Nutzung von Node-RED

Umsetzung von Prozessen zur Auslesung und Verarbeitung von Sensordaten.

- Grundlagen der Verwendung
- Vorgehen für Geräte
  - EBus Geräte (Heiztechnik)
  - TP-Link Kasa WLAN Steckdose HS110 mit Stromaufzeichnung
  - AVM FRITZ!DECT 200 Steckdose
  - Anbindung IP-Symcon Gebäudeautomation
- Beispiel Flows
  - Fahrplanmanagement Wärmepumpe Warmwasser (eBus)
  - Telegram Messaging
- Grünstromzähler (Discovergy Meter)
- CO2 freundlicher Smart-Meter

Grundlagen der Verwendung

Node-RED ist eine von der IBM entwickelte Lösung zur Verknüpfung von Hardware Geräten mit Logik und Steuerung. Die Verwendung von Node-RED zeichnet sich durch eine hohe Effizienz und einfache Bedienung des Systems aus. Theoretisch kann man mit diesem System die unterschiedlichsten Aufgabenstellungen, die sich rund um ein Haus/Unternehmen ergeben abbilden und automatisieren. Innerhalb von Casa Corrently werden jedoch lediglich die Elemente, welche für ein Energie-Management relevant sind betrachtet.

- Anbindung von Sensoren (Messgeräte, Stromzähler, IoT Devices, Smart-Home, Appliances)
- Ansteuerung von OpenEMS
- Erstellung von Last- und Erzeugungsprognosen
- Interaktion der verschiedenen Systemkomponenten
- Einfache Anpassung an die jeweiligen Gegebenheiten

Definitionen

Innerhalb von Node-RED kann im Prinzip mit jeder Art von Daten und Einheit gearbeitet werden. Zur Vereinheitlichung wurden jedoch für Casa Corrently einige Vorgaben gesetzt.

Verwendung von Watt (Auflösung, Einheit, Dimension)

Obwohl sehr viele Prozesse in der Energiewirtschaft auf Watt-Stunden (bzw. Kilowatt-Stunden) festgelegt sind, wird innerhalb von Casa Corrently versucht immer mit Watt zu arbeiten. Dies ist möglich, da im Hintergrund mit InfluxDB eine zeitreihenbasierte Datenbank zum Einsatz kommt.

Der große Vorteil einer Verwendung von Watt als Einheit ist, dass bei unterschiedlichen Sensoren/Messgeräte diese die Daten asynchron zur Verfügung stellen. Durch Bildung einer Differenz der Zeitstempel zwischen den einzelnen Messwerten kann aus Watt (Leistung) recht leicht in Watt-Stunden (Arbeit) umgerechnet werden.

Verwendung von WattStunden (nur bei Speicher SoC)

Lediglich bei der Definition von Speichern wird mit Watt-Stunden gerechnet. Hier wird von der sonst üblichen Darstellung des State-Of-Charge (SoC) in Prozentwerten abgewichen, da die Verarbeitung einfacher und nachvollziehbarer ist.

Verwendung von Ganzzahlen (keine Kommawerte)

Durch die gewählte Auflösung in Watt ist eine ausreichende Genauigkeit vorhanden, um nicht jedem Elektron hinterherzurennen und gleichzeitig den gesamten Stromfluss einer Großstadt

abbilden zu können. Es wird daher auf die Verwendung von Fließkommazahlen verzichtet (Big Integer anstelle von Double oder Float). Dies hat bei modernen 64 Bit Prozessoren zudem den Vorteil, dass mehrere Berechnungen parallel in einem Takt durchgeführt werden können.

Flows ohne Promise

Zwar erlaubt Node-RED, dass man mit Promise (Asynchronen Funktionen) arbeiten könnte, jedoch steigert dies die Komplexität von Prozessen sehr stark. Bei Casa Correntlly kolmmen daher keine Promise in den Funktionen zum Einsatz. Stattdessen werden die Flos so definiert, dass sie entweder durch einen Sensor getriggert oder durch einen `Inject Node` (Zeitschalter) von Anfang bis zum Ende ein Ereignis abarbeiten.

Dashboards in Grafana

Zwar bietet Node-RED mit `UI` eine Möglichkeit zur Visualisierung von Daten, jedoch wird diese bei Casa Correntlly nicht verwendet. Sobald Daten visualisiert werden sollen, werden diese in die InfluxDB gespeichert, wovon sie mittels Grafana visualisiert werden können.

Eine Ausnahme sind hier die Einstellungen, welche in den einzelnen Flows verwend werden. Diese vom Anwender zu setzenden Informationen für die Anpassung des Systems werden mit der Node-RED UI umgesetzt.

Beispiel: Setzen der Postleitzahl für den GrünstromIndex

Im Flow

settings.png
Image not found or type unknown

Innerhalb der UI

settings_ui.png
Image not found or type unknown

Nutzung des Node Status

Da innerhalb von Node-RED auf die Verwendung der Visualisierung (Dashboard) verzichtet wird, jedoch das Ergebnis von Anpassungen und Konfigurationen möglichst schnell sichtbar sein soll, werden die Status Elemente (grauer Text unter den Funktionen) genutzt.

node_status.png
Image not found or type unknown

MQTT für Sensoren

Auf einem Casa Correntlly System läuft ein eigener MQTT Broker (Mosquitto) an Port 1883. Geräte und Sensoren, welches dieses Protokoll unterstützen können diesen für Nachrichten nutzen.

soc_mqtt.png
Image not found or type unknown

Im Bild wird innerhalb des Flows direkt auf den lokalen MQTT Broker zugegriffen, sobald eine neue Meldung des Stromspeichers vorhanden ist wird der Speicherfüllstand in Watt-Stunden umgerechnet.

Flows für Ebenen und Fahrplanmanagement

Auf einem Flow wird entweder eine Ebene oder ein Fahrplan (Gerät) abgebildet. Auf Sub-Flows wird bislang verzichtet.

ebenen_tabs_flows.png

Zusätzlich existiert ein Flow/Tab, welcher zur Gesamtsteuerung dient.

Vorgehen für Geräte

Umsetzung von konkreten Problemstellungen und Anbindung von Geräten/Sensoren an Casa Corrently.

EBus Geräte (Heiztechnik)

EBus_Logo.png
image_not_found or type unknown

eBus ist ein im Bereich der Heizungstechnik verwendetes Schnittstellenprotokoll, das auf der seriellen Schnittstellentechnik RS232 in Twisted-Pair-Technik mit zwei Drähten aufbaut. (Wikipedia)

Zur Verwendung von eBus innerhalb von Casa Corrently hat sich die Installation der Software ebusd als nützlich erwiesen, welcher in den aktuellen Versionen eine direkte MQTT Unterstützung besitzt. Es empfiehlt sich den Daemon direkt auf der Casa Corrently Box zu installieren und wie folgt aufzurufen:

```
/usr/bin/ebusd --scanconfig --mqttclientid=ebusd --mqtthost=127.0.0.1 --mqttport=1883
```

Beispiel: Fahrplanmanagement Wärmepumpe

ebus_waermepumpensteuerrung.png
image_not_found or type unknown

In diesem Beispiel kommt eine Vaillant Wärmepumpe zum Einsatz, deren Warmwasserzubereitung über Casa Corrently optimiert wird. Zunächst wird der kombinierte GrünstromIndex (`eXgsi`) abgefragt, um dann die beste Zeit für die Warmwasserzubereitung zu definieren und im Programm (Timer) des Gerätes zu hinterlegen.

Funktion GSI integrieren

```
let gsi = global.get("eXgsi");

let hours = [];
let hwset = false;

for(let i=0;i<gsi.length;i++) {
    if(flow.get("temp_avg")>15) {
        if(gsi[i].fields.s2 > 0) {
            node.status({text:"In "+i+" Stunden"});
            if(!hwset) {
                flow.set('hwstart',Math.round(gsi[i].timestamp/1000000));
                hwset=true;
            }
        }
```

```
        hours.push(gsi[i].fields.s2 * 1000);
    } else {
        hours.push(gsi[i].fields.s20 * 450);
    }
}
msg.payload = hours;

return msg;
```

Je nach durchschnittlicher Temperatur der folgenden Tage wird entweder der Zweistundenschalter (Zeile 16) oder der 20 Stunden Schalter (Zeile 18) als Basis verwendet.

Über dein `MQTT Out Node` werden im Anschluss die Zeitschaltuhren mittels des jeweiligen MQTT Topics gesetzt. Beim Empfang einer neuen Nachricht wird der ebusd diese Nachricht direkt an die Wärmepumpe senden.

Bei der eingesetzten Wärmepumpe existiert ein spezieller "Schnellschalter" für eine einmalige Speicherladung. Wird dieser gesetzt, so wird die Warmwassertemperatur auf dem höchstmöglichen Wert gehalten (unabhängig vom Zeitplan). Sobald ein Überschuss an Strom vorhanden ist und in das Netz eingespeist werden müsste, wird dieser Schalter gesetzt.

TP-Link Kasa WLAN Steckdose HS110 mit Stromaufzeichnung

Diese Smart-Home Steckdose zeichnet dadurch aus, dass sie keine weiteren Kontroller benötigt und direkt in das WLAN eingebunden werden kann. Für die Verwendung innerhalb von Casa Corrently kommt die Strommessung zum Einsatz zum Aufbau von flexiblen Lasten bei Geräten, die dies eigentlich nicht unterstützen. (vergleiche Ebene 1)

Vorinstallierte Komponente: node-red-contrib-tplink-smarthome

hs110_kasa_smartplug.png
Image not found or type unknown

Damit die Leistung von der Steckdose empfangen werden kann, muss eine Nachricht "getMeterInfo" Nachricht gesendet werden.

Funktion: SmartPlug:getMeterInfo

```
msg.payload="getMeterInfo";
return msg;
```

Die Rückgabe des Leistungswertes steht dann im Feld `power_mw` zur Verfügung und kann weiter verarbeitet werden.

Funktion: Wirkleistung

```
msg.payload = Math.round(msg.payload.power_mw / 1000);
node.status({text:"P: "+msg.payload+" W"});
// msg.payload =  msg.payload;
const subSUM='Consumption';

if(isNaN(msg.payload)) msg.payload = 0;

let sum = flow.get(subSUM) * 1;
```

```
if(isNaN(sum)) sum = 0;

let previous = context.get("previous") * 1;
if(isNaN(previous)) previous = 0;

if(flow.get("SaldoID") != context.get("SaldoID")) {
    context.set("SaldoID",flow.get("SaldoID"));
} else {
    sum -= previous;
}
sum += msg.payload;

context.set("previous",msg.payload);

flow.set(subSUM,sum);

return msg;
```

Intelligentes Schalten (Last Flexibilität)

Das hier gezeigte Beispiel nutzt lediglich einen Einschalter, der das angeschlossene Gerät für eine Stunde aktiviert, wenn innerhalb dieser Stunde nicht ein weiterer Einschalte Impuls kommt (gesetzt über die Funktion "Timer" der TP-Link Kasa App).

flexon.png
Image not found or type unknown

In diesem Fall wird ein Webhook innerhalb von Node-RED genutzt, um von einer anderen Anwendung/Dienst die Steckdose zu aktivieren.

Funktion: Turn On

```
msg.payload=true;
return msg;
```

AVM FRITZ!DECT 200 Steckdose

Die FRITZ!DECT Steckdose zeichnet sich durch ihre nahtlose Anbindung an ein auf die Fritz Box basierendes System aus. Zusätzlich hat diese Steckdose eine Energieaufzeichnung, die sich innerhalb von Casa Corrently nutzen lässt.

Vorinstallierte Komponente: node-red-contrib-fritzapi

Beispiel: Erzeugungsmessung Micro PV Anlage

balkonpv_micropv_fritz.png
Image not found or type unknown

Die Mini-PV Anlage hat keine eigene Messung der Erzeugung, weshalb diese von der FRITZ!DECT gemessen wird. Damit die angesprochene Fritz Box die richtige Steckdose auswählen und als Payload-Information an die Funktion Wirkleistung übergeben kann, ist die AIN der Steckdose zu setzen.

Funktion: BalkonPV

```
msg.payload.ain = "087610221618";
return msg;
```

Anbindung IP-Symcon Gebäudeautomation

Die Software IP-Symcon ist eine Gesamtlösung für die Gebäudeautomation, welche in ihren aktuellen Versionen problemlos via MQTT in Casa Corrently integriert werden kann. Es wird empfohlen, dass IP-Symcon nicht auf demselben Raspberry Pi installiert wird, da der Ressourcenbedarf ansonsten sehr schnell zu hoch werden kann.

Da es Überschneidungen zwischen Casa Corrently und IP-Symcon hinsichtlich der Aufgaben/Funktionen gibt, wird folgende Aufteilung empfohlen:

- Automatisierung => IP Symcon
- Energie Management => Casa Corrently

ipsymcon_mqtt.png
Image not found or type unknown

Innerhalb von IP-Symcon wird eine neue Instanz des `MQTT Configurator` erstellt. Dadurch können Variablen innerhalb von IP-Symcon als MQTT Nachricht genutzt werden (Beschreiben mit RequestAction):

```
RequestAction( 37917, $p2);
```

Innerhalb von Casa Corrently wird der durch IP-Symcon bereitgestellt MQTT Broker eingebunden:

mqtt_symcon.png
Image not found or type unknown

Der Wert der Variable `$p2` von Symcon steht dann als `msg.payload` innerhalb von Node-RED zur Verfügung.

Beispiel Flows

Flows, die mit geringem Anpassungsaufwand und Konfiguration in Casa Corrently übernommen werden können.

Fahrplanmanagement Wärmepumpe Warmwasser (eBus)

Im Exemplarischen Wärmepumpenfahrplan wird davon ausgegangen, dass die WP für 2 Stunden Strom bezieht, wenn sie lediglich Warmwasser (WW) zubereitet (Sommermonate) und 20 Stunden Strom benötigt, wenn auch die Heizfunktion benötigt wird.

s.h. auch eBus Anbindung

- flow_fahrplan_ebus_ww_ehp.png
  Image not found or type unknown

Flow


```
[
    {
        "id": "120b1f7b.ff6e21",
        "type": "tab",
        "label": "Fahrplan - Wärmepumpe",
        "disabled": false,
        "info": "Im Exemplarischen Wärmepumpenfahrplan wird davon ausgegangen, dass die WP für 2 Stunden Strom bezieht, wenn sie lediglich Warmwasser (WW) zubereitet (Sommermonate) und 20 Stunden Strom benötigt, wenn auch die Heizfunktion benötigt wird. \n\nUnterscheidung:\n - Durchschnitt > 15° = nur WW\n - Durchschnitt < 15° = WW + Heizung"
    },
    {
        "id": "23d88f0c.9ff71",
        "type": "openweathermap",
        "z": "120b1f7b.ff6e21",
        "name": "Casa Corrently Weather",
        "wtype": "forecast",
        "lon": "8.800295",
        "lat": "49.342178",
```

```
        "city": "",
        "country": "",
        "language": "de",
        "x": 430,
        "y": 80,
        "wires": [
            [
                "1a79c7f6.50c608"
            ]
        ]
    },
    {
        "id": "c314a744.a0c058",
        "type": "inject",
        "z": "120b1f7b.ff6e21",
        "name": "1h",
        "topic": "",
        "payload": "",
        "payloadType": "date",
        "repeat": "3600",
        "crontab": "",
        "once": true,
        "onceDelay": "5",
        "x": 210,
        "y": 80,
        "wires": [
            [
                "23d88f0c.9ff71"
            ]
        ]
    },
    {
        "id": "1a79c7f6.50c608",
        "type": "function",
        "z": "120b1f7b.ff6e21",
        "name": "Temperatur Stats",
        "func": "let min=99;\nlet cnt=0;\nlet sum=0;\n\nfor(let
i=0;(i<=msg.payload.length)&&(i<24);i++) {\n    sum+=msg.payload[i].main.temp;\n
cnt++;\n    if(i<24) {\n        if(msg.payload[i].main.temp < min) {\n
min=msg.payload[i].main.temp;\n        }\n    }\n}\nlet avg =
```

```
sum/cnt;\nflow.set('temp_min',min);\nflow.set('temp_avg',avg);\nnode.status({text:\"Avg:
\"+avg+\"°C\"});\nreturn msg;",
        "outputs": 1,
        "noerr": 0,
        "x": 790,
        "y": 80,
        "wires": [
            [
                "331a1037.576a5"
            ]
        ]
    },
    {
        "id": "331a1037.576a5",
        "type": "function",
        "z": "120b1f7b.ff6e21",
        "name": "GSI integrieren",
        "func": "let gsi = global.get(\"eXgsi\");\n\nlet hours = [];\nlet hwset =
false;\n\nfor(let i=0;i<gsi.length;i++) {\n    if(flow.get(\"temp_avg\")>15) {\n
if(gsi[i].fields.s2 > 0) {\n            node.status({text:\"In \"+i+\"
Stunden\"});\n            if(!hwset) {\n
flow.set('hwstart',Math.round(gsi[i].timestamp/1000000));\n
hwset=true;\n            }\n        }\n        \n       hours.push(gsi[i].fields.s2 *
1000);\n    } else {\n       hours.push(gsi[i].fields.s20 * 450);\n    }\n}\nmsg.payload =
hours;\n\nreturn msg;",
        "outputs": 1,
        "noerr": 0,
        "x": 400,
        "y": 180,
        "wires": [
            [
                "19814032.62214",
                "42cf6acb.56bbb4",
                "cb226a7c.26af58"
            ]
        ]
    },
    {
        "id": "19814032.62214",
        "type": "function",
```

```
        "z": "120b1f7b.ff6e21",
        "name": "Flex Wirkleistung",
        "func": "let watt =  msg.payload[0];\nconst subSUM='FlexConsumption';\n\nlet sum =
global.get(subSUM) * 1;\nif(isNaN(sum)) sum = 0;\n\nlet previous = context.get(\"previous\") *
1;\nif(isNaN(previous)) previous = 0;\n\nif(global.get(\"FlexSaldoID\") !=
context.get(\"FlexSaldoID\")) {\n
context.set(\"FlexSaldoID\",global.get(\"FlexSaldoID\"));\n    let options =
global.get(\"FlexOptions\");\n    options.push(msg.payload);\n
global.set(\"FlexOptions\",options);\n} else {\n    sum -= previous;\n}\nsum +=
watt;\n\ncontext.set(\"previous\",watt);\n\nglobal.set(subSUM,sum);\nif(watt > 0) {\n
flow.set(\"hwcOperatingMode\",\"on\");\n} else {\n
flow.set(\"hwcOperatingMode\",\"auto\");\n}    \nreturn msg;",
        "outputs": 1,
        "noerr": 0,
        "x": 590,
        "y": 560,
        "wires": [
            [
                "4aaf8b92.9c52c4"
            ]
        ]
    },
    {
        "id": "c273455e.fabee8",
        "type": "mqtt in",
        "z": "120b1f7b.ff6e21",
        "name": "",
        "topic": "ebusd/ehp/HwcTemp",
        "qos": "2",
        "datatype": "auto",
        "broker": "33c6e085.87181",
        "x": 360,
        "y": 760,
        "wires": [
            [
                "6bcfb76.656a448"
            ]
        ]
    },
    {
```

```
        "id": "6bcfb76.656a448",
        "type": "function",
        "z": "120b1f7b.ff6e21",
        "name": "Strip Status",
        "func": "msg.payload  = msg.payload.substr(0,msg.payload.indexOf(';')) *
1;\nnode.status({text:msg.payload});\nreturn msg;",
        "outputs": 1,
        "noerr": 0,
        "x": 570,
        "y": 760,
        "wires": [
            [
                "ba4fdafd.45bd88"
            ]
        ]
    },
    {
        "id": "ba4fdafd.45bd88",
        "type": "influxdb out",
        "z": "120b1f7b.ff6e21",
        "influxdb": "3116358f.1e80ea",
        "name": "hwtemp",
        "measurement": "hwtemp",
        "precision": "",
        "retentionPolicy": "",
        "x": 1060,
        "y": 760,
        "wires": []
    },
    {
        "id": "9dcc6642.e066c8",
        "type": "mqtt out",
        "z": "120b1f7b.ff6e21",
        "name": "ebusd/hwc/Timer.Monday/set",
        "topic": "ebusd/hwc/Timer.Monday/set",
        "qos": "1",
        "retain": "true",
        "broker": "33c6e085.87181",
        "x": 1120,
        "y": 160,
```

```
        "wires": []
    },
    {
        "id": "42cf6acb.56bbb4",
        "type": "function",
        "z": "120b1f7b.ff6e21",
        "name": "Mo-Fr",
        "func": "let tz_offset = 2;\nlet rt = 2;\n\nlet start = new
Date(flow.get('hwstart')).getHours() + tz_offset;\nlet end = start + rt;\nmsg.payload = start
+\":15;\"+end+\":15;-:-;-:-;-:-;-:-;Mo-Fr\";\nnode.status({text:\"Start:
\"+start+\":15\"});\nreturn msg;",
        "outputs": 1,
        "noerr": 0,
        "x": 750,
        "y": 180,
        "wires": [
            [
                "9dcc6642.e066c8",
                "c4db5437.eb92f8",
                "bcce9475.f904c8",
                "bb4f2986.fa5c98",
                "198fe03d.032c6"
            ]
        ]
    },
    {
        "id": "c4db5437.eb92f8",
        "type": "mqtt out",
        "z": "120b1f7b.ff6e21",
        "name": "ebusd/hwc/Timer.Tuesday/set",
        "topic": "ebusd/hwc/Timer.Tuesday/set",
        "qos": "1",
        "retain": "true",
        "broker": "33c6e085.87181",
        "x": 1130,
        "y": 220,
        "wires": []
    },
    {
        "id": "bcce9475.f904c8",
```

```
        "type": "mqtt out",
        "z": "120b1f7b.ff6e21",
        "name": "ebusd/hwc/Timer.Wednesday/set",
        "topic": "ebusd/hwc/Timer.Wednesday/set",
        "qos": "1",
        "retain": "true",
        "broker": "33c6e085.87181",
        "x": 1140,
        "y": 280,
        "wires": []
    },
    {
        "id": "bb4f2986.fa5c98",
        "type": "mqtt out",
        "z": "120b1f7b.ff6e21",
        "name": "ebusd/hwc/Timer.Thursday/set",
        "topic": "ebusd/hwc/Timer.Thursday/set",
        "qos": "1",
        "retain": "true",
        "broker": "33c6e085.87181",
        "x": 1130,
        "y": 340,
        "wires": []
    },
    {
        "id": "198fe03d.032c6",
        "type": "mqtt out",
        "z": "120b1f7b.ff6e21",
        "name": "ebusd/hwc/Timer.Friday/set",
        "topic": "ebusd/hwc/Timer.Friday/set",
        "qos": "1",
        "retain": "true",
        "broker": "33c6e085.87181",
        "x": 1120,
        "y": 400,
        "wires": []
    },
    {
        "id": "bfa4d83b.08dce8",
        "type": "mqtt out",
```

```
        "z": "120b1f7b.ff6e21",
        "name": "ebusd/hwc/Timer.Saturday/set",
        "topic": "ebusd/hwc/Timer.Saturday/set",
        "qos": "1",
        "retain": "true",
        "broker": "33c6e085.87181",
        "x": 1130,
        "y": 460,
        "wires": []
    },
    {
        "id": "497e21df.6117a",
        "type": "mqtt out",
        "z": "120b1f7b.ff6e21",
        "name": "ebusd/hwc/Timer.Sunday/set",
        "topic": "ebusd/hwc/Timer.Sunday/set",
        "qos": "1",
        "retain": "true",
        "broker": "33c6e085.87181",
        "x": 1120,
        "y": 500,
        "wires": []
    },
    {
        "id": "cb226a7c.26af58",
        "type": "function",
        "z": "120b1f7b.ff6e21",
        "name": "Sa-So",
        "func": "let tz_offset = 2;\nlet rt = 2;\n\nlet start = new
Date(flow.get('hwstart')).getHours() + tz_offset;\nlet end = start + rt;\nmsg.payload = start
+\":15;\"+end+\":15;-:-;-:-;-:-;-:-;Sa-So\";\nreturn msg;",
        "outputs": 1,
        "noerr": 0,
        "x": 750,
        "y": 240,
        "wires": [
            [
                "bfa4d83b.08dce8",
                "497e21df.6117a"
            ]
```

```
        ]
    },
    {
        "id": "4aaf8b92.9c52c4",
        "type": "function",
        "z": "120b1f7b.ff6e21",
        "name": "Set hwcOperatingMode",
        "func": "msg.payload =
flow.get(\"hwcOperatingMode\");\nnode.status({text:msg.payload});\nreturn msg;",
        "outputs": 1,
        "noerr": 0,
        "x": 810,
        "y": 560,
        "wires": [
            [
                "8e6eb620.37bf88"
            ]
        ]
    },
    {
        "id": "8e6eb620.37bf88",
        "type": "mqtt out",
        "z": "120b1f7b.ff6e21",
        "name": "ebusd/hwc/OperatingMode/set",
        "topic": "ebusd/hwc/OperatingMode/set",
        "qos": "1",
        "retain": "true",
        "broker": "33c6e085.87181",
        "x": 1130,
        "y": 560,
        "wires": []
    },
    {
        "id": "e5f66c42.8870a",
        "type": "comment",
        "z": "120b1f7b.ff6e21",
        "name": "WarmWasser Programm",
        "info": "",
        "x": 810,
        "y": 140,
```

```
        "wires": []
    },
    {
        "id": "38c7aca7.85be64",
        "type": "influxdb in",
        "z": "120b1f7b.ff6e21",
        "influxdb": "3116358f.1e80ea",
        "name": "Netzsaldo",
        "query": "SELECT mean(\"value\") FROM \"e2saldo\" WHERE time>now()-10m GROUP BY
time(5m) fill(linear)",
        "rawOutput": false,
        "precision": "",
        "retentionPolicy": "",
        "x": 320,
        "y": 700,
        "wires": [
            [
                "58d95890.fb3df8"
            ]
        ]
    },
    {
        "id": "2834f7ea.e27e38",
        "type": "inject",
        "z": "120b1f7b.ff6e21",
        "name": "20m",
        "topic": "",
        "payload": "",
        "payloadType": "date",
        "repeat": "1200",
        "crontab": "",
        "once": true,
        "onceDelay": "20",
        "x": 130,
        "y": 700,
        "wires": [
            [
                "38c7aca7.85be64"
            ]
        ]
```

```
    },
    {
        "id": "58d95890.fb3df8",
        "type": "function",
        "z": "120b1f7b.ff6e21",
        "name": "Ladung bei Überschuss (+250W)",
        "func": "if(msg.payload[msg.payload.length-1].mean > 250) {\n    msg.payload = \"on\"\n} else {\n    msg.payload = \"auto\"\n}\nnode.status({text:msg.payload});\nreturn msg;",
        "outputs": 1,
        "noerr": 0,
        "x": 640,
        "y": 700,
        "wires": [
            [
                "3b6aa897.d31468",
                "1dcb5c61.951c74"
            ]
        ]
    },
    {
        "id": "3b6aa897.d31468",
        "type": "mqtt out",
        "z": "120b1f7b.ff6e21",
        "name": "ebusd/hwc/OperatingMode/set",
        "topic": "ebusd/hwc/OperatingMode/set",
        "qos": "1",
        "retain": "true",
        "broker": "33c6e085.87181",
        "x": 1130,
        "y": 640,
        "wires": []
    },
    {
        "id": "1dcb5c61.951c74",
        "type": "switch",
        "z": "120b1f7b.ff6e21",
        "name": "HW Load",
        "property": "payload",
        "propertyType": "msg",
```

```
        "rules": [
            {
                "t": "eq",
                "v": "on",
                "vt": "str"
            }
        ],
        "checkall": "true",
        "repair": false,
        "outputs": 1,
        "x": 900,
        "y": 700,
        "wires": [
            [
                "eca7dae8.212f18"
            ]
        ]
    },
    {
        "id": "eca7dae8.212f18",
        "type": "mqtt out",
        "z": "120b1f7b.ff6e21",
        "name": "ebusd/mc/load/set",
        "topic": "ebusd/mc/load/set",
        "qos": "1",
        "retain": "true",
        "broker": "33c6e085.87181",
        "x": 1090,
        "y": 700,
        "wires": []
    },
    {
        "id": "33c6e085.87181",
        "type": "mqtt-broker",
        "z": "",
        "name": "Local Mosquito (Port 1883)",
        "broker": "localhost",
        "port": "1883",
        "clientid": "node-red",
        "usetls": false,
```

```
        "compatmode": false,
        "keepalive": "60",
        "cleansession": true,
        "birthTopic": "",
        "birthQos": "0",
        "birthPayload": "",
        "closeTopic": "",
        "closeQos": "0",
        "closePayload": "",
        "willTopic": "",
        "willQos": "0",
        "willPayload": ""
    },
    {
        "id": "3116358f.1e80ea",
        "type": "influxdb",
        "z": "",
        "hostname": "127.0.0.1",
        "port": "8086",
        "protocol": "http",
        "database": "casacorrently",
        "name": "InfluxDB",
        "usetls": false,
        "tls": ""
    }
]
```

```
[
    {
        "id": "120b1f7b.ff6e21",
        "type": "tab",
        "label": "Fahrplan - Wärmepumpe",
        "disabled": false,
        "info": "Im Exemplarischen Wärmepumpenfahrplan wird davon ausgegangen, dass die WP für 2 Stunden Strom bezieht, wenn sie lediglich Warmwasser (WW) zubereitet (Sommermonate) und 20 Stunden Strom benötigt, wenn auch die Heizfunktion benötigt wird. \n\nUnterscheidung:\n - Durchschnitt > 15° = nur WW\n - Durchschnitt < 15° = WW + Heizung"
    },
    {
        "id": "23d88f0c.9ff71",
```

```
        "type": "openweathermap",
        "z": "120b1f7b.ff6e21",
        "name": "Casa Corrently Weather",
        "wtype": "forecast",
        "lon": "8.800295",
        "lat": "49.342178",
        "city": "",
        "country": "",
        "language": "de",
        "x": 430,
        "y": 80,
        "wires": [
            [
                "1a79c7f6.50c608"
            ]
        ]
    },
    {
        "id": "c314a744.a0c058",
        "type": "inject",
        "z": "120b1f7b.ff6e21",
        "name": "1h",
        "topic": "",
        "payload": "",
        "payloadType": "date",
        "repeat": "3600",
        "crontab": "",
        "once": true,
        "onceDelay": "5",
        "x": 210,
        "y": 80,
        "wires": [
            [
                "23d88f0c.9ff71"
            ]
        ]
    },
    {
        "id": "1a79c7f6.50c608",
        "type": "function",
```

```
        "z": "120b1f7b.ff6e21",
        "name": "Temperatur Stats",
        "func": "let min=99;\nlet cnt=0;\nlet sum=0;\n\nfor(let
i=0;(i<=msg.payload.length)&&(i<24);i++) {\n    sum+=msg.payload[i].main.temp;\n
cnt++;\n    if(i<24) {\n        if(msg.payload[i].main.temp < min) {\n
min=msg.payload[i].main.temp;\n        }\n    }\n}\nlet avg =
sum/cnt;\nflow.set('temp_min',min);\nflow.set('temp_avg',avg);\nnode.status({text:\"Avg:
\"+avg+\"°C\"});\nreturn msg;",
        "outputs": 1,
        "noerr": 0,
        "x": 790,
        "y": 80,
        "wires": [
            [
                "331a1037.576a5"
            ]
        ]
    },
    {
        "id": "331a1037.576a5",
        "type": "function",
        "z": "120b1f7b.ff6e21",
        "name": "GSI integrieren",
        "func": "let gsi = global.get(\"eXgsi\");\n\nlet hours = [];\nlet hwset =
false;\n\nfor(let i=0;i<gsi.length;i++) {\n    if(flow.get(\"temp_avg\")>15) {\n
if(gsi[i].fields.s2 > 0) {\n            node.status({text:\"In \"+i+\"
Stunden\"});\n             if(!hwset) {\n
flow.set('hwstart',Math.round(gsi[i].timestamp/1000000));\n
hwset=true;\n            }\n        }\n        \n       hours.push(gsi[i].fields.s2 *
1000);\n    } else {\n       hours.push(gsi[i].fields.s20 * 450);\n    }\n}\nmsg.payload =
hours;\n\nreturn msg;",
        "outputs": 1,
        "noerr": 0,
        "x": 400,
        "y": 180,
        "wires": [
            [
                "19814032.62214",
                "42cf6acb.56bbb4",
                "cb226a7c.26af58"
```

```
            ]
        ]
    },
    {
        "id": "19814032.62214",
        "type": "function",
        "z": "120b1f7b.ff6e21",
        "name": "Flex Wirkleistung",
        "func": "let watt =  msg.payload[0];\nconst subSUM='FlexConsumption';\n\nlet sum =
global.get(subSUM) * 1;\nif(isNaN(sum)) sum = 0;\n\nlet previous = context.get(\"previous\") *
1;\nif(isNaN(previous)) previous = 0;\n\nif(global.get(\"FlexSaldoID\") !=
context.get(\"FlexSaldoID\")) {\n
context.set(\"FlexSaldoID\",global.get(\"FlexSaldoID\"));\n    let options =
global.get(\"FlexOptions\");\n    options.push(msg.payload);\n
global.set(\"FlexOptions\",options);\n} else {\n    sum -= previous;\n}\nsum +=
watt;\n\ncontext.set(\"previous\",watt);\n\nglobal.set(subSUM,sum);\nif(watt > 0) {\n
flow.set(\"hwcOperatingMode\",\"on\");\n} else {\n
flow.set(\"hwcOperatingMode\",\"auto\");\n}    \nreturn msg;",
        "outputs": 1,
        "noerr": 0,
        "x": 590,
        "y": 560,
        "wires": [
            [
                "4aaf8b92.9c52c4"
            ]
        ]
    },
    {
        "id": "c273455e.fabee8",
        "type": "mqtt in",
        "z": "120b1f7b.ff6e21",
        "name": "",
        "topic": "ebusd/ehp/HwcTemp",
        "qos": "2",
        "datatype": "auto",
        "broker": "33c6e085.87181",
        "x": 360,
        "y": 760,
        "wires": [
```

```
            [
                "6bcfb76.656a448"
            ]
        ]
    },
    {
        "id": "6bcfb76.656a448",
        "type": "function",
        "z": "120b1f7b.ff6e21",
        "name": "Strip Status",
        "func": "msg.payload  = msg.payload.substr(0,msg.payload.indexOf(';')) *
1;\nnode.status({text:msg.payload});\nreturn msg;",
        "outputs": 1,
        "noerr": 0,
        "x": 570,
        "y": 760,
        "wires": [
            [
                "ba4fdafd.45bd88"
            ]
        ]
    },
    {
        "id": "ba4fdafd.45bd88",
        "type": "influxdb out",
        "z": "120b1f7b.ff6e21",
        "influxdb": "3116358f.1e80ea",
        "name": "hwtemp",
        "measurement": "hwtemp",
        "precision": "",
        "retentionPolicy": "",
        "x": 1060,
        "y": 760,
        "wires": []
    },
    {
        "id": "9dcc6642.e066c8",
        "type": "mqtt out",
        "z": "120b1f7b.ff6e21",
        "name": "ebusd/hwc/Timer.Monday/set",
```

```
        "topic": "ebusd/hwc/Timer.Monday/set",
        "qos": "1",
        "retain": "true",
        "broker": "33c6e085.87181",
        "x": 1120,
        "y": 160,
        "wires": []
    },
    {
        "id": "42cf6acb.56bbb4",
        "type": "function",
        "z": "120b1f7b.ff6e21",
        "name": "Mo-Fr",
        "func": "let tz_offset = 2;\nlet rt = 2;\n\nlet start = new
Date(flow.get('hwstart')).getHours() + tz_offset;\nlet end = start + rt;\nmsg.payload = start
+\":15;\"+end+\":15;-:-;-:-;-:-;-:-;Mo-Fr\";\nnode.status({text:\"Start:
\"+start+\":15\"});\nreturn msg;",
        "outputs": 1,
        "noerr": 0,
        "x": 750,
        "y": 180,
        "wires": [
            [
                "9dcc6642.e066c8",
                "c4db5437.eb92f8",
                "bcce9475.f904c8",
                "bb4f2986.fa5c98",
                "198fe03d.032c6"
            ]
        ]
    },
    {
        "id": "c4db5437.eb92f8",
        "type": "mqtt out",
        "z": "120b1f7b.ff6e21",
        "name": "ebusd/hwc/Timer.Tuesday/set",
        "topic": "ebusd/hwc/Timer.Tuesday/set",
        "qos": "1",
        "retain": "true",
        "broker": "33c6e085.87181",
```

```
        "x": 1130,
        "y": 220,
        "wires": []
    },
    {
        "id": "bcce9475.f904c8",
        "type": "mqtt out",
        "z": "120b1f7b.ff6e21",
        "name": "ebusd/hwc/Timer.Wednesday/set",
        "topic": "ebusd/hwc/Timer.Wednesday/set",
        "qos": "1",
        "retain": "true",
        "broker": "33c6e085.87181",
        "x": 1140,
        "y": 280,
        "wires": []
    },
    {
        "id": "bb4f2986.fa5c98",
        "type": "mqtt out",
        "z": "120b1f7b.ff6e21",
        "name": "ebusd/hwc/Timer.Thursday/set",
        "topic": "ebusd/hwc/Timer.Thursday/set",
        "qos": "1",
        "retain": "true",
        "broker": "33c6e085.87181",
        "x": 1130,
        "y": 340,
        "wires": []
    },
    {
        "id": "198fe03d.032c6",
        "type": "mqtt out",
        "z": "120b1f7b.ff6e21",
        "name": "ebusd/hwc/Timer.Friday/set",
        "topic": "ebusd/hwc/Timer.Friday/set",
        "qos": "1",
        "retain": "true",
        "broker": "33c6e085.87181",
        "x": 1120,
```

```
        "y": 400,
        "wires": []
    },
    {
        "id": "bfa4d83b.08dce8",
        "type": "mqtt out",
        "z": "120b1f7b.ff6e21",
        "name": "ebusd/hwc/Timer.Saturday/set",
        "topic": "ebusd/hwc/Timer.Saturday/set",
        "qos": "1",
        "retain": "true",
        "broker": "33c6e085.87181",
        "x": 1130,
        "y": 460,
        "wires": []
    },
    {
        "id": "497e21df.6117a",
        "type": "mqtt out",
        "z": "120b1f7b.ff6e21",
        "name": "ebusd/hwc/Timer.Sunday/set",
        "topic": "ebusd/hwc/Timer.Sunday/set",
        "qos": "1",
        "retain": "true",
        "broker": "33c6e085.87181",
        "x": 1120,
        "y": 500,
        "wires": []
    },
    {
        "id": "cb226a7c.26af58",
        "type": "function",
        "z": "120b1f7b.ff6e21",
        "name": "Sa-So",
        "func": "let tz_offset = 2;\nlet rt = 2;\n\nlet start = new
Date(flow.get('hwstart')).getHours() + tz_offset;\nlet end = start + rt;\nmsg.payload = start
+\":15;\"+end+\":15;-:-;-:-;-:-;-:-;Sa-So\";\nreturn msg;",
        "outputs": 1,
        "noerr": 0,
        "x": 750,
```

```
        "y": 240,
        "wires": [
            [
                "bfa4d83b.08dce8",
                "497e21df.6117a"
            ]
        ]
    },
    {
        "id": "4aaf8b92.9c52c4",
        "type": "function",
        "z": "120b1f7b.ff6e21",
        "name": "Set hwcOperatingMode",
        "func": "msg.payload =
flow.get(\"hwcOperatingMode\");\nnode.status({text:msg.payload});\nreturn msg;",
        "outputs": 1,
        "noerr": 0,
        "x": 810,
        "y": 560,
        "wires": [
            [
                "8e6eb620.37bf88"
            ]
        ]
    },
    {
        "id": "8e6eb620.37bf88",
        "type": "mqtt out",
        "z": "120b1f7b.ff6e21",
        "name": "ebusd/hwc/OperatingMode/set",
        "topic": "ebusd/hwc/OperatingMode/set",
        "qos": "1",
        "retain": "true",
        "broker": "33c6e085.87181",
        "x": 1130,
        "y": 560,
        "wires": []
    },
    {
        "id": "e5f66c42.8870a",
```

```
        "type": "comment",
        "z": "120b1f7b.ff6e21",
        "name": "WarmWasser Programm",
        "info": "",
        "x": 810,
        "y": 140,
        "wires": []
    },
    {
        "id": "38c7aca7.85be64",
        "type": "influxdb in",
        "z": "120b1f7b.ff6e21",
        "influxdb": "3116358f.1e80ea",
        "name": "Netzsaldo",
        "query": "SELECT mean(\"value\") FROM \"e2saldo\" WHERE time>now()-10m GROUP BY
time(5m) fill(linear)",
        "rawOutput": false,
        "precision": "",
        "retentionPolicy": "",
        "x": 320,
        "y": 700,
        "wires": [
            [
                "58d95890.fb3df8"
            ]
        ]
    },
    {
        "id": "2834f7ea.e27e38",
        "type": "inject",
        "z": "120b1f7b.ff6e21",
        "name": "20m",
        "topic": "",
        "payload": "",
        "payloadType": "date",
        "repeat": "1200",
        "crontab": "",
        "once": true,
        "onceDelay": "20",
        "x": 130,
```

```
        "y": 700,
        "wires": [
            [
                "38c7aca7.85be64"
            ]
        ]
    },
    {
        "id": "58d95890.fb3df8",
        "type": "function",
        "z": "120b1f7b.ff6e21",
        "name": "Ladung bei Überschuss (+250W)",
        "func": "if(msg.payload[msg.payload.length-1].mean > 250) {\n    msg.payload = \"on\"\n} else {\n    msg.payload = \"auto\"\n}\nnode.status({text:msg.payload});\nreturn msg;",
        "outputs": 1,
        "noerr": 0,
        "x": 640,
        "y": 700,
        "wires": [
            [
                "3b6aa897.d31468",
                "1dcb5c61.951c74"
            ]
        ]
    },
    {
        "id": "3b6aa897.d31468",
        "type": "mqtt out",
        "z": "120b1f7b.ff6e21",
        "name": "ebusd/hwc/OperatingMode/set",
        "topic": "ebusd/hwc/OperatingMode/set",
        "qos": "1",
        "retain": "true",
        "broker": "33c6e085.87181",
        "x": 1130,
        "y": 640,
        "wires": []
    },
    {
```

```
        "id": "1dcb5c61.951c74",
        "type": "switch",
        "z": "120b1f7b.ff6e21",
        "name": "HW Load",
        "property": "payload",
        "propertyType": "msg",
        "rules": [
            {
                "t": "eq",
                "v": "on",
                "vt": "str"
            }
        ],
        "checkall": "true",
        "repair": false,
        "outputs": 1,
        "x": 900,
        "y": 700,
        "wires": [
            [
                "eca7dae8.212f18"
            ]
        ]
    },
    {
        "id": "eca7dae8.212f18",
        "type": "mqtt out",
        "z": "120b1f7b.ff6e21",
        "name": "ebusd/mc/load/set",
        "topic": "ebusd/mc/load/set",
        "qos": "1",
        "retain": "true",
        "broker": "33c6e085.87181",
        "x": 1090,
        "y": 700,
        "wires": []
    },
    {
        "id": "33c6e085.87181",
        "type": "mqtt-broker",
```

```
        "z": "",
        "name": "Local Mosquito (Port 1883)",
        "broker": "localhost",
        "port": "1883",
        "clientid": "node-red",
        "usetls": false,
        "compatmode": false,
        "keepalive": "60",
        "cleansession": true,
        "birthTopic": "",
        "birthQos": "0",
        "birthPayload": "",
        "closeTopic": "",
        "closeQos": "0",
        "closePayload": "",
        "willTopic": "",
        "willQos": "0",
        "willPayload": ""
    },
    {
        "id": "3116358f.1e80ea",
        "type": "influxdb",
        "z": "",
        "hostname": "127.0.0.1",
        "port": "8086",
        "protocol": "http",
        "database": "casacorrently",
        "name": "InfluxDB",
        "usetls": false,
        "tls": ""
    }
]
```

Beispiel Flows

Telegram Messaging

telegram.png
Image not found or type unknown

Mit Hilfe des Messaging Dienstes Telegram lassen sich sehr individuelle Benachrichtigungen und Einstellungen vornehmen. Bei der Verwendung mit Casa Corrently ist aufgefallen, dass man einen Mittelweg zwischen zu vielen Nachrichten und genügend Informationen finden muss.

An dieser Stelle daher die Empfehlung, dass man an die Usecases (Anwendungsfälle) denkt und entsprechend die Benachrichtigungen aufbaut.

Funktion: Answer with Globals

```
let res = {
    content: '',
    type: 'message',
    chatId: 215089981
};
if(msg.payload.type == 'message') {
    if(typeof global.get(msg.payload.content) != 'undefined')  {
        res.content= ''+global.get(msg.payload.content);
    } else
    if(msg.payload.content == "keys") {
        res.content = ''  + global.keys();
    } else
    if(msg.payload.content == "kritisch") {
        res.content = 'in '  + Math.round((global.get("SoC_min_ts") - new Date().getTime()
)/3600000) +'  Stunden ('  + global.get('SoC_min')+'  Wh) '+ new
Date(global.get("SoC_min_ts")).toLocaleString('de-DE',{timeZone:'Europe/Berlin'});
    } else
    if(msg.payload.content == "auto") {
        let eXgsi = global.get('eXgsi');
        let ts = 0;
        for(let i=0;(i<eXgsi.length)&&(ts==0);i++) {
            if(eXgsi[i].fields.s4 == 1) {
                ts = Math.round(eXgsi[i].timestamp / 1000000);
            }
```

```
        }
        res.content = 'Beginne Ladung in '  + Math.round((ts - new Date().getTime() )/3600000)
+' Stunden '+  new Date(ts).toLocaleString('de-DE',{timeZone:'Europe/Berlin'});
    }
}
msg.payload = res;
return msg;
```

Autoladung

Man sitzt noch im Auto und könnte den Ladetimer programmieren. Welche Uhrzeit ist ratsam?

Die Antwort steht über die `Schalter` des kombinierten GrünstromIndex zu Verfügung. Dieser ist in der Global Variable `eXgsi` abrufbar.

Eine Antwort auf die Nachricht "auto" per Instagrmm wird in den Zeilen 16-25 erstellt. Der Startzeitpunkt ist somit der Beginn der besten 4 Stunden laut kombiniertem GrünstromIndex.

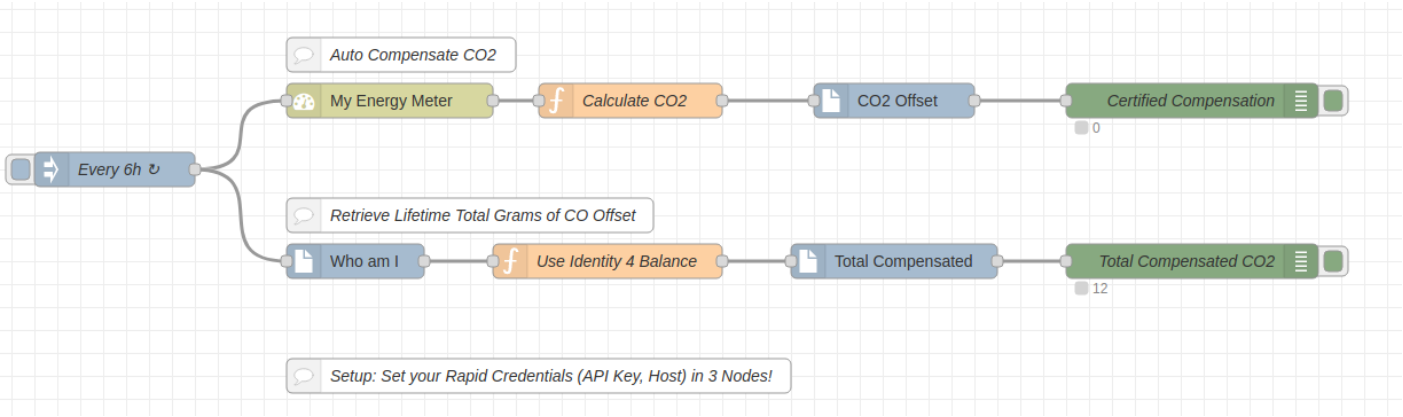
telegram_gsi.png
Image not found or type unknown

Grünstromzähler (Discovergy Meter)

Nutzung von Discovergy Zählern (STROMDAO, Awattar, Commetering,...) innerhalb von Casa Corrently zur wirtschaftlichen Betrachtung der Kosten.

CO2 freundlicher Smart-Meter

Mithilfe dieses Flows kann der Strombezug an einem Discovegy Zähler CO2 neutral gestaltet werden. Gelöst wird dies durch eine automatische Kompensation auf Basis von Gold-Standard Credits (VER), die dem durch den Strombezug entstanden Klimagasen zugeordnet werden. (Hintergrund: Ökostrom und Klimaneutralität.)



Download des Flows (Node-RED) bei GIST.