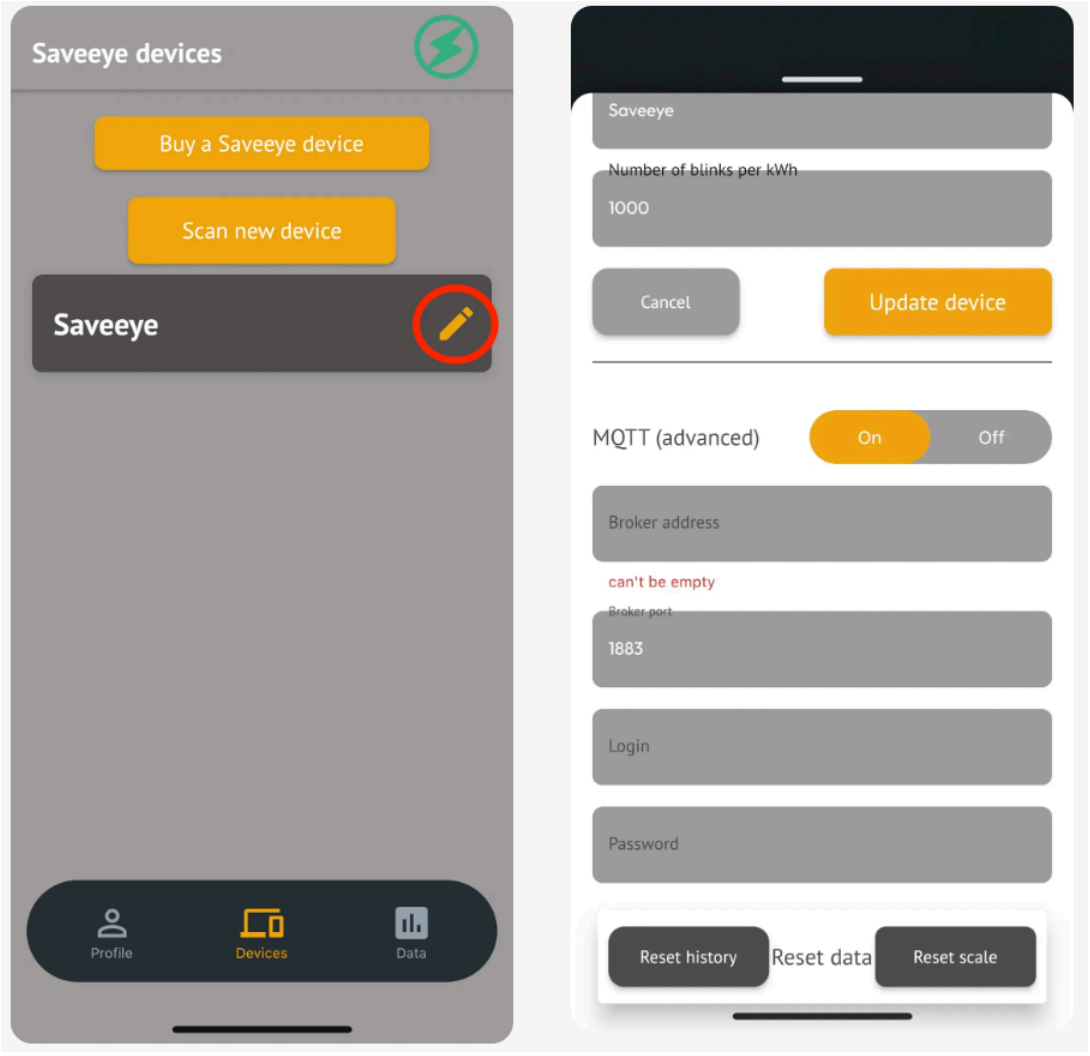
MQTT · Integration

Nu er det muligt at konfigurere Saveeye Base til at sende data direkte til en lokal MQTT broker, så forbruget eksempelvis kan vises i Home Assistant og tilsvarende systemer. Bemærk, at vi hos Saveeye ikke er eksperter i smarthome systemerne, så vi kan desværre ikke hjælpe med support til konfiguration af disse, men alene tilbyde at sende data via MQTT.
For at slå lokal MQTT til, skal du åbne indstillingerne for din Saveeye Base i appen og slå ”MQTT (avanceret)” til.
Dernæst udfylder du adresse for MQTT brokeren (den skal starte med mqtt://) samt port, brugernavn og password, hvorefter du trykker ”Opdater enhed”.
Bemærk, at brugernavn og password gemmes ukrypteret i vores database, så undgå at anvende koder, som du har brugt andre steder!
Nu vil Saveeye begynde at sende forbrugsdata med få sekunders mellemrum til din MQTT broker. Det topic, data sendes på, hedder ”saveeye/telemetry”. Her får du data om det aktuelle forbrug i Watt samt det samlede forbrug i Watt siden enheden blev startet.

Hvis det samlede forbrug nulstilles, skyldes det, at Saveeye er genstartet. Det kan fx skyldes en firmware-opdatering eller problemer med forbindelsen.



Home Assistant

Vores dygtige kunde, Christoffer, har lavet denne korte guide til opsætning af Saveeye i Home Assistant. I denne guide udregner Home Assistant selv forbrug baseret på interval mellem pulserne, men siden har vi tilføjet currentConsumptionWatt og totalConsumptionWh, så man kan undgå selv at regne det ud:

1. Indsæt disse linjer i configuration.yaml

```
template:
  sensor:
    - name: "saveeye Watt"
      unit_of_measurement: W
      state: >
        {{ 3600000000 / (states('sensor.saveeye_latestInterval') | float(0) *
1000)}}

mqtt:
  sensor:
    - state_topic: "saveeye/telemetry"
      name: saveeye_latestInterval
      value_template: '{{ value_json["latestInterval"] }}'
```

Du kan kopiere koden lige nedenfor:

template: sensor: - name: "saveeye Watt" unit_of_measurement: W state: > {{ 3600000000 / (states('sensor.saveeye_latestInterval') | float(0) * 1000)}} mqtt: sensor: - state_topic: "saveeye/telemetry" name: saveeye_latestInterval value_template: '{{ value_json["latestInterval"] }}'

2. lav kontrol at kode i ha under udviklerværktøjer/yaml tryk på tjek konfiguration
3. hvis alt er ok så tryk på genstart
4. derefter har du fået 2 nye sensorer

MQTT data fra Saveeye Remote

Når du modtager data fra Saveeye Remote via lokalt MQTT, skal du selv omregne de rå data til elforbrug.
Det skyldes, at Saveeye Base ikke har oplysninger om Remotens indstillinger for blink pr. kWh og derfor ikke kan lave omregningen.
Der kommer i udgangspunkt en besked for hvert blink, men nogle beskeder kan gå tabt.
Fra Saveeye Remote får du to talværdier. Den første hedder extenderPulses, og den stiger med én for hvert blink elmåleren giver (den nulstilles, når du skifter batteri). Er der således registreret 500 blink i en periode, er der forbrugt 0,5 kWh i perioden, såfremt din måler afgiver 1000 blink pr. kWh. Afgiver måleren 10000 blink pr. kWh, er forbruget i perioden 0,05 kWh. Så at vise forbruget

i kWh for en given time gøres ganske enkelt ved at tage antal pulser i denne time og dividere med målerens antal blink pr. kWh.
Ønsker du at vise det aktuelle forbrug i kilowatt, bliver det lidt mere kompliceret. Her skal du også bruge den anden værdi, som er timerudlæsningen fra remoten. Den hedder extenderTimestamp og har en opløsning på 1 ms.
Lad os antage, at du modtager 2 beskeder lige efter hinanden:
Besked 1: extenderPulses = 100, extenderTimestamp = 5000
Besked 2: extenderPulses = 103, extenderTimestamp = 5300
Der er altså talt 3 pulser på 300 ms, hvilket svarer til 1 puls pr. 100 ms i gennemsnit. Det svarer igen til 10 pulser i sekundet eller 36000 i timen. Så hvis din elmåler giver 10000 blink pr. kWh er det aktuelle forbrug altså 3,6 kW (= 3600 Watt). Og fortsætter det uændret i en time, vil du altså have forbrugt 3,6 kWh.

Hvis dit forbrug pludselig falder meget, for eksempel fordi du stopper med at lade elbilen, kan der komme til at gå relativt lang tid, før der kommer en ny puls, som kan opdatere visningen i realtid. Derfor sender Saveeye Remote hvert 20. sekund en ekstra besked, uanset om der er registreret blink eller ej. Hvis extenderPulses værdien i denne besked er den samme som i sidst modtagne besked, er der altså ikke registreret blink fra elmåleren. Men extenderTimestamp værdien vil naturligvis være forøget. Derfor kan du måle, hvor lang tid der er gået siden seneste blink ved at trække forrige extenderTimestamp fra nuværende extenderTimestamp og se, om tiden er længere end for eksempel de 100 ms vi så i sidste eksempel. Er tiden blevet længere, betyder det, at forbruget er faldet. Og så bør du bruge den højere værdi til at udregne det aktuelle forbrug. Det betyder, at hvis der aldrig bliver registreret et nyt blink fra elmåleren, vil forbruget bevæge sig tættere og tættere på nul, som tiden går.