1f4",
        "middleware" : "http://localhost/middleware.php",
        "identifier" : "1-0:16.7.0"
      },
      {
        // Zaehlerstand
        "uuid" : "5b8ea2a0-b342-11e8-bec6-15c040e6d041",
        "middleware" : "http://localhost/middleware.php",
        "identifier" : "1-0:1.8.0"
      }
    ]
  }]
}
```

volkszaehler Frontend http://rpi-vz-ip showing the two channels defined.
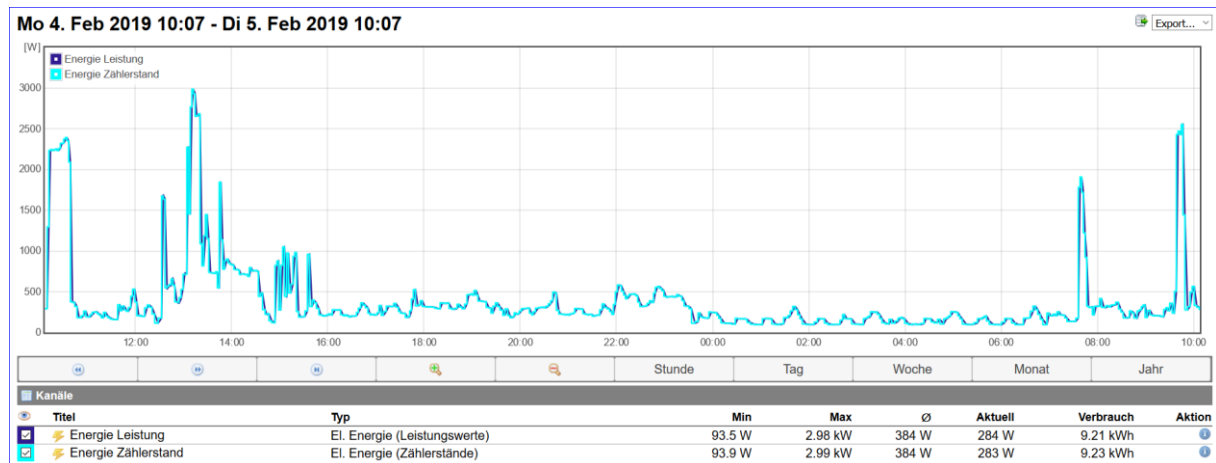
Clicking on the Aktion (i) icon provides the details for the two channels defined:

API Channel Properties

Energie Leistung

```
http://rpi-vz-ip/middleware.php/channel/958bce60-b342-11e8-b54f-bbec0573e1f4.json
```

```
{"version":"0.3","entity":{"uuid":"958bce60-b342-11e8-b54f-
bbec0573e1f4","type":"powersensor","color":"#311b92","fillstyle":0,"public":true,"
style":"lines","title":"Energie Leistung","yaxis":"auto"}}
```

Energie Zaehlerstand

```
http://rpi-vz-ip/middleware.php/channel/5b8ea2a0-b342-11e8-bec6-15c040e6d041.json
```

```
{"version":"0.3","entity":{"uuid":"5b8ea2a0-b342-11e8-bec6-
15c040e6d041","type":"electric
meter","color":"aqua","fillstyle":0,"public":true,"resolution":1000,"style":"steps
","title":"Energie Z\u00e4hlerstand","yaxis":"auto"}}
```

vzlogger Service

Ensure the vzlogger service is correct defined.
Note: Ensure the vzlogger command is using the parameter -c (see ExecStart).

```
cd /etc/systemd/system
sudo nano vzlogger.service
```

Content:

```
 [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

vzlogger Enable as a Service	systemctl enable vzlogger
vzlogger Start	systemctl start vzlogger OR service vzlogger start
vzlogger Stop	systemctl stop vzlogger OR service vzlogger stop
vzlogger Status	systemctl status vzlogger Example output:
vzlogger Daemon reload	systemctl daemon-reload
vzlogger Debug	vzlogger –f

Test

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.
```

API volkszaehler Hints

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

API Obtain Configuration Request:

```
http://rpi-vz-ip/middleware.php/entity/958bce60-b342-11e8-b54f-bbec0573e1f4.json
```

Result

```
{"version":"0.3","entity":{"uuid":"958bce60-b342-11e8-b54f-
bbec0573e1f4","type":"powersensor","color":"#311b92","fillstyle":0,"public":true,"
style":"lines","title":"Energie Leistung","yaxis":"auto"}}
```

API Channel Properties

```
http://rpi-vz-ip/middleware.php/channel/958bce60-b342-11e8-b54f-bbec0573e1f4.json
```

Result

```
{"version":"0.3","entity":{"uuid":"958bce60-b342-11e8-b54f-
bbec0573e1f4","type":"powersensor","color":"#311b92","fillstyle":0,"public":true,"
style":"lines","title":"Energie Leistung","yaxis":"auto"}}
```

API Entities

```
http://rpi-vz-ip/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://rpi-vz-ip/middleware.php/data/23844c00-8895-11e5-a642-6d8ce36c9a44.json
```

Get One Record

```
http://rpi-vz-ip/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":[1447316802613,246.491
34831111],"max":[1447316802613,246.49134831111],"average":246.491,"consumption":37
52.3,"rows":1}}
```

Get the last value

http://rpi-vz-ip/middleware.php/data/23844c00-8895-11e5-a642-
6d8ce36c9a44.json?from=now&to=now

Result
```
{"version":"0.3","data":{"tuples":[[1447317119885,201.681,1],[1447317121700,198.34
7,1]],"uuid":"23844c00-8895-11e5-a642-
6d8ce36c9a44","from":1447317118100,"to":1447317121700,"min":[1447317121700,198.347
10743513],"max":[1447317119885,201.68067227331],"average":200,"consumption":0.2,"r
ows":3}}
```

API Logging

```
http://rpi-vz-ip/middleware.php/data/23844c00-8895-11e5-a642-
6d8ce36c9a44.json?ts=1284677961150&value=12
```

Using the volkzaehler Frontend to display data
http://rpi-vz-ip/frontend/?uuid=23844c00-8895-11e5-a642-6d8ce36c9a44

Appendix TODO

Topic	Details
Notifications	Explore usage of notifications other then Email.
Thresholds	Set threshold via another UI, i.e. Node-RED or Custom UI.
Build house control functions	These are implemented but not shared yet.
DomoticzEventScriptsViewer	Share the solution developed with Lazarus.