Heizung Steuerung - Thermostate

Überlegung und der Plan

Anforderung

- Jeder Raum soll einen eigenen Zeitplan für die Absenkung erhalten
- Die Heizkörper sollen beim Öffnen der Fenster ausschalten
- Die Heizkörper Thermostate sollen sich per Switch An- und Ausschalten lassen
- Der Zeitplan der Absenkung soll sich per Switch An- und Ausschalten lassen
- Per Pushbutton soll sich die aktuelll richtige Temperatur einstellen lassen
- Urlaubsmodus

Umsetzung

Damit das am Ende mit den wenigen Automatisierungen funktioniert, bedarf es einer Namenskonvention für die einzelnen Entititäten.

Alles dreht sich immer um den Raum. Dieser wird in allen Entitäten gleich sein und von "_" eingeschlossen sein. Daraus folgernd können dann daraus alle nötigen Entitäten dynamisch erstellt werden.

In "Heizung An- / Aus" habe ich auch die Details der Templates beschrieben.

Konvention

- Thermostate: climate.thermostat_<RAUM>
- Helper: Komfort Temperatur: input_number.thermostat_<RAUM>_comfort
- Helper: Eco Temperatur: input_number.thermostat_<RAUM>_eco
- Helper: Zeitplan: schedule.heizung_<RAUM>_eco
- Helper: Zeitplan Umschalter: input_boolean.heizung_<RAUM>_zeitplan
- Helper: An / Aus: input_boolean.heizung_<RAUM>_on_off
- Fenster Sensoren oder Helper: binary_sensor.fenster_<RAUM>_open_close

Allgemeine Helper

- Urlaubs Switch: input_boolean.urlaub

Gruppen

Ich habe Räume mit 2 Thermostaten oder mehreren Fenster Sensoren. Hierfür müssen Gruppen angelegt werden.

In HACS gibt es die Integration "climate_group" hierfür:

https://github.com/daenny/climate_group

In die Fenster Helper Entitäten werden alle dem Raum zugehörigen Sensoren gepackt.

Thermostat Danfoss Ally

Ich benutze als Regler das Danfoss Ally Radiator Thermostat mit ZHA

Dieses Thermostat kenn keinen HVAC Modus OFF, obwohl es in den Lovelace Objekten angezeigt wird. Auch in den Entwicklertools ist der Modus OFF sichtbar.

Da dieser aber nicht ansprechbar ist, wird für OFF die Temperatur 5 Grad gesetzt.

Thermostat Shelly TRV

In Räumen, in denen ein externer Temperatur Sensor nötig wird, habe ich nun die Shelly TRVs eingesetzt. (Büro, Wohnzimmer)

Konfiguration ShellyTRV

Automatisierung

Zur Steuerung der Heizkörper Thermostate habe ich einige wenige Automationen erstellt.

- Heizung Absenkung Zeit
- Heizung An / Aus
- Heizung Fenster Absenkung
- Heizung Set Default Temperature
- Heizung Switch Zeitplan

Heizung Absenkung Zeit

Wenn der Status eines Zeitplans wechselt soll die dazugehörige Temperatur gesetzt werden.

Beim Wechsel von Komfort nach Absenkung wird die aktuelle Temperatur in die Komfort Temperatur Entität geschrieben.

Alle nötigen Stati und Entitäten werden im Variablen Block gesetzt, damit bleibt die Sequence schön kurz.

> Wichtig ist der Modus "parallel" und max sollte größer sein als die Anzahl der Räume! Ansonsten ist die Gefahr groß, dass Automatisierungsläufe nicht ausgeführt werden.

Code

```
alias: Heizung Absenkung Zeit
description: ""
trigger:
  - platform: state
    entity_id:
      - schedule.heizung_wc_eco
      - schedule.heizung_eingang_eco
      - schedule.heizung_flur_eco
      - schedule.heizung_buero_eco
      - schedule.heizung_werkstatt_eco
      - schedule.heizung_ez_eco
      - schedule.heizung_wz_eco
      - schedule.heizung_sz_eco
      - schedule.heizung_gruen_eco
      - schedule.heizung_rosa_eco
      - schedule.heizung_lydi_eco
      - schedule.heizung_kinderbad_eco
      - schedule.heizung_carrera_eco
      - schedule.heizung_wcdg_eco
      - schedule.heizung_studio_eco
condition:
  - condition: state
    entity_id: input_boolean.urlaub
```

```
    state: "off"
action:
  - variables:
      room: "{{ trigger.entity_id.split('_')[1] }}"
      device: "{{ 'climate.thermostat_' + room }}"
      comfort: "{{ 'input_number.thermostat_' + room + '_comfort' }}"
      eco: "{{ 'input_number.thermostat_' + room + '_eco' }}"
      device_temp: "{{ state_attr( device, 'temperature') }}"
      eco_temp: "{{ states(eco) }}"
      comfort_temp: "{{ states(comfort) }}"
      scheduler: "{{ 'schedule.heizung_' + room + '_eco' }}"
      scheduler_state: "{{ states(scheduler) | bool }}"
      scheduler_switch: "{{ states('input_boolean.heizung_' + room + '_zeitplan') | bool }}"
      climate_state: "{{ states('input_boolean.heizung_' + room + '_on_off') | bool }}"
      window_state: "{{ states('binary_sensor.fenster_' + room + '_open_close') }}"
      store_temp: "{{ device_temp > eco_temp }}"
      temp: |
        {% if scheduler_state %}
          {{ eco_temp }}
        {% else %}
          {{ comfort_temp }}
        {% endif %}
  - if:
      - condition: template
        value_template: "{{ scheduler_switch }}"
      - condition: template
        value_template: "{{ climate_state }}"
    then:
      - if:
          - condition: template
            value_template: "{{ store_temp }}"
          - condition: template
            value_template: "{{ window_state }} == 'off'"
          - condition: template
            value_template: "{{ not scheduler_state }}"
        then:
          - service: input_number.set_value
            data_template:
              value: "{{ device_temp }}"
            target:
```

```
            entity_id: "{{ comfort }}"
      - service: script.turn_on
        data:
          variables:
            room: "{{ room }}"
            temp: "{{ temp }}"
        target:
          entity_id: script.heizung_set_temperature
mode: parallel
max: 20
```

Heizung Absenkung Fenster

Wenn der Status eines Fenster Sensors wechselt, wird das Thermostat An- oder ausgeschaltet.

Im Komfort Zeitfenster wird hierzu die aktuelle Temperatur gesichert.

Alle nötigen Stati und Entitäten werden im Variablen Block gesetzt, damit bleibt die Sequence schön kurz.

> Wichtig ist der Modus "parallel" und max sollte größer sein als die Anzahl der Räume! Ansonsten ist die Gefahr groß, dass Automatisierungsläufe nicht ausgeführt werden.

Als Optimierung könnte man noch einen Zeit Trigger einbauen, so dass die Aktion erst nach 1 oder 2 Minuten ausgeführt wird. Die wird auf jeden Fall der Batterie Laufzeit des Thermostates zu Gute kommen.

Code:

```
alias: Heizung Fenster Absenkung
description: ""
trigger:
  - platform: state
    entity_id:
      - binary_sensor.fenster_wc_open_close
      - binary_sensor.fenster_buero_open_close
      - binary_sensor.fenster_werkstatt_open_close
      - binary_sensor.schiebetuere_wz_open_close
      - binary_sensor.fenster_sz_open_close
      - binary_sensor.fenster_gruen_open_close
      - binary_sensor.fenster_rosa_open_close
      - binary_sensor.fenster_lydi_open_close
      - binary_sensor.fenster_carrera_open_close
      - binary_sensor.fenster_wcdg_open_close
      - binary_sensor.fenster_studio_open_close
    id: FensterAUF
    to: "on"
  - platform: state
    entity_id:
```

```
      - binary_sensor.fenster_wc_open_close
      - binary_sensor.fenster_buero_open_close
      - binary_sensor.fenster_werkstatt_open_close
      - binary_sensor.schiebetuere_wz_open_close
      - binary_sensor.fenster_sz_open_close
      - binary_sensor.fenster_gruen_open_close
      - binary_sensor.fenster_rosa_open_close
      - binary_sensor.fenster_lydi_open_close
      - binary_sensor.fenster_carrera_open_close
      - binary_sensor.fenster_wcdg_open_close
      - binary_sensor.fenster_studio_open_close
    id: FensterGESCHLOSSEN
    to: "off"
condition:
  - condition: state
    entity_id: input_boolean.urlaub
    state: "off"
action:
  - variables:
      room: "{{ trigger.entity_id.split('_')[2] }}"
      device: "{{ 'climate.thermostat_' + room }}"
      climate_state: "{{ states('input_boolean.heizung_' + room + '_on_off') | bool }}"
      comfort: "{{ 'input_number.thermostat_' + room + '_comfort' }}"
      eco: "{{ 'input_number.thermostat_' + room + '_eco' }}"
      device_temp: "{{ state_attr( device, 'temperature') }}"
      eco_temp: "{{ states(eco) }}"
      comfort_temp: "{{ states(comfort) }}"
      scheduler: "{{ 'schedule.heizung_' + room + '_eco' }}"
      scheduler_state: "{{ states(scheduler) | bool }}"
      scheduler_switch: "{{ states('input_boolean.heizung_' + room + '_zeitplan') | bool }}"
      store_temp: "{{ device_temp > eco_temp }}"
  - choose:
      - conditions:
          - condition: trigger
            id: FensterAUF
        sequence:
          - if:
              - condition: template
                value_template: "{{ scheduler_state }}"
            then:
```

```
        - service: script.turn_on
          data:
            variables:
              room: "{{ room }}"
              temp: "5"
          target:
            entity_id: script.heizung_set_temperature
      else:
        - if:
            - condition: template
              value_template: "{{ store_temp }}"
          then:
            - service: input_number.set_value
              data_template:
                value: "{{ device_temp }}"
              target:
                entity_id: "{{ comfort }}"
        - service: script.turn_on
          data:
            variables:
              room: "{{ room }}"
              temp: "5"
          target:
            entity_id: script.heizung_set_temperature
  - conditions:
      - condition: trigger
        id: FensterGESCHLOSSEN
      - condition: template
        value_template: "{{ climate_state }}"
    sequence:
      - if:
          - condition: template
            value_template: "{{ scheduler_state }}"
        then:
          - service: script.turn_on
            data:
              variables:
                room: "{{ room }}"
                temp: "{{ eco_temp }}"
            target:
```

```
            entity_id: script.heizung_set_temperature
        else:
          - service: script.turn_on
            data:
              variables:
                room: "{{ room }}"
                temp: "{{ comfort_temp }}"
            target:
              entity_id: script.heizung_set_temperature
max: 10
mode: parallel
```

Heizung An- / Aus

Wenn der Status eines Switches wechselt wird das Thermostat an- oder ausgeschaltet.

Beim Wechsel nach Aus innerhalb des Komfort Zeitplanes wird die aktuelle Temperatur in die Komfort Temperatur Entität geschrieben.

Alle nötigen Stati und Entitäten werden im Variablen Block gesetzt, damit bleibt die Sequence schön kurz.

2023.02.19: Überprüfung ob Fenster geschlossen sind bevor die Komfort Temperatur gespeichert wird

Code:

```
alias: Heizung An / Aus
description: ""
trigger:
  - platform: state
    entity_id:
      - input_boolean.heizung_buero_on_off
      - input_boolean.heizung_wc_on_off
      - input_boolean.heizung_eingang_on_off
      - input_boolean.heizung_flur_on_off
      - input_boolean.heizung_werkstatt_on_off
      - input_boolean.heizung_ez_on_off
      - input_boolean.heizung_wz_on_off
      - input_boolean.heizung_sz_on_off
      - input_boolean.heizung_gruen_on_off
      - input_boolean.heizung_rosa_on_off
      - input_boolean.heizung_lydi_on_off
      - input_boolean.heizung_carrera_on_off
      - input_boolean.heizung_wcdg_on_off
      - input_boolean.heizung_studio_on_off
    id: HeizungAN
    to: "on"
  - platform: state
    entity_id:
      - input_boolean.heizung_buero_on_off
```

```
      - input_boolean.heizung_wc_on_off
      - input_boolean.heizung_eingang_on_off
      - input_boolean.heizung_flur_on_off
      - input_boolean.heizung_buero_on_off
      - input_boolean.heizung_werkstatt_on_off
      - input_boolean.heizung_ez_on_off
      - input_boolean.heizung_wz_on_off
      - input_boolean.heizung_sz_on_off
      - input_boolean.heizung_gruen_on_off
      - input_boolean.heizung_rosa_on_off
      - input_boolean.heizung_lydi_on_off
      - input_boolean.heizung_carrera_on_off
      - input_boolean.heizung_wcdg_on_off
      - input_boolean.heizung_studio_on_off
    id: HeizungAUS
    to: "off"
condition:
  - condition: state
    entity_id: input_boolean.urlaub
    state: "off"
action:
  - variables:
      room: "{{ trigger.entity_id.split('_')[2] }}"
      device: "{{ 'climate.thermostat_' + room }}"
      comfort: "{{ 'input_number.thermostat_' + room + '_comfort' }}"
      eco: "{{ 'input_number.thermostat_' + room + '_eco' }}"
      device_temp: "{{ state_attr( device, 'temperature') }}"
      eco_temp: "{{ states(eco) }}"
      comfort_temp: "{{ states(comfort) }}"
      scheduler: "{{ 'schedule.heizung_' + room + '_eco' }}"
      scheduler_state: "{{ states(scheduler) | bool }}"
      scheduler_switch: "{{ states('input_boolean.heizung_' + room + '_zeitplan') | bool }}"
      window_state: "{{ states('binary_sensor.fenster_' + room + '_open_close') }}"
      store_temp: "{{ device_temp > eco_temp }}"
  - choose:
      - conditions:
          - condition: trigger
            id: HeizungAUS
        sequence:
          - if:
```

```
      - condition: template
        value_template: "{{ scheduler_state }}"
    then:
      - service: climate.set_temperature
        data:
          temperature: "5"
        target:
          entity_id: "{{ device }}"
    else:
      - if:
          - condition: template
            value_template: "{{ scheduler_state }}"
          - condition: template
            value_template: "{{ window_state }} == 'off'"
        then:
          - service: input_number.set_value
            data_template:
              value: "{{ device_temp }}"
            target:
              entity_id: "{{ comfort }}"
        - service: climate.set_temperature
          data:
            temperature: "5"
          target:
            entity_id: "{{ device }}"
- conditions:
    - condition: trigger
      id: HeizungAN
  sequence:
    - if:
        - condition: template
          value_template: "{{ scheduler_state }}"
      then:
        - service: climate.set_temperature
          data:
            temperature: "{{ eco_temp }}"
          target:
            entity_id: "{{ device }}"
      else:
        - service: climate.set_temperature
```

```
        data:
          temperature: "{{ comfort_temp }}"
        target:
          entity_id: "{{ device }}"
mode: parallel
max: 20
```

Heizung set Default Temperatur

Per Push Button lässt sich die Temperatur aus den Speicher Entitäten an das Thermostat senden.

Code:

```
alias: Heizung Set Default Temperature
description: ""
trigger:
  - platform: state
    entity_id:
      - input_button.set_zeitplan_wc
      - input_button.set_zeitplan_eingang
      - input_button.set_zeitplan_flur
      - input_button.set_zeitplan_buero
      - input_button.set_zeitplan_werkstatt
      - input_button.set_zeitplan_ez
      - input_button.set_zeitplan_wz
      - input_button.set_zeitplan_sz
      - input_button.set_zeitplan_gruen
      - input_button.set_zeitplan_rosa
      - input_button.set_zeitplan_lydi
      - input_button.set_zeitplan_kinderbad
      - input_button.set_zeitplan_carrera
      - input_button.set_zeitplan_wcdg
      - input_button.set_zeitplan_studio
condition:
  - condition: state
    entity_id: input_boolean.urlaub
    state: "off"
action:
  - variables:
      room: "{{ trigger.entity_id.split('_')[3] }}"
      device: "{{ 'climate.thermostat_' + room }}"
```

```
    climate_state: "{{ states('input_boolean.heizung_' + room + '_on_off') | bool }}"
    comfort: "{{ 'input_number.thermostat_' + room + '_comfort' }}"
    eco: "{{ 'input_number.thermostat_' + room + '_eco' }}"
    device_temp: "{{ state_attr( device, 'temperature') }}"
    eco_temp: "{{ states(eco) }}"
    comfort_temp: "{{ states(comfort) }}"
    scheduler: "{{ 'schedule.heizung_' + room + '_eco' }}"
    scheduler_state: "{{ states(scheduler) | bool }}"
    scheduler_switch: "{{ states('input_boolean.heizung_' + room + '_zeitplan') | bool }}"
    temp: |
      {% if scheduler_state %}
       {{ eco_temp }}
      {% else %}
       {{ comfort_temp }}
      {% endif %}
  - if:
    - condition: template
      value_template: "{{ climate_state }}"
  then:
    - service: script.turn_on
      data:
        variables:
          room: "{{ room }}"
          temp: "{{ temp }}"
      target:
        entity_id: script.heizung_set_temperature
  else:
    - service: climate.set_temperature
      data:
        temperature: 5
      target:
        device_id: "{{ device }}"
mode: parallel
max: 20
```

Heizung Switch Zeitplan

Per Switch kann der Zeitplan aktiviert oder deaktiviert werden. Dabei wird die dann korrekte Temperatur gesendet.

Zeitplan AUS bedeutet Komfort Temperatur

2023.02.19: Überprüfung ob Fenster geschlossen sind bevor die Komfort Temperatur gespeichert wird

Code:

```
alias: Heizung Switch Zeitplan
description: ""
trigger:
  - platform: state
    entity_id:
      - input_boolean.heizung_wc_zeitplan
      - input_boolean.heizung_eingang_zeitplan
      - input_boolean.heizung_flur_zeitplan
      - input_boolean.heizung_buero_zeitplan
      - input_boolean.heizung_werkstatt_zeitplan
      - input_boolean.heizung_ez_zeitplan
      - input_boolean.heizung_wz_zeitplan
      - input_boolean.heizung_sz_zeitplan
      - input_boolean.heizung_gruen_zeitplan
      - input_boolean.heizung_rosa_zeitplan
      - input_boolean.heizung_lydi_zeitplan
      - input_boolean.heizung_kinderbad_zeitplan
      - input_boolean.heizung_carrera_zeitplan
      - input_boolean.heizung_wcdg_zeitplan
      - input_boolean.heizung_studio_zeitplan
    id: ZeitplanOFF
    to: "off"
  - platform: state
    entity_id:
      - input_boolean.heizung_wc_zeitplan
      - input_boolean.heizung_eingang_zeitplan
```

```
      - input_boolean.heizung_flur_zeitplan
      - input_boolean.heizung_buero_zeitplan
      - input_boolean.heizung_werkstatt_zeitplan
      - input_boolean.heizung_ez_zeitplan
      - input_boolean.heizung_wz_zeitplan
      - input_boolean.heizung_sz_zeitplan
      - input_boolean.heizung_gruen_zeitplan
      - input_boolean.heizung_rosa_zeitplan
      - input_boolean.heizung_lydi_zeitplan
      - input_boolean.heizung_kinderbad_zeitplan
      - input_boolean.heizung_carrera_zeitplan
      - input_boolean.heizung_wcdg_zeitplan
      - input_boolean.heizung_studio_zeitplan
    id: ZeitplanON
    to: "on"
condition:
  - condition: state
    entity_id: input_boolean.urlaub
    state: "off"
action:
  - variables:
      room: "{{ trigger.entity_id.split('_')[2] }}"
      device: "{{ 'climate.thermostat_' + room }}"
      comfort: "{{ 'input_number.thermostat_' + room + '_comfort' }}"
      eco: "{{ 'input_number.thermostat_' + room + '_eco' }}"
      device_temp: "{{ state_attr( device, 'temperature') }}"
      eco_temp: "{{ states(eco) }}"
      comfort_temp: "{{ states(comfort) }}"
      scheduler: "{{ 'schedule.heizung_' + room + '_eco' }}"
      scheduler_state: "{{ states(scheduler) | bool }}"
      scheduler_switch: "{{ states('input_boolean.heizung_' + room + '_zeitplan') | bool }}"
      window_state: "{{ states('binary_sensor.fenster_' + room + '_open_close') }}"
      store_temp: "{{ device_temp > eco_temp | bool }}"
      temp: |
        {% if scheduler_state %}
          {{ eco_temp }}
        {% else %}
          {{ comfort_temp }}
        {% endif %}
  - choose:
```

```
    - conditions:
        - condition: trigger
          id: ZeitplanOFF
      sequence:
        - if:
            - condition: template
              value_template: "{{ not scheduler_state }}"
          then:
            - if:
                - condition: template
                  value_template: "{{ scheduler_state }}"
                - condition: template
                  value_template: "{{ window_state }} == 'off'"
              then:
                - service: input_number.set_value
                  data_template:
                    value: "{{ device_temp }}"
                  target:
                    entity_id: "{{ comfort }}"
    - conditions:
        - condition: trigger
          id: ZeitplanON
      sequence:
        - service: script.turn_on
          data:
            variables:
              room: "{{ room }}"
              temp: "{{ temp }}"
          target:
            entity_id: script.heizung_set_temperature
mode: parallel
max: 20
```

Heizung Urlaubs Modus

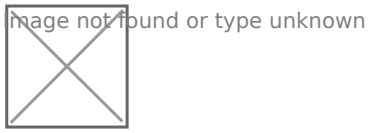
folgt...

Lovelace Ansichten

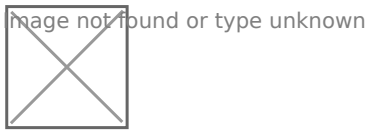
Ich habe die verschiedene Ansichten für die Heizung.

Es ist ein Mischung aus Mushroom Cards, Vertical- und horizontal Stack.

Die Detail Zeilen sind "Multiple Entity Row" aus HACS mit Tap Action um die Detail View des Raumes anzuzeigen.



Über den Raum erhalte ich diese Anzeige



Shelly TRV

Q1 2024: ich habe alle Shelly TRVs verkauft. Das war mit zu instabil mit der WLAN Kommunikation, obwohl ich gerade dieses auf ein komplett neues WIFI mit 9 Als im Haus upgedatet hatte.

Die ShellyTRV lassen es zu, als Temperatur Fühler einen externen Sensor zu nutzen und dessen Temperatur zu setzen.

Da es mir hier zu lästig ist nur auf Shelly Geräte zu setzen, nutze ich weiter meine bestehenden Sensoren und habe ein Skript und eine Automation, die die Temperatur setzen.

Im übrigen konfiguriere ich alle meine Shelly Geräte mit einer statischen IP. Das hat sich über die letzten Jahre als am stabilsten erwiesen. Damit kommt es nicht zum Verbindungsverlust, wenn das WIFI einmal "wackelt". Die zusätzliche Protokoll Last des DHCP wird damit unterbunden.

Shell Command in der configuration.yaml

```
shell_command:
  trv_set_ext_temp: "curl http://{{ trv_ip }}/ext_t?temp={{ ext_temp }}"
```

Die IP Adresse des TRV vergebe ich statisch und Speicher sie in einer input_number Entität.

Regex zur Überprüfung auf eine korrekte IP Adresse

```
[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}
```

TRV_BUUERO_IP.png
Image not found or type unknown

Im TRV muss der externe Sensor aktiviert sein

Shelly TRV.png
Image not found or type unknown

Automation

```
alias: Set TRV Büro Ext Temperature
description: ""
trigger:
  - platform: state
    entity_id:
      - sensor.mittelwert_buero_temperature
```

```
condition: []
action:
  - service: shell_command.trv_set_ext_temp
    data_template:
      trv_ip: "{{ states('input_text.trv_buero_ip') }}"
      ext_temp: "{{ states('sensor.mittelwert_buero_temperatur') }}"
mode: single
```

Mittelwert Sensor (Helfer)

Mittelwert_1.png Mittelwert_2.png

Image not found or type unknown Image not found or type unknown

SET Temperature Script

Ich habe bemerkt, dass die Temperatur nicht immer mit einem Befehl an die TRVs richtig gesetzt wird. Das habe ich sowohl bei den Shellys als auch bei den Danfoss Thermostaten festgestellt.

Darum habe ich nun ein Skript erstellt, das solange die Temperatur setzt, bis sie auch tatsächlich anliegt.

Aktuell suche ich noch nach der Erklärung, in welchem Modus das Skript laufen muss um Überschneidungen zu vermeiden und auch keine Endlos Schleife zu erhalten sodass es nicht endente Jobs gibt.

Um das Script variabel zu halten werden der Raum und die Zieltemperatur übergeben.

Die gesetzte Temperatur liest das Script in jedem Run erneut aus.

Das Delay für den nächsten Test sind 30s.

Das Script wird mit "turn_on" gestartet. Damit wartet die Automatisierung nicht auf das Script Ende.

```
- service: script.turn_on
  data:
    variables:
      room: "{{ room }}"
      temp: "{{ temp }}"
  target:
    entity_id: script.heizung_set_temperature
```

```
alias: Heizung Set Temperature
sequence:
  - variables:
      device: "{{ 'climate.thermostat_' + room }}"
  - repeat:
      until:
        - or:
            - condition: template
              value_template: "{{ state_attr(device, 'temperature') == temp  }}"
            - condition: template
              value_template: "{{ repeat.index  == 10 }}"
```

```
    sequence:
      - service: climate.set_temperature
        continue_on_error: true
        data:
          temperature: "{{ temp }}"
        target:
          entity_id: "{{ device }}"
      - delay:
          hours: 0
          minutes: 0
          seconds: 30
          milliseconds: 0
mode: queued
max: 20
```

Aktuell stelle ich mir die Frage, wenn sich der Sollzustand während der Script Ausführung ändert, was dann die Beste Lösung ist.

Änderungen können sein:

- Fenster Zustand
- Absenkung
- Status Heizung
- ...

Und auch falls ein Thermostat nicht erreichbar ist darf das Script nicht unendlich laufen. "{{ repeat.index <= 10 }}"

Changelog

12.03.2023 Run Trigger changed

Bugfix im set temperature script. Mit der vorigen Einstellung <= 10 wurde die Bedingung direkt beim ersten Run erfüllt :-(

```
 - condition: template
              value_template: "{{ repeat.index  == 10 }}"   <<<< NEU "=="
```

08.03.2023 Fenster Offen Prüfung

von: sensor.window == off

zu: sensor.window != on

Damit funktioniert die Automatisierung auch, wenn der Fenstersensor unavailable ist

02.03.2023: Set Temperature Skript

Es kam vor, dass die Thermostate nicht auf Einein einzigen Befehl die Temperatur eingestellt haben.

Nun übernimmt das ein Skript, dass nach Trigger alle 30s überprüft, ob die Temperatur auch ansteht.

19.02.2003: Überprüfung ob Fenster geschlossen sind bevor die Komfort Temperatur gespeichert wird

Beim Temperaturwechsel von Komfort -> Eco wird die aktuelle Komfort Temperatur in einen input_number Helfer gespeichert. Dies wird nun nur ausgeführt, wenn die Fenster geschlossen sind, also keine Absenkung bzgl. Fenster eingestellt ist.

Es kam vor, dass die Absenk Temperatur 5Grad als Komfort Temperatur gespeichert wird, wenn zum Wechsel Zeitpunkt die Fenster offen waren.

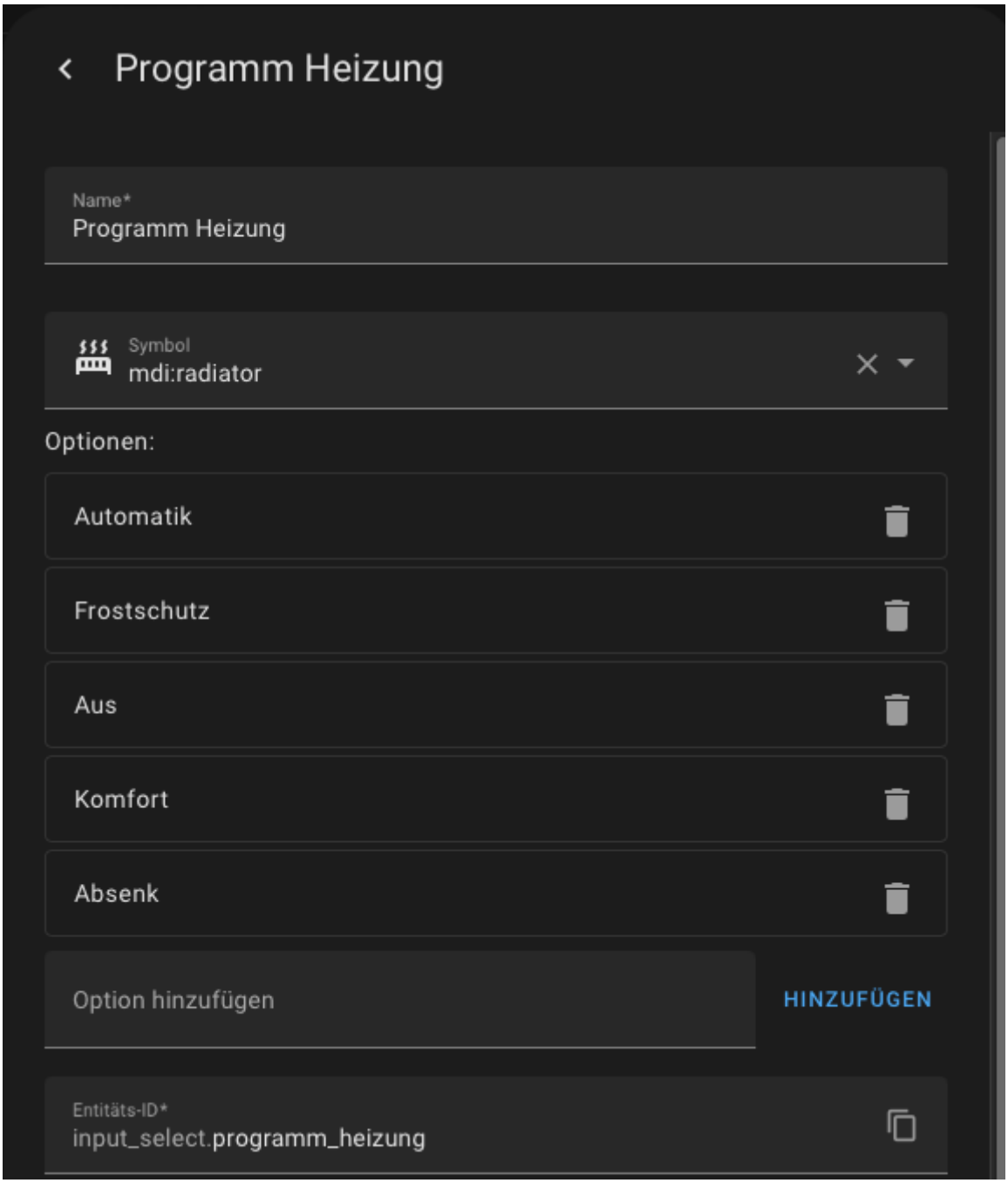
Kesselprogramm einstellen (KWB)

Die Basis dieser Funktion stammt von Michael Trebes.

Die Abfrage des Heizkreis Programms geschieht durch den bekannten modbus Adapter:

```
  # Heizkreis Programm wählen (hk_programm_t)
    # 0-Automatik, 1-Frostschutz, 2-Aus, 3-Komfort, 4-Absenk
    - name: "KWB Heizkreis Programm wählen (24589)"
      address: 24589
      scan_interval: 30
      slave: 1
      data_type: int16
      input_type: holding
      count: 1
```

Dazu einen Helfer als Dropdown definieren



Zum Schalten eine Automation erstellen:

Die "id" Werte entsprechen den Werten für die KWB Steuerung.

```
alias: Heizung Programm Wahl
description: ""
trigger:
  - platform: state
    entity_id:
      - input_select.programm_heizung
    to: Automatik
    id: "0"
```

```
  - platform: state
    entity_id:
      - input_select.programm_heizung
    to: Frostschutz
    id: "1"
  - platform: state
    entity_id:
      - input_select.programm_heizung
    to: Aus
    id: "2"
  - platform: state
    entity_id:
      - input_select.programm_heizung
    to: Komfort
    id: "3"
  - platform: state
    entity_id:
      - input_select.programm_heizung
    to: Absenk
    id: "4"
condition: []
action:
  - service: modbus.write_register
    data:
      address: 24589
      value: "{{ trigger.id }}"
      hub: heizung
      unit: 1
mode: single
```

Kopiere Temperature auf 2tes Thermostat

Stand Version 2024.1.0 funktioniert die HACS Integration "climate_group" nicht mehr.

Darum musste ich in Räumen mit 2 Thermostaten eine andere Lösung suchen.

Nun also den führenden Thermostat in der Namensgebung gesetzt und eine Automatisierung für das Kopieren der Temperatur auf den Slave erstellt.

Step 1 - Namensänderung der führenden Thermostate

climate.thermostat_ez_rechts --> climate.thermostat_ez

climate.thermostat_sz_links --> climate.thermostat_sz

climate.thermostat_studio_vorne --> climate.thermostat_studio

Step 2 - Namensänderung der 2ten Thermostate

Dies mache ich um nur einen Automatisierung erstellen zu müssen

climate.thermostat_ez_links --> climate.thermostat_ez_second

climate.thermostat_sz_rechts --> climate.thermostat_sz_second

climate.thermostat_studio_dach --> climate.thermostat_studio_second

Step 2 - Automatisierung

Heizung Copy EZ

```
alias: Heizung Copy Temperature
description: ""
trigger:
  - platform: state
    entity_id:
      - climate.thermostat_ez
      - climate.thermostat_sz
```

```
      - climate.thermostat_studio
    attribute: temperature
condition: []
action:
  - service: climate.set_temperature
    metadata: {}
    data:
      temperature: "{{ trigger.to_state.attributes.temperature }}"
    target:
      entity_id: "{{ trigger.entity_id + '_second' }}"
mode: single
```

Heizung speichere aktuelle Temperatureinstellung

Ich war bislang auf dem Weg die Komfort Temperatur über den input_number Helfer fix einzustellen.

Das Leben mit meiner Frau zeigt mir aber, dass es besser ist, die Komfort Temperatur vom Thermostat zu übernhemen.

Ich speichere also die Temperatur vom Thermostat in den input Helfer, solange sie über der ECO Temperatur ist.

```
alias: Heizung speichere aktuelle Temperatureinstellung
description: ""
trigger:
  - platform: state
    entity_id:
      - climate.thermostat_buero
      - climate.thermostat_carrera
      - climate.thermostat_ez
      - climate.thermostat_flur
      - climate.thermostat_gruen
      - climate.thermostat_kinderbad
      - climate.thermostat_lydi
      - climate.thermostat_rosa
      - climate.thermostat_studio
      - climate.thermostat_sz
      - climate.thermostat_wc
      - climate.thermostat_wcdg
      - climate.thermostat_werkstatt
      - climate.thermostat_wz
    attribute: temperature
condition: []
action:
  - variables:
      room: "{{ trigger.entity_id.split('_')[1] }}"
```

```
    device: "{{ 'climate.thermostat_' + room }}"
    eco: "{{ 'input_number.thermostat_' + room + '_eco' }}"
    comfort: "{{ 'input_number.thermostat_' + room + '_comfort' }}"
    device_temp: "{{ state_attr( device, 'temperature') }}"
    eco_temp: "{{ states(eco) }}"
    store_temp: "{{ device_temp > eco_temp }}"
  - if:
    - condition: template
      value_template: "{{ store_temp }}"
    then:
    - service: input_number.set_value
      data_template:
        value: "{{ state_attr( device, 'temperature') }}"
      target:
        entity_id: "{{ comfort }}"
mode: single
```

Heizung Büro EXT Temperature SET

Die Danfoss Thermostate können Werte für einen externen Sensor erhalten. Dies lässt sich über MQTT setzen.

```
alias: Heizung Büro EXT Temperature SET
description: ""
trigger:
  - platform: time_pattern
    minutes: /20
condition: []
action:
  - variables:
      temp: "{{ states.sensor.luftqualitaet_buero_air_temperature.state }}"
      ext_temp_value: "{{ ( temp | float * 100 ) | int }}"
  - service: mqtt.publish
    metadata: {}
    data:
      qos: 0
      retain: false
      topic: zigbee2mqtt/Thermostat Büro/set
      payload: "{\"external_measured_room_sensor\": {{ ext_temp_value }} }"
mode: single
```

> **External measured room sensor**
>
> Der Temperatursensor des TRV befindet sich konstruktionsbedingt relativ nah an der Wärmequelle (d. h. dem Warmwasser im Heizkörper). Daher gibt es Situationen, in denen die vom TRV gemessene „lokale_Temperatur“ nicht genau genug ist: Wenn der Heizkörper hinter Vorhängen oder Möbeln verdeckt ist, der Raum ziemlich groß ist oder wenn der Heizkörper selbst groß ist und die Vorlauftemperatur hoch ist Die Temperatur im Raum kann leicht um 5 °C bis 8 °C von der vom TRV gemessenen „lokalen Temperatur“ abweichen.
>
> In diesem Fall können Sie sich dafür entscheiden, einen externen Raumsensor zu verwenden und den Messwert des externen Raumsensors an die Eigenschaft „External_measured_room_sensor“ zu senden.

> Die Art und Weise, wie der TRV mit dem „External_measured_room_sensor“ arbeitet, hängt von der Einstellung der Eigenschaft „Radiator_covered“ ab:
>
> Wenn „Radiator_covered“ „false“ ist (Auto-Offset-Modus): Sie *müssen* die Eigenschaft „External_measured_room_sensor“ *mindestens* alle 3 Stunden festlegen. Nach 3 Stunden deaktiviert der TRV diese Funktion und setzt den Wert der Eigenschaft „External_measured_room_sensor“ auf -8000 (deaktiviert) zurück. Sie *sollten* die Eigenschaft „External_measured_room_sensor“ *höchstens* alle 30 Minuten oder jede Änderung der gemessenen Raumtemperatur um 0,1 K festlegen.
>
> Wenn „Radiator_covered“ „true“ ist (Raumsensormodus): Sie *müssen* den „External_measured_room_sensor“ festlegen Eigentum *mindestens* alle 30 Minuten. Nach 35 Minuten deaktiviert der TRV diese Funktion und setzt den Wert der Eigenschaft „External_measured_room_sensor“ auf -8000 (deaktiviert) zurück. Sie *sollten* die Eigenschaft „External_measured_room_sensor“ *höchstens* alle 5 Minuten oder jede 0,1K-Änderung der gemessenen Raumtemperatur festlegen.