

Fortgeschritten

wenn du die Basics verstanden hast kannst du dich an diese Schritte machen.

- [Cloudflare](#)
- [Alexa ohne Cloud](#)
- [Bluetooth Repeater - Reichweite erhöhen](#)
- [Configuration.YAML strukturieren](#)
- [MQTT Broker](#)
- [Zigbee2mqtt](#)
- [Homeassistant App Features:](#)

Cloudflare

Gratis Domain beantragen

Freenom gibt es so leider nicht mehr.

Alternativ kann man sich bei Serverprofis eine Domain für etwa 8€ im Jahr holen.

Achtet hier darauf das ihr das DNS Paket mit nehmt.

Cloudflare Account

Erstell dir einen Cloudflare Account



Addon Repo für Addon Store

Füge die Repository unter Addons in Homeassistant ein:

<https://github.com/brenner-tobias/ha-addons>

Konfiguration in Homeassistant

Konfiguration in Homeassistant

```
http:
  use_x_forwarded_for: true
  trusted_proxies:
    - 172.30.33.0/24
ip_ban_enabled: true      ## IP Ban aktivieren - falls ihr euch selbst sperrt einfach den Eintrag in der IP
Ban YAML löschen und neustarten
```

login_attempts_threshold: 5 ## Loginversuche

Addon Configuration

Host Config:

```
- hostname: "router.example.com"
  service: "http://192.168.1.1"
- hostname: "diskstation.example.com"
  service: "https://192.168.1.2:5001"
- hostname: "website.example.com"
  service: "http://192.168.1.3:8080"
```

ACHTUNG! Du teilst diesen Dienst im Netz bitte beachte das du keinerlei Dienste teilst die kein Passwort benötigen - wie z.B. die 3D Druck Seite etc.

Meine Cloudflare Settings (für Alexa bitte prüfen)

Damit meine Alexa anständig funktioniert habe ich folgende Settings in Cloudflare geändert bzw. angepasst:

1. SSL
 1. Überblick
 1. Verschlüsselungsmodus = Vollständig
 2. SSL TLS Recommender = aktiv
2. Security
 1. Settings
 1. Security Level = medium
 2. Challenge Passage = 30 min
3. Speed
 1. Optimization
 1. Auto Minify = alles aktiv
 2. Early Hints = aktiv
 3. Rocket Loader = nicht aktiv
4. Caching
 1. Tiered Cache
 1. Argo Tiered Cache = aktiv
5. Network
 1. 0-RTT Connection Resumption = aktiv

<https://www.youtube.com/embed/RR7F3tR3ZzM>

Alexa ohne Cloud

hier erfährst du wie man Alexa ohne Nabu Casa Cloud in Homeassistant integrieren kannst

Voraussetzungen

Dein Homeassistant muss von aussen erreichbar sein - schau dir hierzu gern meinen Cloudflare Betrag an:



melde dich mit deinem Amazon Konto im Dev Portal anmelden



jetzt benötigst du eine Kreditkarte (keine Angst das hier kostet nichts im Monat nur einmalig 1€) für die Anmeldung bei AWS



Verwendet überall das Konto womit eure Alexas verbunden sind

Installation

Skill erstellen

Starte auf der Developer Seite und erstelle dort einen neuen Skill. Als Skillname kannst du verwenden was du möchtest. Bei Sprache wählst du bitte German.

Im nächsten Step wählst du bitte Smart Home

1. Choose a type of experience

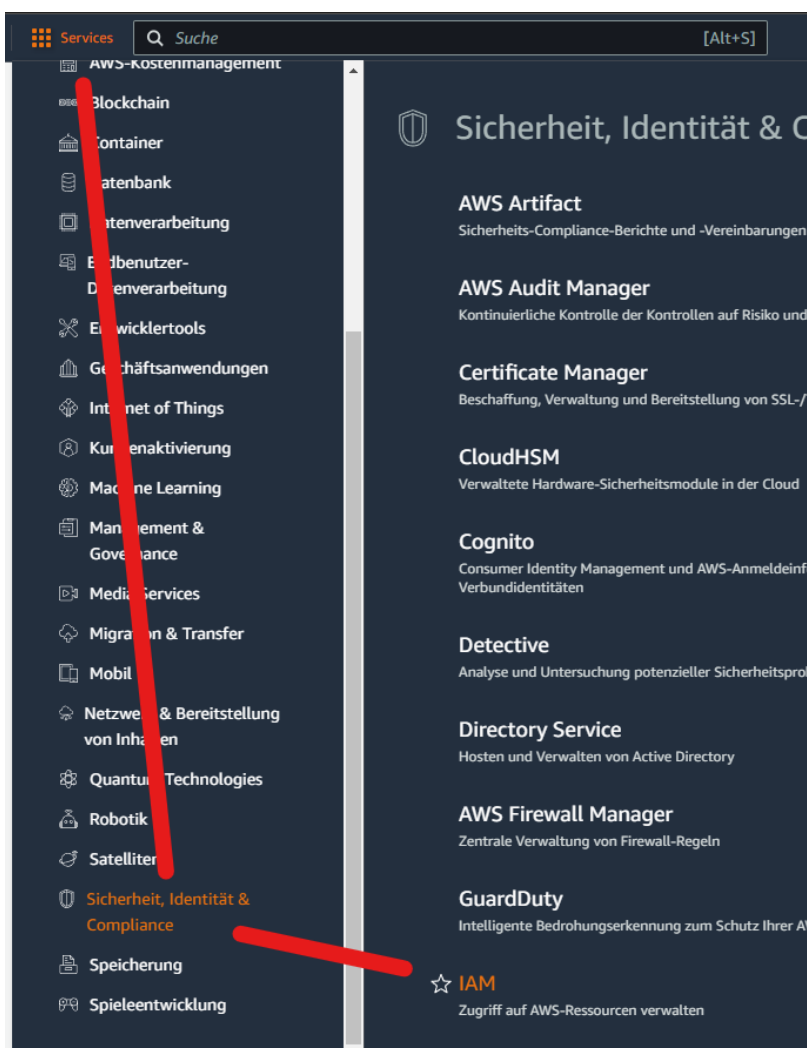
Tell us what kind of experience you want to build and we'll recommend a voice interaction model (also known as a skill model) to get you started. A skill model has pre-defined words and phrases that users can say when interacting with your skill. You'll be able to customize what Alexa responds with that is unique to your skill's use-case.

- ☐ Food & Drink ☐ Games & trivia ☐ Movies & TV ☐ Music & Audio ☐ News ☒ Smart home ☐ Other

und dann klickst du dich durch bis der Skill erstellt ist.

Lambda Rolle erstellen

Jetzt wechseln wir auf die AWS Seite und gehen dort auf Services - Sicherheit - IAM



Dort im Abschnitt Rollen

Q IAM suchen

Dashboard

▼ Zugriffsverwaltung

Benutzergruppen

Benutzer

Rollen

Richtlinien

erstellen wir eine neue Rolle.

Wählt hier AWS Service und Lambda.

☒ AWS-Service
EC2, Lambda oder andere, Aktionen in diesem Konto auszuführen.

☐ SAML-2.0-Verbund
Benutzer, die mit SAML 2.0 in einem Unternehmensverzeichnis verbunden sind, können Aktionen in diesem Konto ausführen.

Anwendungsfall

Erlauben Sie AWS-Services wie EC2, Lambda oder anderen, Aktionen

Häufige Anwendungsfälle

☐ EC2
Allows EC2 instances to call AWS services on your behalf.

☐ Lambda
Allows Lambda functions to call AWS services on your behalf.

Anwendungsfälle für andere AWS-Services:

Wählen Sie einen Service aus, um den Anwendungsfall a

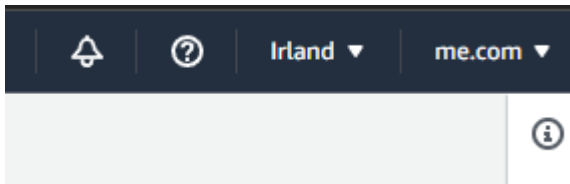
Jetzt sucht ihr euch:

AWSLambdaBasicExecutionRole

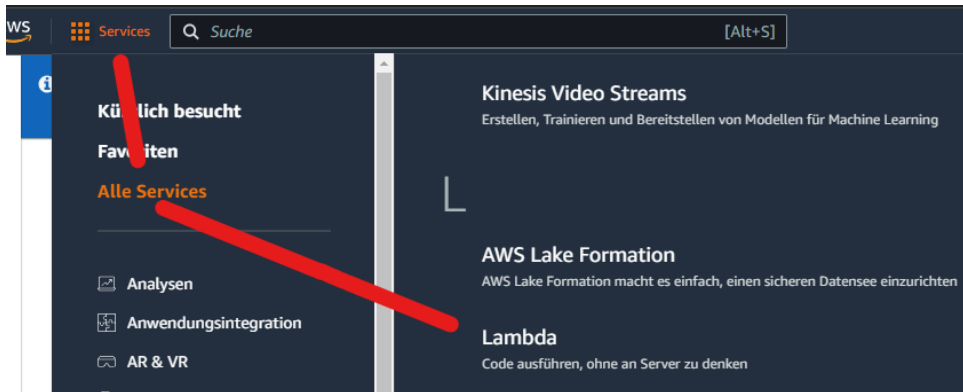
aus der Liste und setzt einen Hacken davor - scrollt nach unten und drückt auf weiter. Jetzt noch einen beliebigen Namen vergeben und Rolle erstellen.

Lambda konfigurieren und testen

Als erstes checkt bitte das eure AWS auf Irland gestellt ist



Jetzt müssen wir noch den Lambda Funktion erstellen. Dazu öffnet ihr bitte die Lambda über Services / Alle Services



Jetzt erstellt ihr eine Funktion mit folgenden Werten:

Name: wie ihr wollt

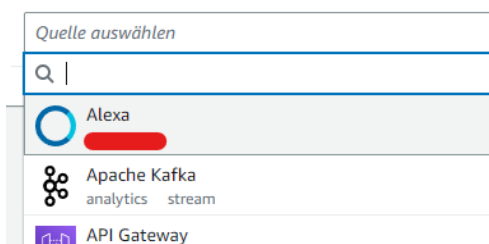
Laufzeit: Python 3.9

Standard Ausführungsrolle: hier wählt ihr eure zuvor erstellte Rolle aus

Jetzt noch

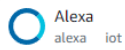
+ Auslöser hinzufügen

und dort



auswählen und konfigurieren:

Auslöser-Konfiguration [Info](#)



Choose an Alexa product

☐ Alexa Skills Kit

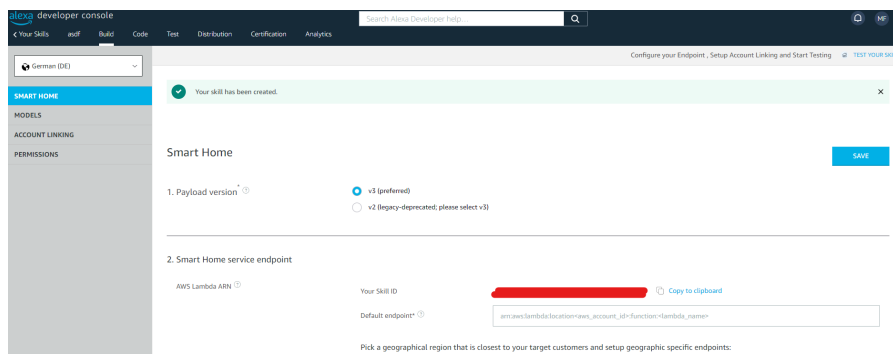
☒ Alexa Smart Home

Skill ID

FINDEST DU IN DEINER DEV CONSOLE

⚠ Ungültige Eingabe.

Lambda fügt die erforderlichen Berechtigungen für Amazon Alexa hinzu, um die Lambda-Funktion von diesem Auslöser aufzurufen. [Weitere Informationen](#) zum Lambda-Berechtigungsmodell.



Nachdem der Auslöser hinterlegt habt fügt ihr folgenden Code in die Lambda Function ein:

Lambda Function Code

====

Copyright 2019 Jason Hu <awaregit at gmail.com>

Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.

====

import os

```

import json
import logging
import urllib3

_debug = bool(os.environ.get('DEBUG'))

_logger = logging.getLogger('HomeAssistant-SmartHome')
_logger.setLevel(logging.DEBUG if _debug else logging.INFO)

def lambda_handler(event, context):
    """Handle incoming Alexa directive."""

    _logger.debug('Event: %s', event)

    base_url = os.environ.get('BASE_URL')
    assert base_url is not None, 'Please set BASE_URL environment variable'
    base_url = base_url.strip("/")

    directive = event.get('directive')
    assert directive is not None, 'Malformatted request - missing directive'
    assert directive.get('header', {}).get('payloadVersion') == '3', \
        'Only support payloadVersion == 3'

    scope = directive.get('endpoint', {}).get('scope')
    if scope is None:
        # token is in grantee for Linking directive
        scope = directive.get('payload', {}).get('grantee')
    if scope is None:
        # token is in payload for Discovery directive
        scope = directive.get('payload', {}).get('scope')
    assert scope is not None, 'Malformatted request - missing endpoint.scope'
    assert scope.get('type') == 'BearerToken', 'Only support BearerToken'

    token = scope.get('token')
    if token is None and _debug:
        token = os.environ.get('LONG_LIVED_ACCESS_TOKEN') # only for debug purpose

```

```

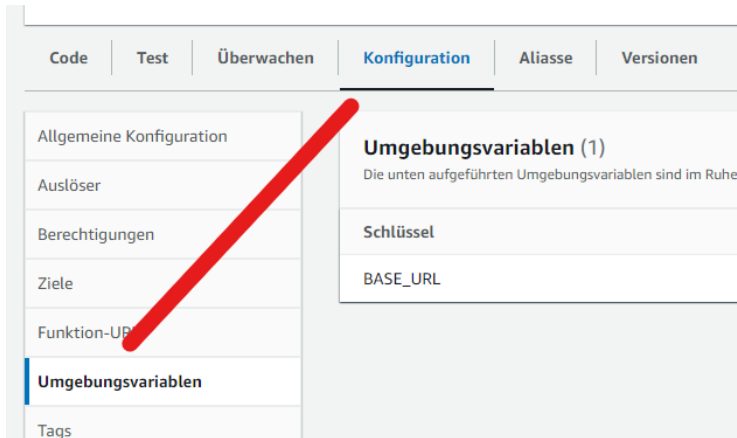
verify_ssl = not bool(os.environ.get('NOT_VERIFY_SSL'))

http = urllib3.PoolManager(
    cert_reqs='CERT_REQUIRED' if verify_ssl else 'CERT_NONE',
    timeout=urllib3.Timeout(connect=2.0, read=10.0)
)

response = http.request(
    'POST',
    '{}api/alexa/smart_home'.format(base_url),
    headers={
        'Authorization': 'Bearer {}'.format(token),
        'Content-Type': 'application/json',
    },
    body=json.dumps(event).encode('utf-8'),
)
if response.status >= 400:
    return {
        'event': {
            'payload': {
                'type': 'INVALID_AUTHORIZATION_CREDENTIAL'
                if response.status in (401, 403) else 'INTERNAL_ERROR',
                'message': response.data.decode("utf-8"),
            }
        }
    }
return json.loads(response.data.decode('utf-8'))

```

Danach auf Deploy und jetzt noch die Umgebungsvariablen bearbeiten:



klickt auf Bearbeiten und fügt folgenden Schlüssel:

BASE_URL

WICHTIG: falls ihr DuckDNS etc. verwendet! Die URL darf keinen Port enthalten!

hinzu und als Wert dahinter eure URL womit ihr von aussen auf euren Homeassistant connecten könnt.

Jetzt passt ihr eure `Configuration.yaml` noch an um die Lambda zu testen:

```
alexa:
  smart_home:
```

wenn du die `alexa.yaml` verwendest darfst du nicht vergessen diese mit `alexa:`
`!include alexa.yaml` einzubinden

für den Lambda Test nutzt ihr dann folgenden Code:

Test Code

```
{
  "directive": {
    "header": {
      "namespace": "Alexa.Discovery",
      "name": "Discover",
      "payloadVersion": "3",
```

```
"messageId": "1bd5d003-31b9-476f-ad03-71d471922820"
},
"payload": {
  "scope": {
    "type": "BearerToken"
  }
}
}
```

Als Ergebnis sollte folgender Fehler erscheinen:



Sollte hier ein 404 Fehler kommen - bitte die BASE_URL prüfen (<https://deineurl.de>)
hier muss https davor stehen und kein / dahinter.

Skill fertigstellen in Amazon Dev

Fügt jetzt euren Endpoint von AWS in Amazon Dev ein und prüft nochmals die Location im URL und speichert.

Danach klickt ihr auf:

Setup Account Linking

und füllt dies wie folgt aus:

- Authorization URI : `https://[YOUR HOME ASSISTANT URL]/auth/authorize`
- Access Token URI : `https://[YOUR HOME ASSISTANT URL]/auth/token`
- Client ID : `https://layla.amazon.com/`
- Client Secret könnt ihr eintragen was ihr wollt.
- Your Authentication Scheme: `Credentials in request body`
- Scope add -> `smart_home`

ACHTUNG! Aktiviert den Skill erst wenn ihr den nachfolgenden Schritt erledigt habt und eure Geräte in die Alexa Config eingetragen habt! Sonst habt ihr alle Geräte in Alexa

Sollte das Accountlinking fehlschlagen probier mal deine Domain von Homeassistant zu hinterlegen:

Your Authentication Scheme [?]

Credentials in request body

Scope [?]

smart_home

x

+ Add scope

Domain List [?]

+ Add domain

Default Access Token Expiration Time [?]

Alexa App und yaml

Jetzt müsst ihr nur noch euren Alexa Skill aktivieren und die yaml anpassen - hier mein Beispielcode

```
smart_home:
  locale: de-DE
  endpoint: https://api.eu.amazonalexa.com/v3/events
  filter:
    include_entities:
      - light.licht_speise
      - light.licht_waschkuche
      - light.ambiente_pc
```

Homeassistant neustarten nicht vergessen ;)

Link zum Video



Die Webansicht gibt es leider nicht mehr somit muss alles in der Handyapp gemacht werden.

<https://www.youtube.com/embed/JWICstV-C8E>

Bluetooth Repeater - Reichweite erhöhen

Hardware

bestell dir die unten aufgeführten Produkte und dann gehts weiter...

[NodeMCU](#)

[Netzteil](#)

ESP Code zur Gerätesuche

diesen Code nimmst du um alle Bluetooth Geräte in Reichweite zu finden.



Jetzt müsst ihr folgenden Code einfügen:

```
esp32_ble_tracker:  
bluetooth_proxy:
```

Als Ergebnis solltet ihr dann folgendes im Log finden:

```
[22:45:13][D][api:102]: Accepted ::FFFF:C0A8:B241  
[22:45:13][D][api.connection:861]: Home Assistant 2022.9.0 (::FFFF:C0A8:B241): Connected successfully
```

Sensoren integrieren

mit Home Assistant 2022.9 ist der Bluetooth Proxy integriert damit findest du die Sensoren automatisch

image not found or type unknown



die Sensoren werden automatisch gefunden.

Macht einen Sensor nach dem anderen sonst kommt man schnell durcheinander

image not found or type unknown



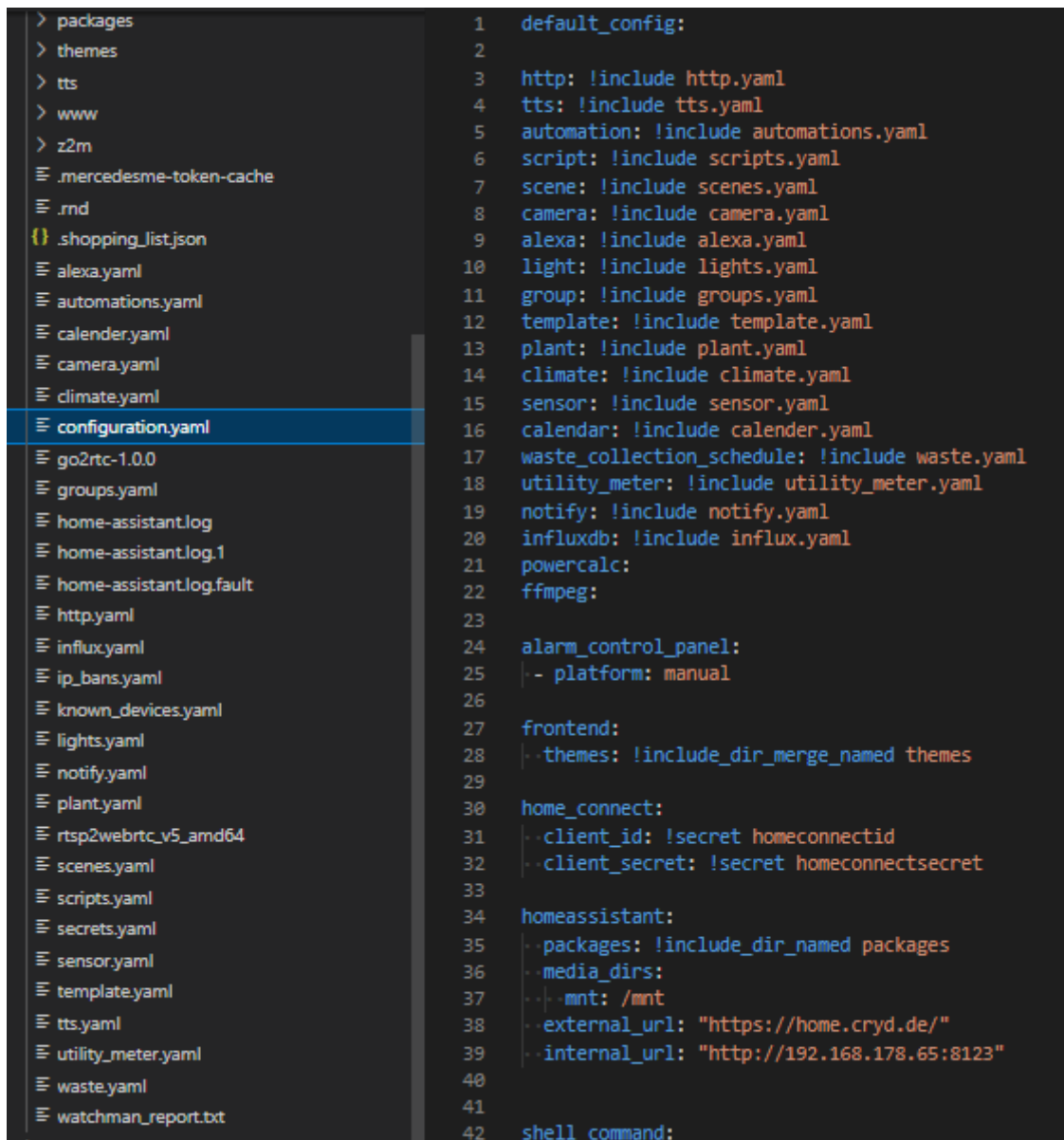
ABSENDEN drücken und der Sensor ist angelegt

Jetzt nur noch die Sensoren in der Plant.yaml anlegen

<https://www.youtube.com/embed/8Uld-cU0y6w>

Configuration.YAML strukturieren

Deine Configuration yaml wird bald aus allen Nähten platzen oder ist es bereits.



```
1 default_config:
2
3 http: !include http.yaml
4 tts: !include tts.yaml
5 automation: !include automations.yaml
6 script: !include scripts.yaml
7 scene: !include scenes.yaml
8 camera: !include camera.yaml
9 alexa: !include alexa.yaml
10 light: !include lights.yaml
11 group: !include groups.yaml
12 template: !include template.yaml
13 plant: !include plant.yaml
14 climate: !include climate.yaml
15 sensor: !include sensor.yaml
16 calendar: !include calender.yaml
17 waste_collection_schedule: !include waste.yaml
18 utility_meter: !include utility_meter.yaml
19 notify: !include notify.yaml
20 influxdb: !include influx.yaml
21 powercalc:
22 ffmpeg:
23
24 alarm_control_panel:
25   - platform: manual
26
27 frontend:
28   - themes: !include_dir_merge_named themes
29
30 home_connect:
31   - client_id: !secret homeconnectid
32   - client_secret: !secret homeconnectsecret
33
34 homeassistant:
35   - packages: !include_dir_named packages
36   - media_dirs:
37     - mnt: /mnt
38   - external_url: "https://home.cryd.de/"
39   - internal_url: "http://192.168.178.65:8123"
40
41
42 shell_command:
```

Du kannst dir für jeden Abschnitt eine eigene Datei anlegen und diesen Part dann komplett dort auslagern.

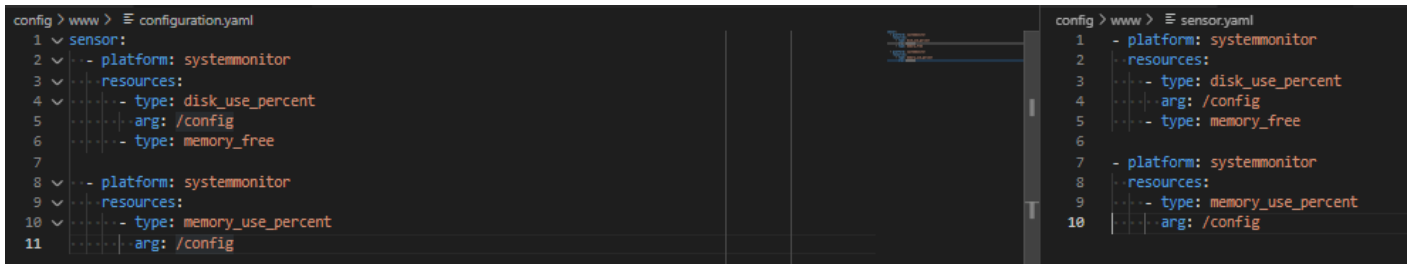
Hierzu musst du eine yaml Datei erstellen und in die Configuration.yaml inkludieren. Als Beispiel kannst du deine Sensoren auslagern in die sensor.yaml

Als erstes erstellst du eine Datei mit dem Namen sensor.yaml und anschließend fügst du noch:

```
sensor: !include sensor.yaml
```

in deine Configuration.yaml ein um sozusagen dort auf die andere Datei zu verweisen.

Jetzt kannst du in der anderen Datei deine Config Einträge für alles was unter Sensor fällt erstellen - beachte jedoch dort musst du sensor: nicht mehr hinterlegen wie in folgendem Bild hinterlegt (links = alles in Configuration.yaml / rechts = ausgelagert in sensor.yaml):



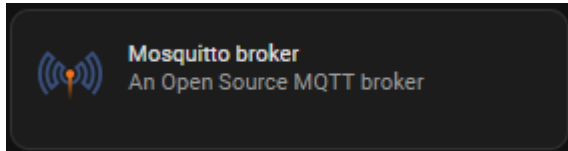
```
config > www > configuration.yaml
1 sensor:
2   - platform: systemmonitor
3     resources:
4       - type: disk_use_percent
5         arg: /config
6       - type: memory_free
7
8   - platform: systemmonitor
9     resources:
10      - type: memory_use_percent
11        arg: /config

config > www > sensor.yaml
1 - platform: systemmonitor
2   resources:
3     - type: disk_use_percent
4       arg: /config
5     - type: memory_free
6
7 - platform: systemmonitor
8   resources:
9     - type: memory_use_percent
10      arg: /config
```

MQTT Broker

Installation

Um den Broker zu installieren einfach im Addon Store folgendes Addon downloaden

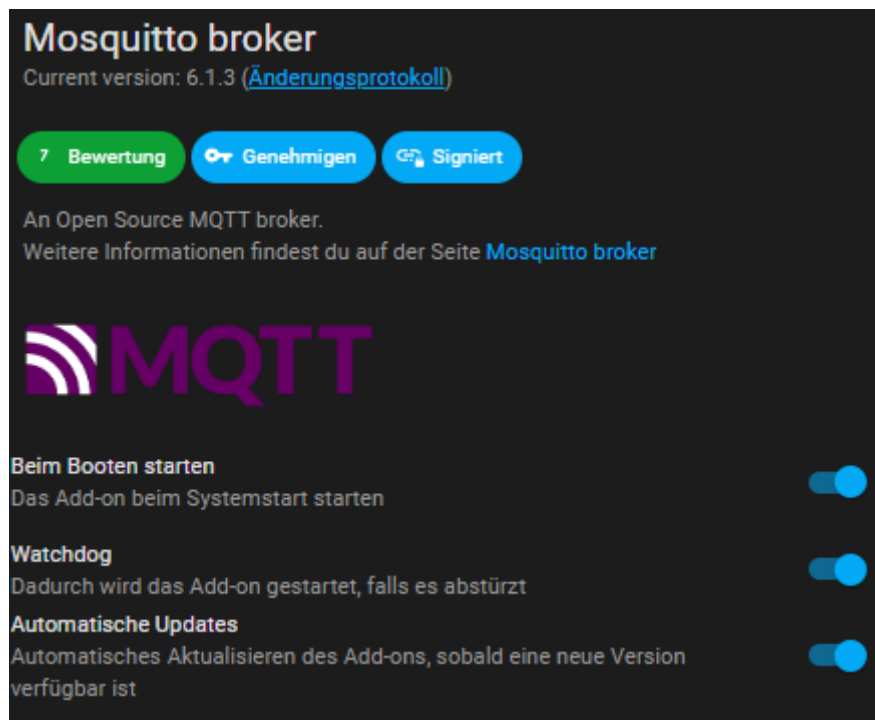


Config

Bei Logins müsst ihr einen Login anlegen z.B.:

- username: mqtt_user
password: 1234asdf

Start Parameter



Zigbee2mqtt

<https://www.youtube.com/embed/ObrCkJ9SQwk>

Welche Geräte unterstützt Zigbee2mqtt im Gegensatz zu den anderen Zigbee Integrationen und Addons?

 **Vergleich**

Welche Vorteile habe ich noch durch Z2M? (hier spielen meine persönlichen Erfahrungen mit rein)

- schneller
- stabiler
- mehr implementierte Geräte
- Community Addon - somit kommen schnell neue Geräte dazu
- sehr angenehmes VerwaltungsUI
- OTA Updates der Geräte

benötigte Hardware

Hier ein paar Sticks welche von Z2M unterstützt werden - ich empfehle den Sonoff Stick.

 **Sonoff**

oder

 **Conbee 2**

(beachte extra Config weiter unten)

Vorbereitung

Für Z2M benötigt ihr folgende Addons:

- Mqtt Broker
- Zigbee2mqtt

<https://github.com/zigbee2mqtt/hassio-zigbee2mqtt>

Der Mqtt Broker sollte bei euch schon eingerichtet sein falls nicht schaut gern hier mal vorbei:

 **MQTT Broker einrichten**

Einrichtung Zigbee2mqtt

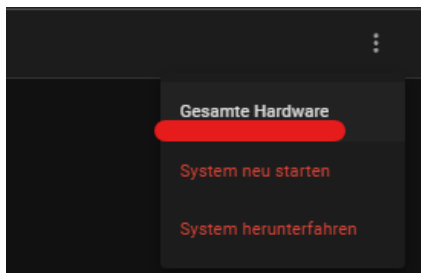
- lasst den datapath und socat wie er ist
- tragt eure Mqtt daten wie folgt ein:

server: mqtt://IPADRESSE-MQTT-BROKER:1883

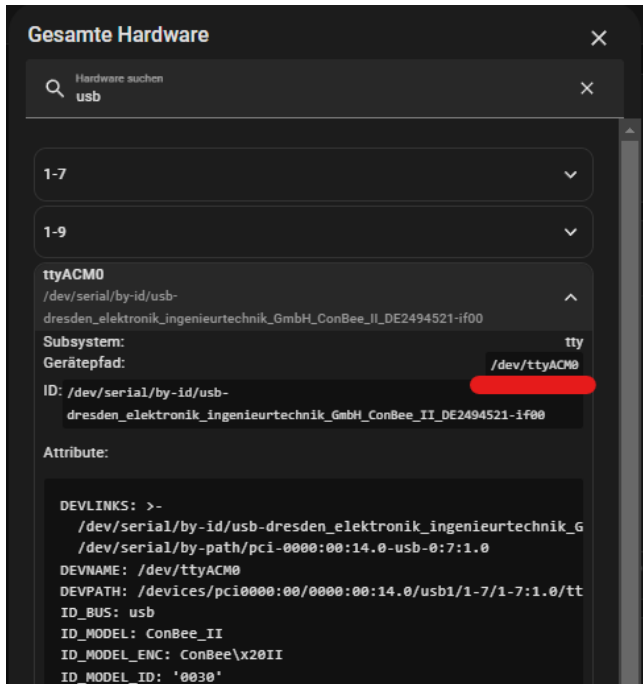
user: mqtt_user

password: 1234asdf

jetzt müssen wir noch den USB Port vom eurem Stick ausfindig machen. Hierzu gehst du einfach im Homeassistant auf Einstellungen/System/Hardware und klickst oben rechts auf die drei Punkte



suche nach



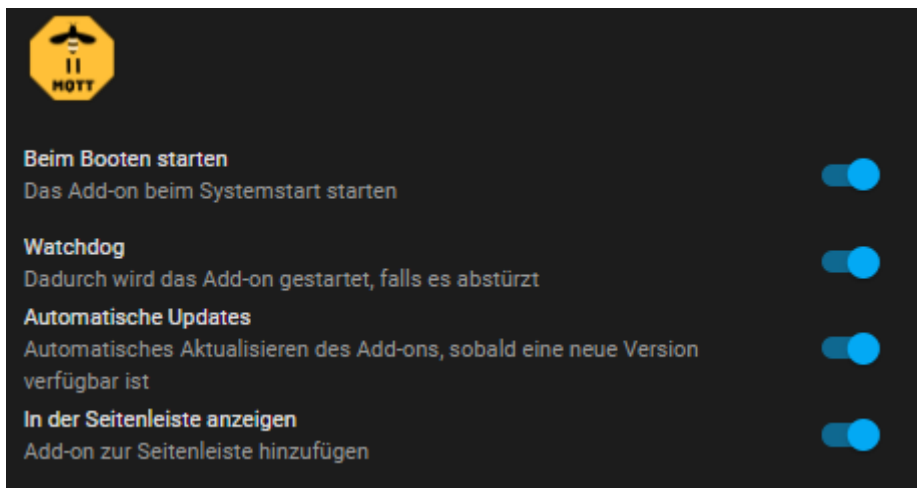
diesen Pfad notierst du dir und fügst ihn im Z2M Addon wie folgt ein:

port: /dev/ttyACM0

adapter: deconz #diese Zeile nur falls ihr wie ich einen Conbee2 benutzt

adapter: ezsp #bei manchen Sonoff Modellen ist diese Zeile noch notwendig

Jetzt aktiviert bitte noch folgende Optionen:



Automatische Updates ist optional

Jetzt startest du dein Addon und wartest kurz....

falls ihr ZHA installiert habt muss das deaktiviert werden in den Integrationen

Die ersten Schritte

Wie du welche Geräte anlernen kannst und was diese Geräte können findest du unter folgender Seite



Tipps beim anlernen:

- vergib eindeutige Namen z.B. Bewegungsmelder - Bad / Steckdose - Büro / Fensterkontakt - Büro
- aktiviere immer die Funktion Homeassistant Namen übertragen
- lege ggf. Gruppen an z.B. 3 LED in einer Lampe
- fange mit den Repeater an (alle Geräte welche direkt mit Strom versorgt sind)

Homeassistant App

Features:

Hier findest du alles Rund um die Homeassistant APP:

- Kritische Mitteilungen (Benachrichtigung)
- Mitteilung / Notifications mit Foto
- Mitteilung / Notifications mit Foto und Action

Kritische Mitteilungen

Warum sollte ich kritische Mitteilungen verwenden?

Mit diesen Mitteilungen kannst du dich auch wenn dein Handy auf Lautlos / DND Modus ist benachrichtigen lassen von wichtigen Ereignissen wie z.B. Wassersensor schlägt aus, Feuermelder reagiert oder unbefugter Zutritt.

Um kritische Mitteilungen per Homeassistant zu versenden kannst du folgende Codes verwenden:

iOS

Komplette Action:

```
action:
- service: notify.mobile_app_<your_device_id_here>
  data:
    title: "Wake up!"
    message: "The house is on fire and the cat's stuck in the dryer!"
  data:
    push:
      sound:
        name: "default"
        critical: 1
```

volume: 1.0

Data Field:

push:
 sound:
 name: default
 critical: 1
 volume: 1

Android

Komplette Action:

action:
 - service: notify.mobile_app_<your_device_id_here>
 data:
 title: "Wake up!"
 message: "The house is on fire and the cat's stuck in the dryer!"
 data:
 ttl: 0
 priority: high

Data Field:

ttl: 0
priority: high
channel: alarm_stream

TTS Benachrichtigung:

ttl: 0
priority: high
media_stream: alarm_stream
tts_text: "HIER DEIN GEWÜNSCHTER TEXT"

Max Volume:

```
ttl: 0
priority: high
media_stream: alarm_stream_max
tts_text: "HIER DEIN GEWÜNSCHTER TEXT"
```

Notification mit Foto

In der Notification kannst du folgende Aktion benutzen:

```
- service: camera.snapshot
  data:
    filename: /media/local/haustuer.jpg
  target:
    entity_id: camera.camera4
- service: notify.all_devices
  data:
    title: Haustür
    message: jemand hat geklingelt
    data:
      image: /media/local/local/haustuer.jpg
```

Du musst hier natürlich deine Entitys entsprechend anpassen.

Notification mit Foto und Action

In der Notification kannst du folgende Aktion benutzen:

```
- service: notify.all_devices
  data:
    title: Haustür
    message: jemand hat geklingelt
    data:
      image: /media/local/local/haustuer.jpg
    actions:
      - action: URI
```

title: Livestream

uri: /smartphone-1/doorbell