

# Heimautomation mit Home Assistant

Eine Plattform, die es schafft einem riesigen und chaotischen Ökosystem einen Sinn und Verstand zu geben.

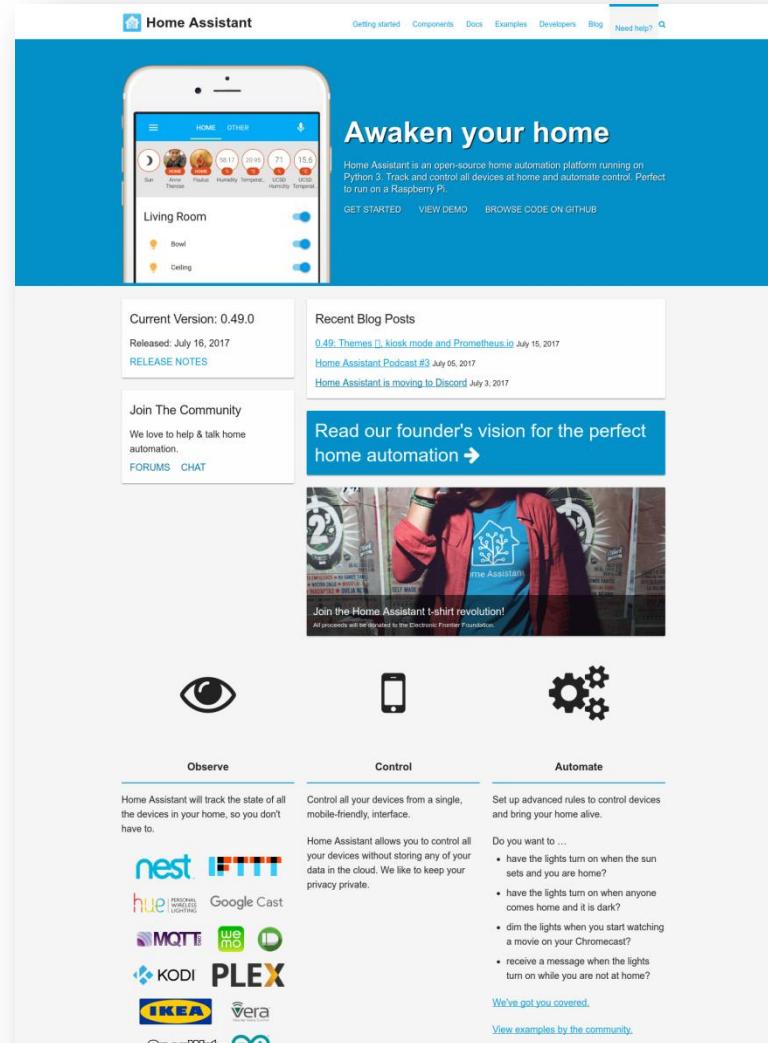
Sebastian Muszynski

17.07.2017

# Home Assistant

## ■ Plattform zur Heimautomation

- Verfolgung von Gerätezuständen
- Steuerung von Geräten
- Automatisierung
- Mobilgerätefreundliche Web-Oberfläche
- Lokale Datenspeicherung
- Leichtgewichtig
- Durchdacht!
  
- freie Lizenz, leicht zu installieren, erweiterbar



The screenshot shows the official Home Assistant website. At the top, there's a navigation bar with links for "Getting started", "Components", "Docs", "Examples", "Developers", "Blog", and "Need help?". Below the navigation is a banner with the text "Awaken your home" and a subtext about Home Assistant being an open-source platform for Python 3. It includes links to "GET STARTED", "VIEW DEMO", and "BROWSE CODE ON GITHUB".  
  
On the left, there's a sidebar with "Current Version: 0.49.0" (Released: July 16, 2017), a link to "RELEASE NOTES", and a "Join The Community" section with links to "FORUMS" and "CHAT".  
  
The main content area features a large image of a person wearing a "Home Assistant" t-shirt. Below it, there's a call-to-action button: "Read our founder's vision for the perfect home automation →".  
  
At the bottom, there are three sections: "Observe" (with icons for eye, smartphone, and gears), "Control" (with icons for eye, smartphone, and gears), and "Automate" (with icons for eye, smartphone, and gears). Each section contains text and logos for various integrations like Nest, IFTTT, Hue, Google Cast, MQTT, WeMo, Kodi, Plex, IKEA, and Vera.

# Motivation

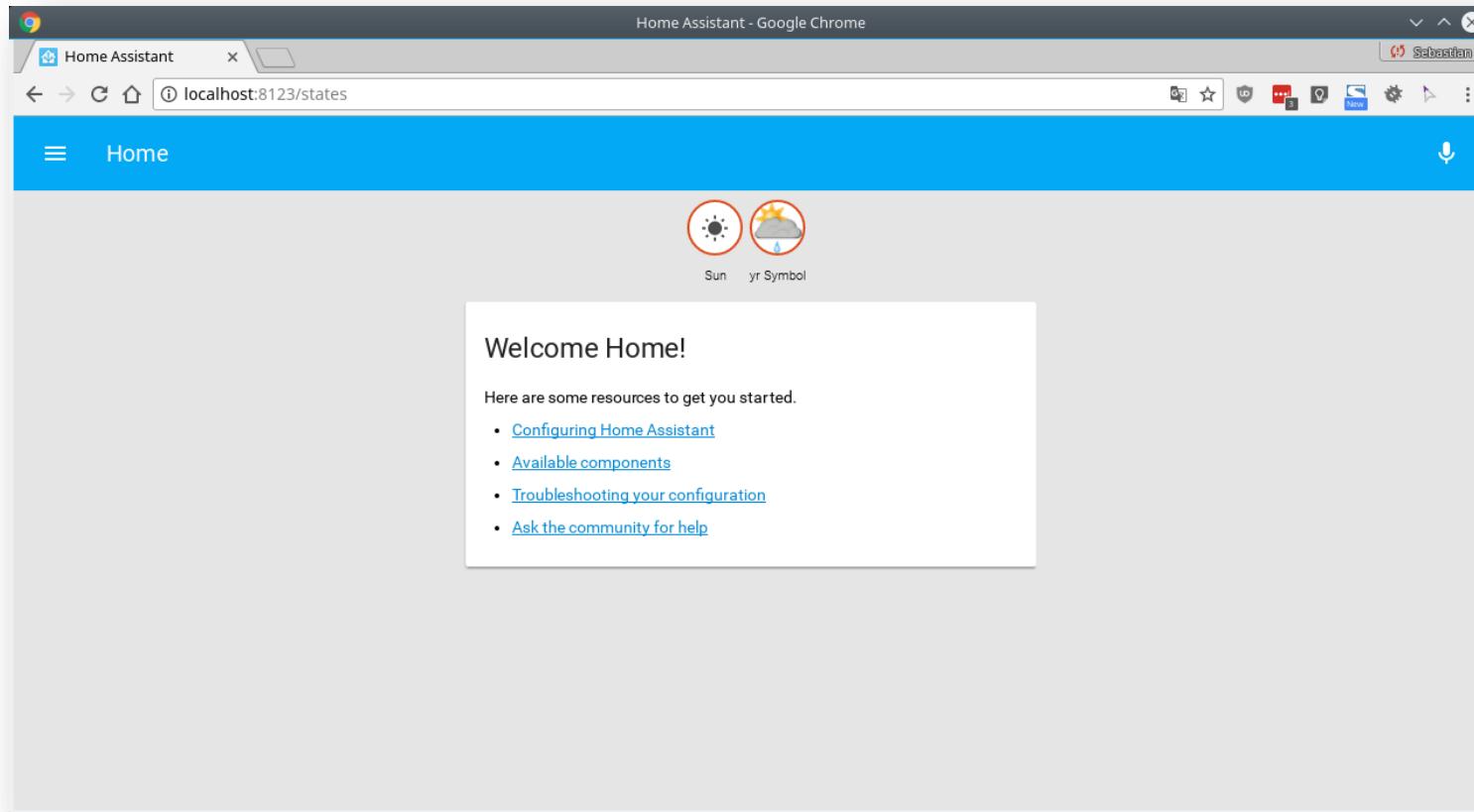
## Das Internet der Dinge: Ein riesiges und chaotisches Ökosystem

Home Assistant		Getting started Components Docs Examples Devsres Blog Need help?																		
// Components																				
Acer Projector Switch	Actiontec	ALARM.COM	Alarm Decoder	Alarm Decoder Alarm Control Panel	Alarm Decoder Binary Sensor	Alarm Decoder Binary Sensor	Alarm Decoder Sensor	Alert	Amazon Echo	Alexa / Amazon Echo	Amazon Polly	AMCREST	Amcrest IP Camera	Amcrest IP Camera	Amcrest IP Camera	Android IP Webcam	Android IP Webcam Binary Sensor	Android IP Webcam Sensor	Arlo Camera	
ANEL PoE Switch	Aruba	APC UPS Daemon	APCUPSd	APCUPSd Binary Sensor	APCUPSd Sensor	API Stream Sensor	aptAI	Apple TV	APNS	ARDUINO	Arduino Sensor	ARDUSWITCH	Arduino Switch	aREST	aREST Binary Sensor	aREST Sensor	aREST Switch	Arlo	Arlo Camera	
Arlo Sensor	Aruba	ARVN Sensor	ASUSWRT	Aurora sensor	Automatic	Automation	AVION	AVM FRITZ!DECT Switch	AWS Lambda	AWS SNS	AWS SQS	AXIS FOUNDATIONS	Axis	Axis Binary Sensor	Axis Camera	B3OX	B3OX Sensor	BloomSky	BloomSky Binary Sensor	
BeagleBone Black GPIO	BeagleBone Black GPIO Binary Sensor	BeagleBone Black GPIO Switch	Belkin WeMo	Belkin WeMo Lights	Belkin WeMo Switch	BH1750 Light sensor	Bitcoin	Blink	Blink	Blink Camera	Blink Sensor	BLINKSTICK	BitVibes	BLOCKCHAIN	Blockchain.info	Blue Clock	Blue Spark	Buienradar	Buienradar Weather	
Bloomsky Camera	Bloomsky Sensor	Bluetooth	Bluetooth Lr Tracker	Bluetooth Tracker	BME280 Sensor	BOM Australia	BOM Australia Sensor	BroadLink RM Switch	BroadLink RM2 and A1 sensor	Browser	BT Home Hub 5	Buienradar	Buienradar Weather	Calor ID Sensor	Certificate Expiry	Cisco IOS	Cisco Spark	ComEd	ComEd Hourly Pricing	
CityBike API Sensor	Clymene Music Player	ClickSend	CMUS	CoMarketCap	Cometd	Command Line Binary Sensor	Command Line Cover	Command Line Notify	Command Line Sensor	Command Line Switch	Command Line Timer	Concord32 Alarm Control Panel	Concord32 Binary Sensor	Configurator	Conversation	CPU Speed	Cortex Monitor	Cortex Processing	Dark Sky	
Crime Reports	CUPSS Sensor	Currencylayer	D-Link	D-Link Switch	Datadog	DD-WRT	Decora	Denon AVR Network Receiver	DENON	Denon Network Receiver	DB	DHT Sensor	Digital Loggers Switch	Digital Ocean	Digital Ocean Binary Sensor	Digital Ocean Switch	Digital Oceanic	DIRECTV	DISCOVER	
Discord	Discovery	Dispatcher IP Camera	DLIB	Dlib Face Detect	Dlib Face Identify	DNS IP	Dowoda	Downloader	DSMR/Innme Meter	DT Energy Bridge	DTEL BUS	DTel Bus Transport	DTel Media Players	Dweet.io	Dweet.io History	dyson	dyson	Dyson Purifier Fan	dyson	
Dyson Purifier Sensor	EBOX	ecobee	Ecobee Binary Sensor	Ecobee Notify	Ecobee Smart Sensor	Ecobee Thermostat	Eddystone Beacon	Edimax Switch	Energy	Energy	Energy	ENERGY 8IGHT	Eight Sleep Sensor	elia	Emby	Emoncms History	Emoncms Sensor	Emulated Hue Bridge	EnOcean Sensor	
FedEx Sensor	Feedr	FFmpeg	FFmpeg Motion Binary Sensor	FFmpeg Noise Binary Sensor	FFmpeg Sensor	FIDO	Fido	File Sensor	File Sensor	FIRE TV	FIRE TV	Fitbit	Flexit A/C Controller	Flux LED Smart Light	Flux LED Light Adjustment	Facebook Messenger	fb	FAST	FOURSQUARE	
Foscam IP Camera	Free Mobile	FRANZ	FRITZ!Box	FRITZ!Box	Frontend	Frontier Silicon Radios	GARADGET	Generic MJPEG IP Camera	Generic Thermalstat	GITTER	Glances	GRTP (Grow)	Google Calendar Event	Google Cast	Google Calendar Event	GRTP (Grow)	HISTORY	HISTORY	History Statistics	
Google Maps Travel Time	Google Play Music Desktop Player	Google Text-to-Speech	GPSD	GPSLogger	GRAPHITE	GROUP	GSTREAMER	GTREAMER	Harmony Hub Remote	Have I Been Pwned?	HDMI	HEATMISER	HIKVISION	HIKVISION	Histary	Histary	Hydro Quebec	HYPERON	iCloud	
HomeMatic	Homematic Binary Sensor	Homematic Cover	Homematic Light	Homematic Sensor	Homematic Switch	Homematic Thermostats	Honeywell	Honeywell TotalConnect Alarm Control Panel	Honeywell Wireless Sensor	HTTP	HTTP Binary Sensor	HTTPD	INFLUXDB	InfluxDB Sensor	INSTAPUSH	INSTEON (Local)	INSTEON (Local) Fan	INSTEON (Local) Light	INSTEON (Local) Switch	INSTEON Hub
IKEA Tradfri	IKEA Tradfri Lights	IMAP	Unred E-Mail	INFUSIONDB	INFUSIONDB	INPUT BOOLEAN	INPUT SELECT	INPUT SLIDER	INSTEON (Local)	Instapush	Instapush	INSTEON (Local) Fan	Instapush	Instapush	Instapush	Instapush	Instapush	Instapush	Instapush	Instapush

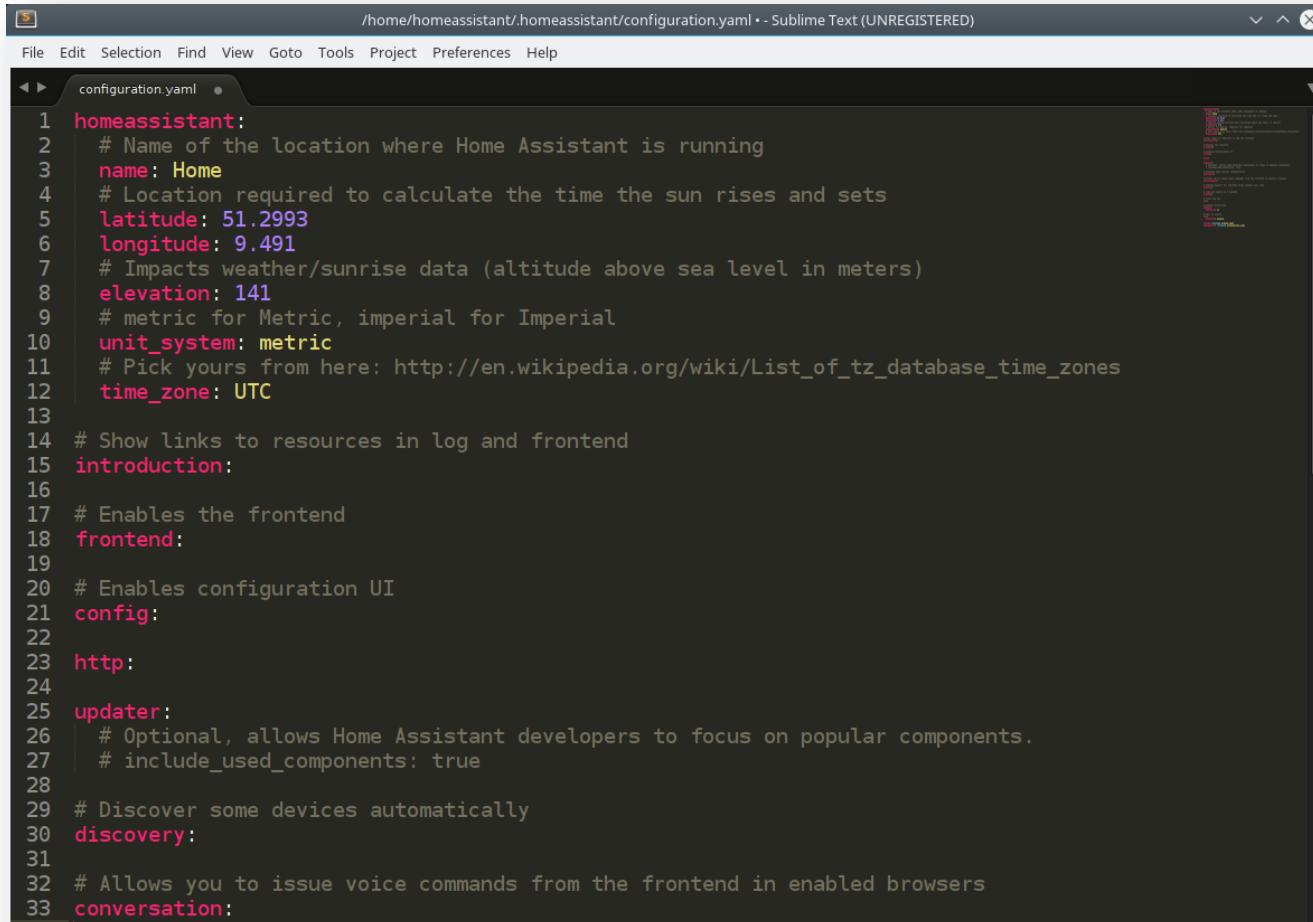
# Installation

```
sebastian : bash — Konsole
Datei Bearbeiten Ansicht Lesezeichen Einstellungen Hilfe
(homeassistant) homeassistant@amy:~/homeassistant$ 
(homeassistant) homeassistant@amy:~/homeassistant$ 
(homeassistant) homeassistant@amy:~/homeassistant$ 
(homeassistant) homeassistant@amy:~/homeassistant$ pip3 install --upgrade homeassistant
Collecting homeassistant
  Downloading homeassistant-0.49.0-py2.py3-none-any.whl (7.2MB)
    100% |██████████| 7.2MB 219kB/s
Collecting requests==2.14.2 (from homeassistant)
  Downloading requests-2.14.2-py2.py3-none-any.whl (560kB)
    100% |██████████| 563kB 1.3MB/s
Collecting aiohttp==2.2.3 (from homeassistant)
  Downloading aiohttp-2.2.3-cp35-cp35m-manylinux1_x86_64.whl (758kB)
    100% |██████████| 768kB 1.5MB/s
Collecting pytz>=2017.02 (from homeassistant)
  Downloading pytz-2017.2-py2.py3-none-any.whl (484kB)
    100% |██████████| 491kB 2.4MB/s
Collecting voluptuous==0.10.5 (from homeassistant)
  Downloading voluptuous-0.10.5.tar.gz (41kB)
    100% |██████████| 51kB 4.0MB/s
Collecting astral==1.4 (from homeassistant)
  Downloading astral-1.4-py2.py3-none-any.whl
Collecting chardet==3.0.4 (from homeassistant)
  Downloading chardet-3.0.4-py2.py3-none-any.whl (133kB)
    100% |██████████| 143kB 3.8MB/s
Requirement already up-to-date: pip>=7.1.0 in ./lib/python3.5/site-packages (from homeassistant)
Collecting typing<4,>=3 (from homeassistant)
  Downloading typing-3.6.1.tar.gz (66kB)
```

# Erster Start



# Konfiguration



The screenshot shows a Sublime Text editor window displaying the `configuration.yaml` file for Home Assistant. The file is a YAML configuration document with the following content:

```
1 homeassistant:
2   # Name of the location where Home Assistant is running
3   name: Home
4   # Location required to calculate the time the sun rises and sets
5   latitude: 51.2993
6   longitude: 9.491
7   # Impacts weather/sunrise data (altitude above sea level in meters)
8   elevation: 141
9   # metric for Metric, imperial for Imperial
10  unit_system: metric
11  # Pick yours from here: http://en.wikipedia.org/wiki/List_of_tz_database_time_zones
12  time_zone: UTC
13
14 # Show links to resources in log and frontend
15 introduction:
16
17 # Enables the frontend
18 frontend:
19
20 # Enables configuration UI
21 config:
22
23 http:
24
25 updater:
26   # Optional, allows Home Assistant developers to focus on popular components.
27   # include_used_components: true
28
29 # Discover some devices automatically
30 discovery:
31
32 # Allows you to issue voice commands from the frontend in enabled browsers
33 conversation:
```

# Komponenten finden

 **Home Assistant**

Getting started Components Docs Examples Developers Blog Need help? 

## // Ping (ICMP) Binary Sensor

---

The `ping` binary sensor platform allows you to use `ping` to send ICMP echo requests. This way you can check if a given host is online and determine the round trip times from your Home Assistant instance to that system.

To use this sensor in your installation, add the following to your `configuration.yaml` file:

```
# Example configuration.yaml entry
binary_sensor:
  - platform: ping
    host: 192.168.0.1
```

Configuration variables:

- **host (Required)**: The IP address or hostname of the system you want to track.
- **count (Optional)**: Number of packets to send. Defaults to 5.
- **name (Optional)**: Let you overwrite the name of the device. Defaults to `Ping Binary sensor`.

The sensor exposes the different round trip times values measured by `ping` as attributes:

- `round_trip_time_mdev`
- `round_trip_time_avg`
- `round_trip_time_min`
- `round_trip_time_max`

**Note**

When run on Windows systems, the round trip time attributes are rounded to the

Edit this page on GitHub 

Introduced in release: 0.43

This is a platform for the [Binary Sensor component](#).

### Related components

[Ping \(ICMP\)](#)

**Category** [Binary Sensor](#)

[AlarmDecoder Binary Sensor](#)

[Android IP Webcam Binary Sensor](#)

[Aurora sensor](#)

[Axis Binary Sensor](#)

[BeagleBone Black GPIO Binary Sensor](#)

[Blink Binary Sensor](#)

[BloomSky Binary Sensor](#)

[Command line Binary Sensor](#)

[Concord232 Binary Sensor](#)

[Ecobee Binary Sensor](#)

[Ecobee Sensor](#)

[Eight Sleep Binary Sensor](#)

[EnOcean Binary Sensor](#)

[Envisalink Binary Sensor](#)

[FFmpeg Motion Binary Sensor](#)

# Komponenten finden

 **Home Assistant**

Getting started Components Docs Examples Developers Blog Need help? 

Edit this page on GitHub 

## // Ping (ICMP) Binary Sensor

---

The `ping` binary sensor platform allows you to make requests. This way you can check if a device is online or not by sending ICMP requests from your Home Assistant instance.

To use this sensor in your installation, add it to your `configuration.yaml` file:

```
# Example configuration.yaml entry
binary_sensor:
  - platform: ping
    host: 192.168.0.1
```

Configuration variables:

- **host (Required)**: The IP address of the device to ping.
- **count (Optional)**: Number of ICMP requests to send.
- **name (Optional)**: Let you override the name of the binary sensor.

The sensor exposes the different route attributes:

- `round_trip_time_mdev`
- `round_trip_time_avg`
- `round_trip_time_min`
- `round_trip_time_max`

**Note**  
When run on Windows systems, the

 **Home Assistant**

Getting started Components Docs Examples Developers Blog Need help? 

Edit this page on GitHub 

## // Owntracks

---

This platform allows you to detect presence using [Owntracks](#). OwnTracks allows users to track their location on Android and iOS phones and publish it to an MQTT broker. This platform will connect to the broker and monitor for new locations.

This component requires [the MQTT component](#) to be set up and works very well together with [the zone component](#).

To integrate Owntracks in Home Assistant, add the following section to your `configuration.yaml` file:

```
# Example configuration.yaml entry
device_tracker:
  - platform: owntracks
```

Configuration variables:

- **max\_gps\_accuracy (Optional)**: Sometimes Owntracks can report GPS location with a very low accuracy (few kilometers). That can trigger false zoning in your Home Assistant installation. With the parameter, you can filter these GPS reports. The number has to be in meter. For example, if you put 200 only GPS report with an accuracy under 200 will be taken in account.

Introduced in release: 0.7.4  
This is a platform for the [Device Tracker component](#).

### Category Presence Detection

- ASUSWRT
- Actiontec
- Aruba
- Automatic
- BT Home Hub 5
- Bbox
- Bluetooth LE Tracker
- Bluetooth Tracker
- Cisco IOS
- DD-WRT
- FRITZ!Box

# Komponenten finden

**Home Assistant**

[Getting started](#) [Components](#)

## // Ping (ICMP) Binary Sensor

The `ping` binary sensor platform allows you to make requests. This way you can check if certain services are available at specific times from your Home Assistant instance.

To use this sensor in your installation file:

```
# Example configuration.yaml entry
binary_sensor:
  - platform: ping
    host: 192.168.0.1
```

Configuration variables:

- **host (Required)**: The IP address of the device to ping.
- **count (Optional)**: Number of requests to send.
- **name (Optional)**: Let you override the name of the binary sensor.

The sensor exposes the different route attributes:

- `round_trip_time_mdev`
- `round_trip_time_avg`
- `round_trip_time_min`
- `round_trip_time_max`

**Note**

When run on Windows systems, the

**Home Assistant**

[Getting started](#) [Components](#) [Docs](#) [Examples](#) [Developers](#) [Blog](#) [Need help?](#) [GitHub](#)

## // FRITZ!Box

The `fritz` platform offers presence detection by looking at connected devices to a [AVM Fritz!Box](#) based router.

**Warning**

It might be necessary to install additional packages: `$ sudo apt-get install libxml2-dev libxml2-dev python3-lxml` If you are working with the All-in-One installation, you may also need to execute also within your virtual environment the command `pip install lxml`; be patient this will take a while.

To use an Fritz!Box router in your installation, add the following to your `configuration.yaml` file:

```
# Example configuration.yaml entry
device_tracker:
  - platform: fritz
```

Configuration variables:

- **host (Optional)**: The IP address of your router, e.g. `192.168.1.1`. It is optional since every fritzbox is also reachable by using the IP address `169.254.1.1`.
- **username (Optional)**: The username of an user with administrative privileges, usually `admin`.
- **password (Optional)**: The password for your given admin account.

**Note**

It seems that it is not necessary to use it in current generation Fritz!Box routers because the necessary data can be retrieved anonymously.

**AVM**

Introduced in release: 0.10

This is a platform for the [Device Tracker component](#).

**Category Presence Detection**

[ASUSWRT](#)

[Actiontec](#)

[Aruba](#)

[Automatic](#)

[BT Home Hub 5](#)

[Bbox](#)

[Bluetooth LE Tracker](#)

[Bluetooth Tracker](#)

[Cisco IOS](#)

[DD-WRT](#)

[FRITZ!Box](#)

[GPSLogger](#)

[JSON MQTT Device Tracker](#)

[Linksys Access Points](#)

[Linksys Smart Wifi Router](#)

[Locative](#)

# Komponenten finden

 **Home Assistant**

Getting started Components Docs Examples Developers **Blog** Need help? 

[Edit this page on GitHub](#)

## // Deutsche Bahn

The `deutsche_bahn` sensor will give you the departure time of the next train for the given connection. In case of a delay, the delay is also shown. Additional details are used to inform about eg. the type of the train, price, and if it is on time.

To enable this sensor, add the following lines to your `configuration.yaml` file:

```
# Example configuration.yaml entry
sensor:
  - platform: deutsche_bahn
    from: NAME_OF_START_STATION
    to: NAME_OF_FINAL_STATION
```

Configuration variables:

- **from (Required)**: The name of the start station.
- **to (Required)**: The name of the end/destination station.

This sensor stores a lot of attributes which can be accessed by other sensors eg. a [template sensor](#).

```
# Example configuration.yaml entry
sensor:
  platform: template
  sensors:
```

**DB**  
IoT class: Cloud Polling  
Introduced in release: 0.14

### Category Transport

Deutsche Bahn

Dublin Bus Transport

Google Maps Travel Time

London Undergound

Lyft Sensor

MVG

Public Transit (GTFS)

Swiss Public Transport

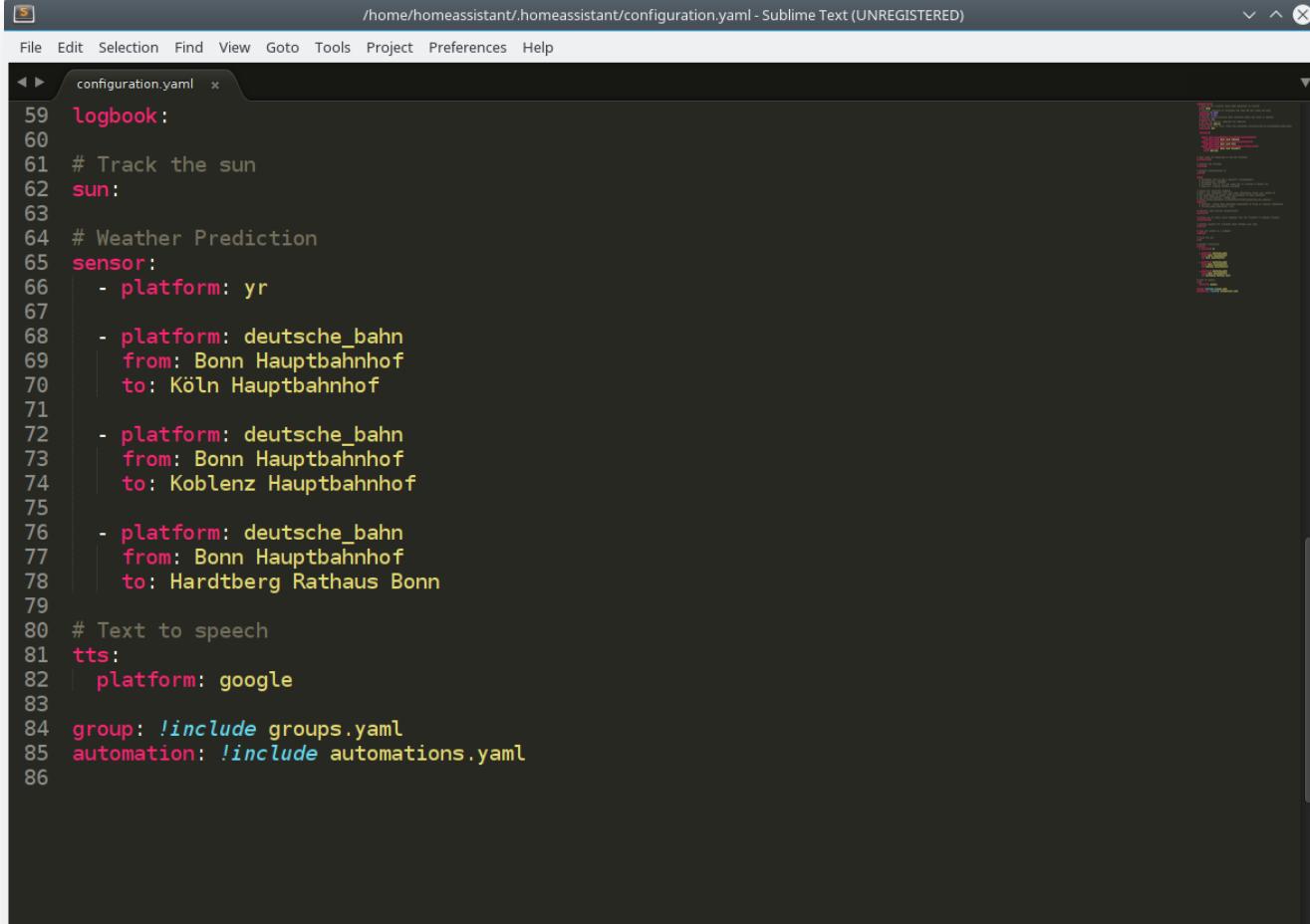
Torque (OBD2)

Uber

Västtrafik Public Transport

Washington State DOT

# Komponente laden



The screenshot shows a Sublime Text editor window with a dark theme. The title bar reads "/home/homeassistant/.homeassistant/configuration.yaml - Sublime Text (UNREGISTERED)". The menu bar includes File, Edit, Selection, Find, View, Goto, Tools, Project, Preferences, and Help. A tab bar at the top left shows "configuration.yaml x". The main editor area contains the following YAML configuration:

```
59 logbook:
60
61 # Track the sun
62 sun:
63
64 # Weather Prediction
65 sensor:
66   - platform: yr
67
68   - platform: deutsche_bahn
69     from: Bonn Hauptbahnhof
70     to: Köln Hauptbahnhof
71
72   - platform: deutsche_bahn
73     from: Bonn Hauptbahnhof
74     to: Koblenz Hauptbahnhof
75
76   - platform: deutsche_bahn
77     from: Bonn Hauptbahnhof
78     to: Hardtberg Rathaus Bonn
79
80 # Text to speech
81 tts:
82   - platform: google
83
84 group: !include groups.yaml
85 automation: !include automations.yaml
86
```

# Darstellung in der Web-Oberfläche

The screenshot displays the Home Assistant web interface with a sidebar on the left and a main content area on the right.

**Left Sidebar:**

- States** (selected)
- Map**
- Logbook**
- History**
- Automations**
- Configuration**
- Log Out**
- Push Notifications** (switched off)
- Developer Tools**

**Main Content Area:**

**Top Header:** Home Assistant < Home

**Top Bar:** Sun 13:27 Bonn Hauptbahnhof to Koblenz 13:27 Bonn Hauptbahnhof to Köln 13:32 yr Symbol

**Card:** Welcome Home! (with a list of resources)

**Card:** Bonn Hauptbahnhof to Hardtberg 1 minute ago (with a timeline and a table of travel details)

# Gruppieren

The screenshot shows the Home Assistant interface. On the left, a Sublime Text window displays a YAML configuration file:

```
1
2
3 all_transports:
4   name: Öffentlicher Personennahverkehr
5   entities:
6     - sensor.bonn_hauptbahnhof_to_koblenz_hauptbahnhof
7     - sensor.bonn_hauptbahnhof_to_koln_hauptbahnhof
8     - sensor.bonn_hauptbahnhof_to_hardtberg_rathaus_bonn
9
10 |
```

The main interface shows a weather forecast for "Sun" and "Cloudy" conditions. A sidebar titled "Öffentlicher Personennahverkehr" lists three transport events:

Event	From	To	Time
1	Bonn Hauptbahnhof	Koblenz Hauptbahnhof	13:27
2	Bonn Hauptbahnhof	Köln Hauptbahnhof	13:32
3	Bonn Hauptbahnhof	Hardtberg Rathaus Bonn	13:20

# Individualisieren

The screenshot shows the Home Assistant interface with a configuration file open in Sublime Text and the Home screen displayed.

**Sublime Text (Left):**

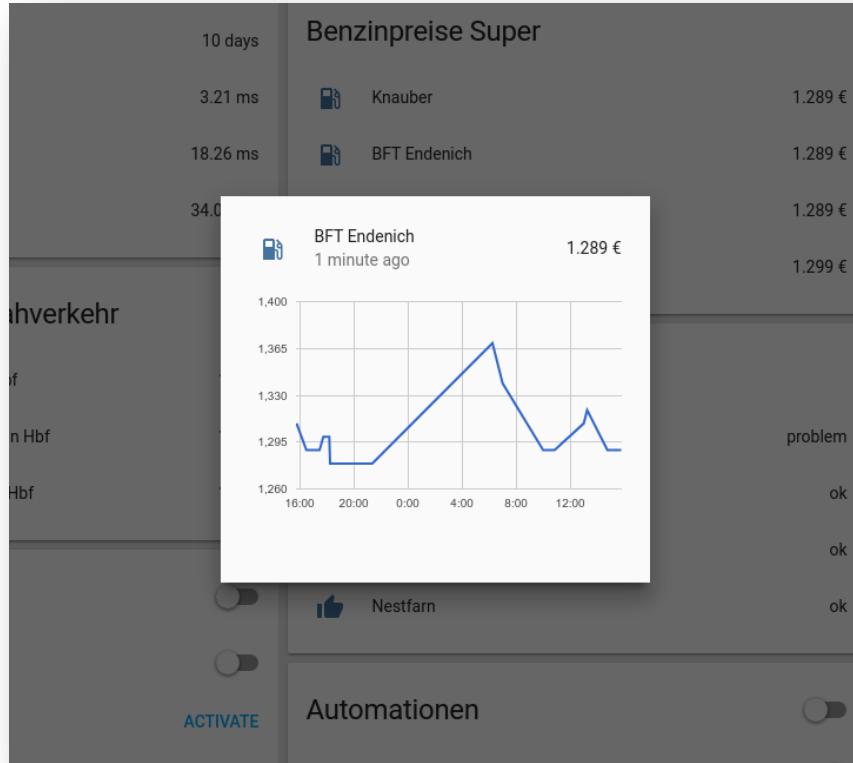
```
File Edit Selection Find View Goto Tools Project Preferences Help
configuration.yaml groups.yaml

1 homeassistant:
2   # Name of the location where Home Assistant is running
3   name: Home
4   # Location required to calculate the time the sun rises and sets
5   latitude: 51.2993
6   longitude: 9.491
7   # Impacts weather/sunrise data (altitude above sea level in meters)
8   elevation: 141
9   # metric for Metric, imperial for Imperial
10  unit_system: metric
11  # Pick yours from here: http://en.wikipedia.org/wiki/List_of_tz_database_time_zones
12  time_zone: UTC
13
14 customize:
15
16   sensor.bonn_hauptbahnhof_to_koblenz_hauptbahnhof:
17     friendly_name: Bonn nach Koblenz
18   sensor.bonn_hauptbahnhof_to_koeln_hauptbahnhof:
19     friendly_name: Bonn nach Köln
20   sensor.bonn_hauptbahnhof_to_hardtberg_rathaus_bonn:
21     friendly_name: Bonn nach Duisdorf
22     icon: mdi:bus
23
24 # Show links to resources in log and frontend
25 introduction:
26
27 # Enables the frontend
28 frontend:
29
30 # Enables configuration UI
31 config:
32
33
34 http:
35   # Uncomment this to add a password (recommended!)
36   # api_password: PASSWORD
37   # Uncomment this if you are using SSL or running in Docker etc
```

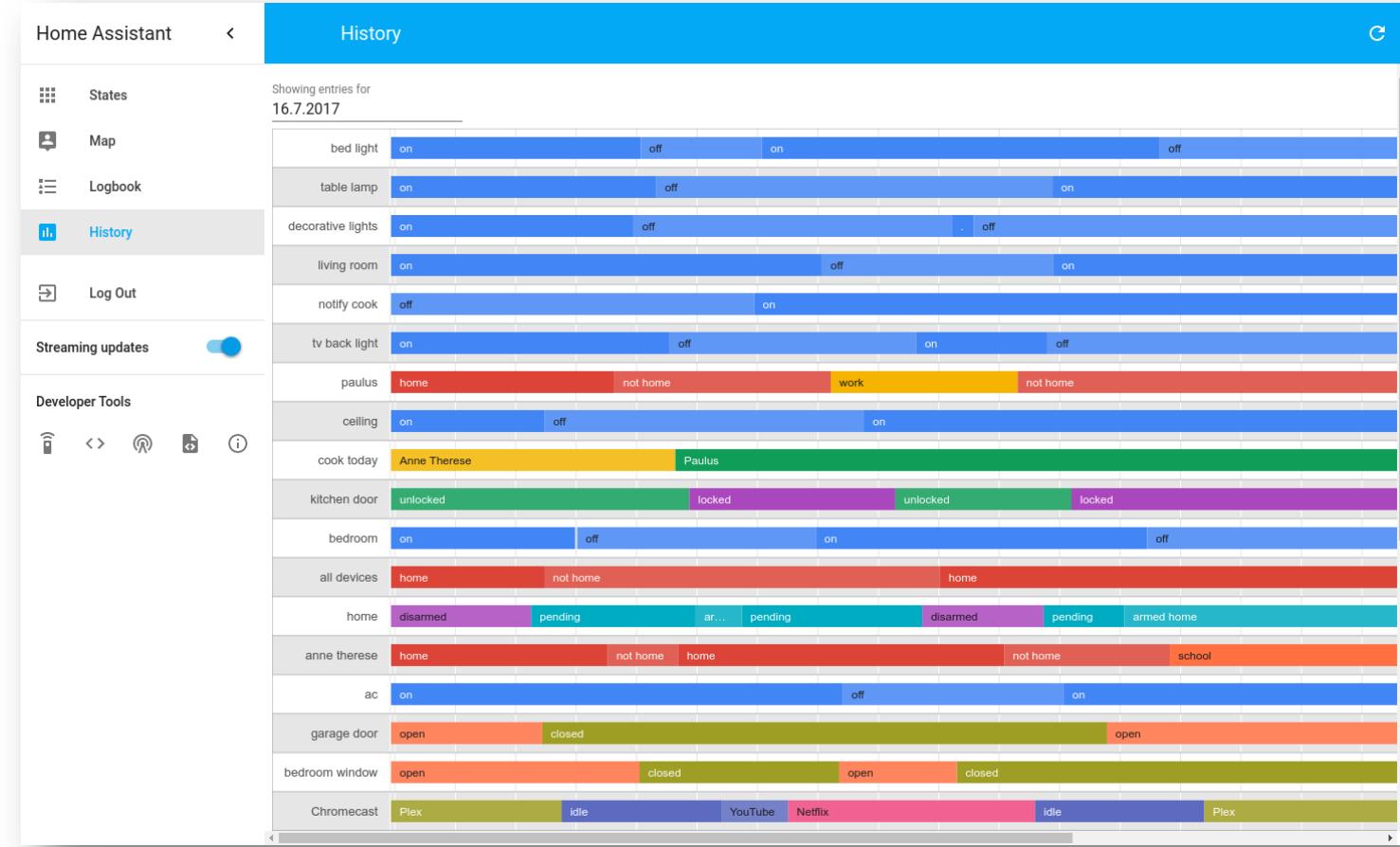
**Home Screen (Right):**

- Sun and Cloud icons
- Symbol
- Öffentlicher Personennahverkehr
  - Bonn nach Koblenz 13:27
  - Bonn nach Köln 13:32
  - Bonn nach Duisdorf 13:27

# Zahlenwerte



# Historie der Zustände



# Werbung

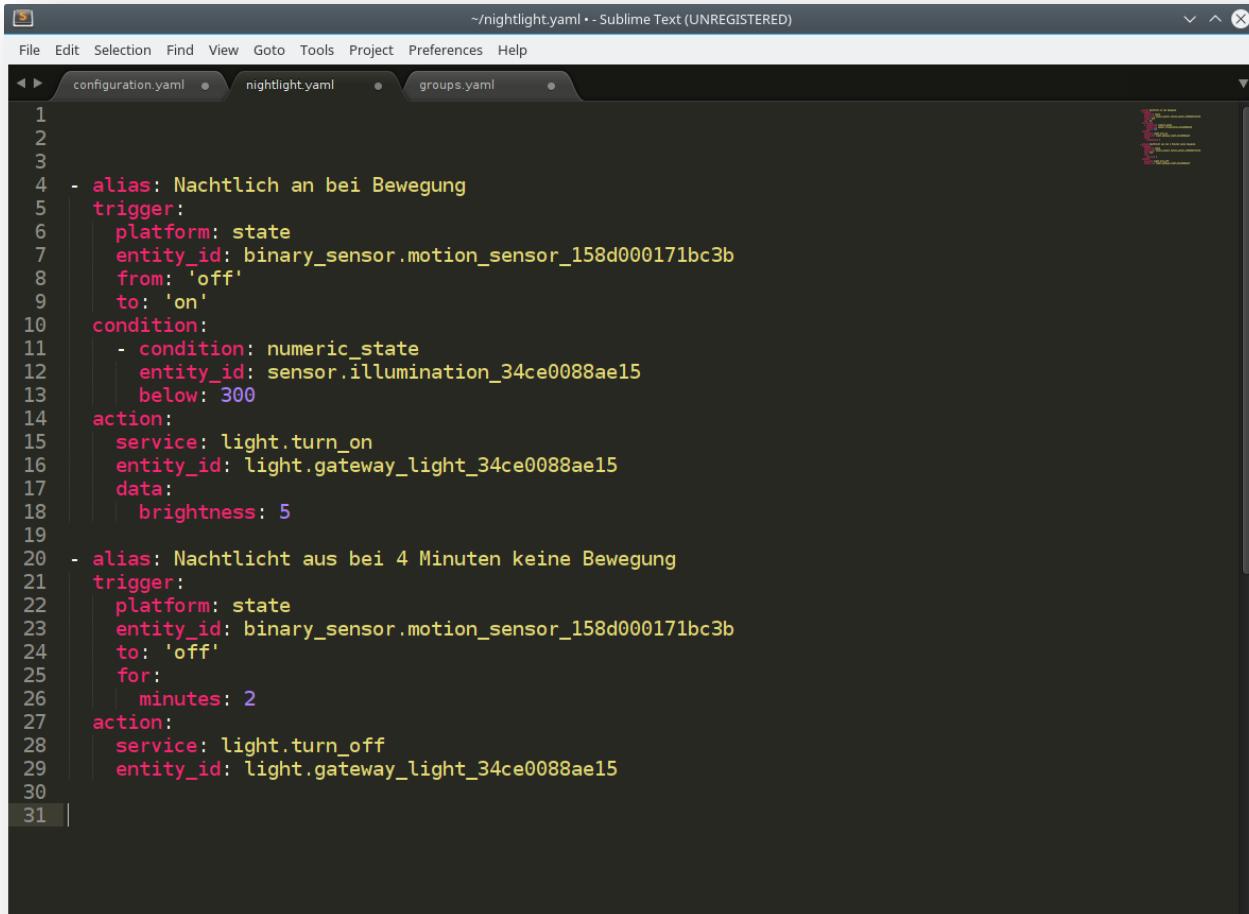
Xiaomi Aqara



Ikea Tradfri



# Automation: Bewegungsmelder

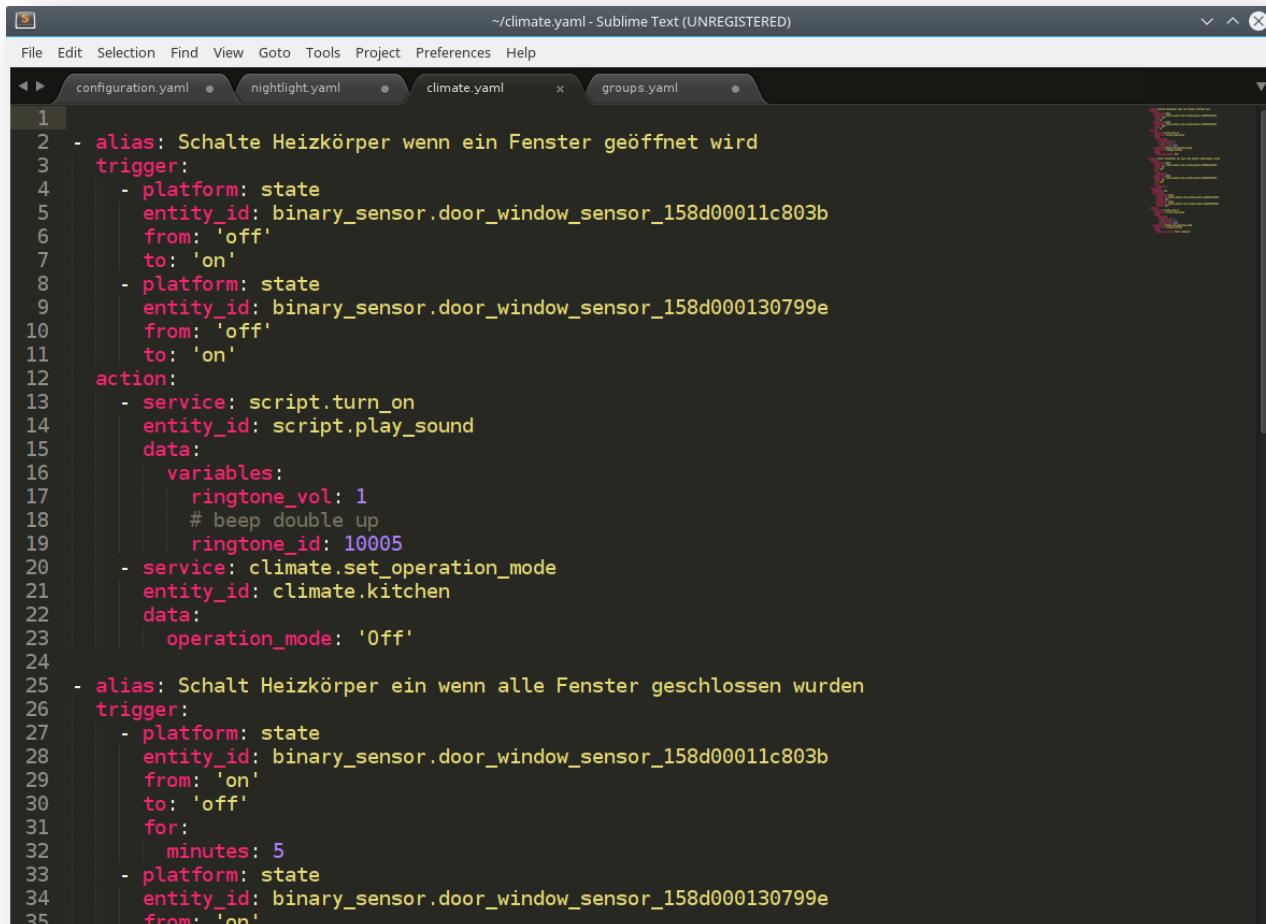


The screenshot shows a Sublime Text window with three tabs open: 'configuration.yaml', 'nightlight.yaml', and 'groups.yaml'. The 'nightlight.yaml' tab is active and displays the following YAML code:

```
~/nightlight.yaml - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
configuration.yaml • nightlight.yaml • groups.yaml •

1
2
3
4 - alias: Nachtlich an bei Bewegung
  trigger:
    platform: state
    entity_id: binary_sensor.motion_sensor_158d000171bc3b
    from: 'off'
    to: 'on'
  condition:
    - condition: numeric_state
      entity_id: sensor.illumination_34ce0088ae15
      below: 300
  action:
    service: light.turn_on
    entity_id: light.gateway_light_34ce0088ae15
    data:
      brightness: 5
19
20 - alias: Nachtlicht aus bei 4 Minuten keine Bewegung
21   trigger:
22     platform: state
23     entity_id: binary_sensor.motion_sensor_158d000171bc3b
24     to: 'off'
25     for:
26       minutes: 2
27   action:
28     service: light.turn_off
29     entity_id: light.gateway_light_34ce0088ae15
30
31
```

# Automation: Heizung aus bei offenem Fenster



The screenshot shows a Sublime Text window with the title bar reading "/climate.yaml - Sublime Text (UNREGISTERED)". The window displays a YAML configuration file for Home Assistant automation. The code is color-coded, with comments in German and some English words like 'alias', 'trigger', 'platform', 'entity\_id', 'from', 'to', 'action', 'service', 'script', 'play\_sound', 'variables', 'ringtones', 'climate.set\_operation\_mode', and 'operation\_mode'. The file contains two main automation rules:

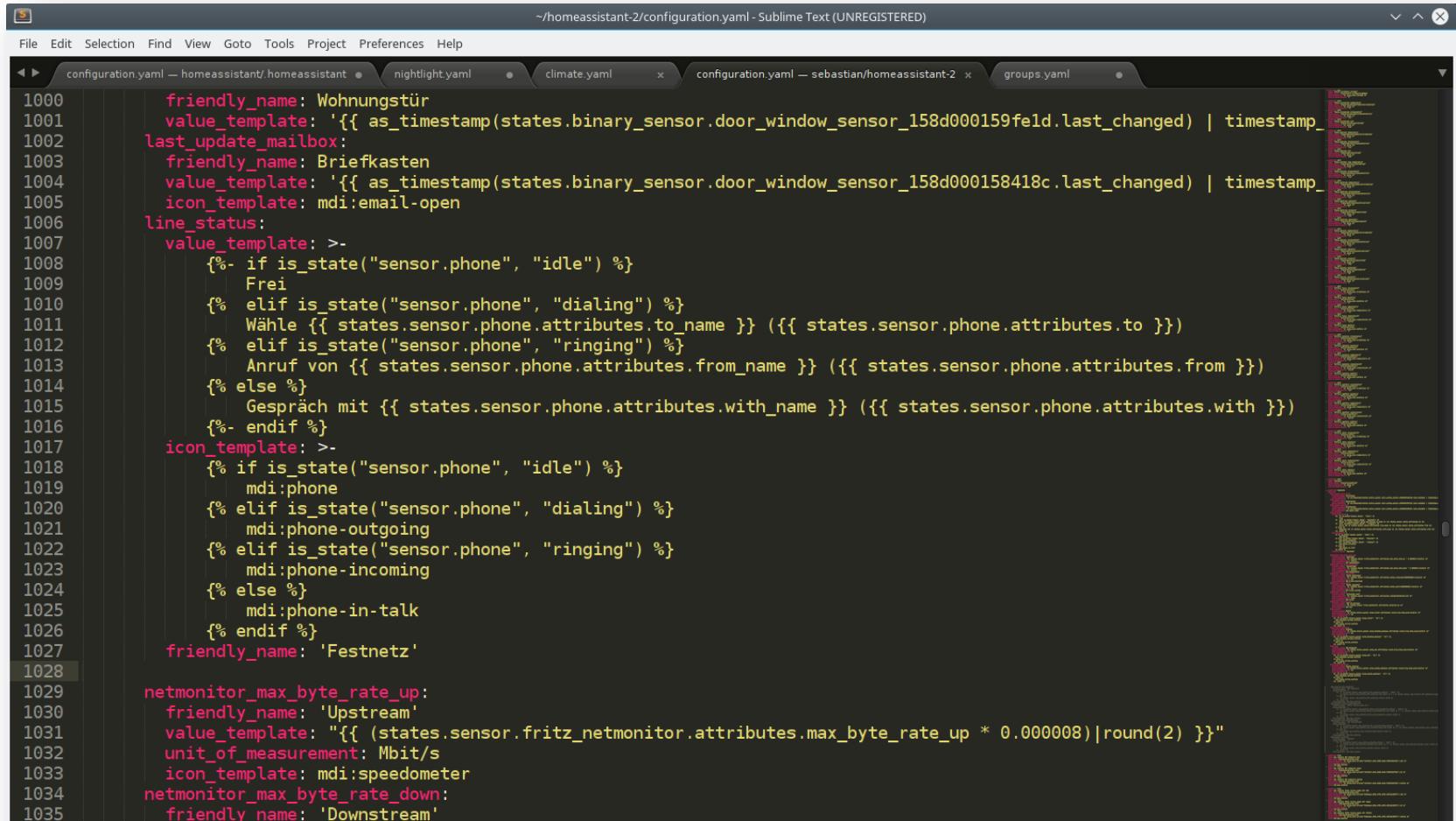
```
1
2 - alias: Schalte Heizkörper wenn ein Fenster geöffnet wird
  trigger:
  - platform: state
    entity_id: binary_sensor.door_window_sensor_158d00011c803b
    from: 'off'
    to: 'on'
  - platform: state
    entity_id: binary_sensor.door_window_sensor_158d000130799e
    from: 'off'
    to: 'on'
  action:
  - service: script.turn_on
    entity_id: script.play_sound
    data:
      variables:
        ringtones:
          - ringtones_id: 10005
            # beep double up
            ringtones_vol: 1
  - service: climate.set_operation_mode
    entity_id: climate.kitchen
    data:
      operation_mode: 'Off'
  alias: Schalt Heizkörper ein wenn alle Fenster geschlossen wurden
  trigger:
  - platform: state
    entity_id: binary_sensor.door_window_sensor_158d00011c803b
    from: 'on'
    to: 'off'
    for:
    - minutes: 5
  - platform: state
    entity_id: binary_sensor.door_window_sensor_158d000130799e
    from: 'on'
```

# Virtuelle Sensoren: Templates

The screenshot shows a Sublime Text window with multiple tabs open. The active tab is 'configuration.yaml' located at `~/homeassistant-2/configuration.yaml`. The code displays Jinja2 templates for creating virtual sensors. It includes sections for 'netmonitor\_uptime', 'ping\_router', and 'ping\_telekom\_gateway'. Each section defines a friendly name, value template, unit of measurement, and icon template. The 'ping\_router' section uses an if-else conditional to determine the icon based on the state of the binary sensor.

```
1049     netmonitor_uptime:
1050         friendly_name: 'Verbunden seit'
1051         value_template: "{{ (states.sensor.fritz_netmonitor.attributes.uptime/60/60/24)|int }}"
1052         unit_of_measurement: days
1053         icon_template: mdi:clock
1054
1055     netmonitor_external_ip:
1056         friendly_name: 'WAN IP-Adresse'
1057         value_template: "{{ states.sensor.fritz_netmonitor.attributes.external_ip }}"
1058         icon_template: mdi:wan
1059
1060
1061 ping_router:
1062     friendly_name: Router
1063     value_template: "{{ states.binary_sensor.ping_router.attributes.round_trip_time_avg|round(2) }}"
1064     unit_of_measurement: ms
1065     icon_template: >-
1066         {% if is_state("binary_sensor.ping_router", "on") %}
1067             mdi:checkbox-marked-outline
1068         {% else %}
1069             mdi:close-circle-outline
1070         {% endif %}
1071
1072
1073
1074 ping_telekom_gateway:
1075     friendly_name: Telekom
1076     value_template: "{{ states.binary_sensor.ping_telekom_gateway.attributes.round_trip_time_avg|round(2) }}"
1077     unit_of_measurement: ms
1078     icon_template: >-
1079         {% if is_state("binary_sensor.ping_telekom_gateway", "on") %}
1080             mdi:checkbox-marked-outline
1081         {% else %}
1082             mdi:close-circle-outline
1083         {% endif %}
1084
ping_vpn:
```

# Virtuelle Sensoren: Templates



The screenshot shows a Sublime Text window with multiple tabs open. The active tab is 'configuration.yaml' located at '~/homeassistant-2/configuration.yaml'. The code in this tab defines various virtual sensors using YAML syntax. The configuration includes sections for door/window sensors, a phone sensor, and network monitoring.

```
1000 friendly_name: Wohnungstür
1001 value_template: '{{ as_timestamp(states.binary_sensor.door_window_sensor_158d000159feld.last_changed) | timestamp_
1002 last_update_mailbox:
1003 friendly_name: Briefkasten
1004 value_template: '{{ as_timestamp(states.binary_sensor.door_window_sensor_158d000158418c.last_changed) | timestamp_
1005 icon_template: mdi:email-open
1006 line_status:
1007     value_template: >-
1008         %- if is_state("sensor.phone", "idle") %
1009             Frei
1010         %- elif is_state("sensor.phone", "dialing") %
1011             Wähle {{ states.sensor.phone.attributes.to_name }} ({{ states.sensor.phone.attributes.to }})
1012         %- elif is_state("sensor.phone", "ringing") %
1013             Anruf von {{ states.sensor.phone.attributes.from_name }} ({{ states.sensor.phone.attributes.from }})
1014         %- else %
1015             Gespräch mit {{ states.sensor.phone.attributes.with_name }} ({{ states.sensor.phone.attributes.with }})
1016         %- endif %
1017     icon_template: >-
1018         %- if is_state("sensor.phone", "idle") %
1019             mdi:phone
1020         %- elif is_state("sensor.phone", "dialing") %
1021             mdi:phone-outgoing
1022         %- elif is_state("sensor.phone", "ringing") %
1023             mdi:phone-incoming
1024         %- else %
1025             mdi:phone-in-talk
1026         %- endif %
1027     friendly_name: 'Festnetz'
1028
1029 netmonitor_max_byte_rate_up:
1030     friendly_name: 'Upstream'
1031     value_template: "{{ (states.sensor.fritz_netmonitor.attributes.max_byte_rate_up * 0.000008)|round(2) }}"
1032     unit_of_measurement: Mbit/s
1033     icon_template: mdi:speedometer
1034 netmonitor_max_byte_rate_down:
1035     friendly_name: 'Downstream'
```

# MQTT

The screenshot shows the Home Assistant web interface with the MQTT integration. The top navigation bar includes sections for HOME, KÜCHE, WOHNZIMMER, SCHLAFZIMMER, BADEZIMMER, BALKON, and FLUR. The BALKON section is currently active, displaying sensor data for the balcony. The CHILI section is also visible, showing data from a sensor connected to a device named 'Chili'.

**Balkon Sensors:**

- Außenbeleuchtung (Light): Off
- Bewässerung (Irrigation): Off
- Helligkeit (Light): 1728.50 lux
- Temperatur (Temperature): 35.55 °C
- Luftfeuchtigkeit (Humidity): 35.67 %
- Luftdruck (Pressure): 1012.74 hPa
- Bodenfeuchtigkeit (Soil Moisture): 32 %
- Hitzeindex (Heat Index): 36.94 HI

**Chili Sensors:**

- Temperatur (Temperature): 23.3 °C
- Bodenfeuchtigkeit (Soil Moisture): 66 %
- Helligkeit (Light): 13327 lux
- Leitfähigkeit (Conductivity): 655 us/cm
- Batterie (Battery): 100 %
- Chili (Status): problem

# MQTT

~/homeassistant-2/configuration.yaml - Sublime Text (UNREGISTERED)

File Edit Selection Find View Goto Tools Project Preferences Help

configuration.yaml — homeassistant/homeassistant • nightlight.yaml • climate.yaml • — sebastian •

```
863 sensor:
864   - platform: mqtt
865     name: "homie-balcony temperature"
866     state_topic: "homie/balcony/temperature/degrees"
867     value_template: "{{ value }}"
868     unit_of_measurement: "°C"
869   - platform: mqtt
870     name: "homie-balcony illumination"
871     state_topic: "homie/balcony/illumination/lux"
872     value_template: "{{ value }}"
873     unit_of_measurement: "lux"
874   - platform: mqtt
875     name: "homie-balcony humidity"
876     state_topic: "homie/balcony/humidity/percent"
877     value_template: "{{ value }}"
878     unit_of_measurement: "%"
879   - platform: mqtt
880     name: "homie-balcony pressure"
881     state_topic: "homie/balcony/pressure/hpa"
882     value_template: "{{ value }}"
883     unit_of_measurement: "hPa"
884   - platform: mqtt
885     name: "homie-balcony moisture"
886     state_topic: "homie/balcony/moisture/percent"
887     value_template: "{{ value }}"
888     unit_of_measurement: "%"
889
890 switch:
891   - platform: mqtt
892     name: "homie-balcony watering"
893     state_topic: "homie/balcony/watering/on"
894     command_topic: "homie/balcony/watering/on/set"
895     qos: 1
896     payload_on: "true"
897     payload_off: "false"
898     retain: true
```

Line 898, Column 1      Spaces: 2      YAML

# MQTT

The screenshot shows two open files in Sublime Text:

- File 1 (Left):** configuration.yaml (homeassistant/homeassistant)
- File 2 (Right):** configuration.yaml (sebastian)

The configuration files define MQTT entities for a balcony sensor. The entities include:

- sensor.homiebalcony\_temperature:** A temperature sensor with icon "mdi:thermometer" and friendly name "Temperatur".
- sensor.homiebalcony\_illumination:** An illumination sensor with icon "mdi:weather-sunset" and friendly name "Helligkeit".
- sensor.homiebalcony\_pressure:** A pressure sensor with icon "mdi:water-percent" and friendly name "Luftdruck".
- sensor.homiebalcony\_humidity:** A humidity sensor with icon "mdi:water-percent" and friendly name "Luftfeuchtigkeit".
- sensor.homiebalcony\_heatindex:** A heat index sensor with icon "mdi:water-percent" and friendly name "Hitzeindex".
- sensor.homiebalcony\_moisture:** A moisture sensor with icon "mdi:water-percent" and friendly name "Bodenfeuchtigkeit".
- switch.homiebalcony\_watering:** A switch entity for watering with icon "mdi:water" and friendly name "Bewässerung".

Each entity is defined with an MQTT topic prefix ("homie-balcony") and specific topic paths (e.g., "temperature/degrees" for temperature). The configuration uses Jinja templating for the MQTT topic and message template.

# MQTT

The screenshot shows a Sublime Text window with four tabs open, each displaying a YAML configuration file for Home Assistant:

- File 1: configuration.yaml (Line 372 to 407)
- File 2: groups.yaml (Line 166 to 201)
- File 3: nightlight.yaml (Large portion visible)
- File 4: climate.yaml (Large portion visible)

The configuration files define various entities, sensors, and switches, primarily for a balcony environment. The groups.yaml file defines a group named "balcony" containing entities like "group.balcony\_environment" and "group.miflora\_chili". The configuration.yaml file includes definitions for sensors and switches related to the balcony, such as "sensor.homiebalco" and "switch.homiebalco". The nightlight.yaml and climate.yaml files contain extensive definitions for climate control and nightlighting, including entities like "climate.therm" and "climate.humidity".

# Erweiterbarkeit

- AppDaemon
- Homebridge
- Haaska – Home Assistant Alexa Skill Adapter (AWS Lambda Function)
- Eigene Komponenten

# Vielen Dank!

Fragen?

Sebastian Muszynski, @syssi84  
s.muszynski@constructiva.de

17.07.2017