Projekt: solaranzeige.de

API Erweiterung

Dokument AP021

Stand Juni 2024

Inhaltsverzeichnis

Übersicht:

Mit Hilfe der API Schnittstelle (*Application Programming Interface*, Programmierschnittstelle) können beliebige Daten aus dem lokalen Netzwerk in die Datenbanken der Solaranzeige geschrieben werden. So können z.B. Gaszähler oder Stromzähler mit anderen Programmen ausgelesen werden und dann die Daten über die API Schnittstelle in die Datenbank der Solaranzeige geschrieben werden. Genauso ist es möglich die zuletzt gespeicherten Daten aus den Datenbanken zu lesen. Die Daten werden in einem XML Dokument per HTTP POST übertragen. Dabei können mehrere Daten in mehreren Measurements und mehreren Datenbanken mit einem XML Dokument geschrieben bzw. gelesen werden. Als Antwort kommt ein XML Dokument zurück, was den Erfolg oder die Fehlermeldung beinhaltet. Die Daten können mit den verschiedensten Programmiersprachen übertragen werden. Bei fast allen Programmiersprachen gibt es fertige XML Unterstützung. Die Daten werden sofort in die Datenbank geschrieben. Es können jede Sekunde Daten übertragen werden, ob das bei einer SD-Karte Sinn macht ist eine andere Sache.

Die API Schnittstelle soll es vereinfachen, Daten mit Node-RED, IOBroker und SmartHome Geräten auszutauschen.

Daten in die Datenbank schreiben:

Das Herzstück ist die XML Datei, die per HTTP POST Request übertragen werden muss.

```
<?xml version="1.0" encoding="UTF-8" ?>
<solaranzeige>
  <version>1.0</version>
  <in_out>in</in_out>
  <database name="test">
   <measurement name="api">
    <fieldname name="Batterie_Strom"><value >-3.4</value></fieldname>
   </measurement>
  </database>
</solaranzeige>
```

Folgende Elemente sind Pflicht und dürfen nur einmal pro XML Datei vorkommen:

<?xml version="1.0" encoding="UTF-8" ?>
<solaranzeige>

```
  <version>1.0</version>
  <in_out>in</in_out>
</solaranzeige>
```

<in_out>in</in_out>
„in" muss dort stehen, wenn man Daten in die Datenbank schreiben will, „out", wenn man Daten aus der Datenbank lesen möchte.

Zwischen den Elementen <solaranzeige> und </solaranzeige> darf folgendes Element mehrfach vorkommen:

<database name="test">

</database>

Im Attribut „name" muss der Name der Datenbank stehen. Diese Datenbank muss in der InfluxDB vorhanden sein, ansonsten kommt ein Fehler zurück.

Zwischen
<database name="test"> und </database>

kännen mehrfach folgende Elemente stehen:

<measurement name="api">

</measurement>

Das Attribut „name" beinhaltet wieder den Namen des Measurements in das die Daten geschrieben werden sollen. **Existiert das Measurement nicht, wird es automatisch angelegt!**

Zischen <measurement name="api"> und </measurement>

können mehrfach folgende Elemente stehen:

<fieldname name="Batterie_Spannung">
<value type="num">48,3</value>
</fieldname>

Hier befinden sich die eigentlichen Datenwerte. Bei dem Element <value> kann man das Attribut „type" angeben. Das bewirkt zum Beispiel, dass ein Wert 34,5 in 34.5 umgewandelt wird, da es nur so in der Datenbank in einem Nummerischen Feld abgespeichert werden kann. Es gibt folgende Typen:
'num' = Zahlen +/-
'str' = Zeichenketten
'hex' = Hex Werte
Fehlt das Attribut „type" dann erfolgt keine Wandlung und es wird so wie gesendet in die Influx Datenbank gespeichert. D.h. Nummerische Werte dürfen kein Komma enthalten.

Mit diesen XML Elementen kann ein Wert oder viele Werte in unterschiedlichen Measurements und unterschiedlichen Datenbanken abgespeichert werden. **Achtung die Datenbank muss vorhanden sein, also eventuell erst vorher anlegen!**
Übertragen werden muss die xml Datei mit folgendem URL Aufruf:

http://solaranzeige.local/api/control.php oder

http://<IP-Adresse des Raspberry>/api/control.php

Fehlermeldungen:

Wird ein XML Dokument an http://solaranzeige.local übertragen, so kommt ein XML Dokument wieder zurück.

Wurden alle Daten verarbeitet und gespeichert, dann sieht die Antwort so aus:

```
<?xml version="1.0" encoding="UTF-8"?>
<solaranzeige>
  <version>1.0</version>
  <error_code>0</error_code>
  <error></error>
</solaranzeige>
```

Wichtig sind die Elemente <error_code> und <error>

<error_code> kann nur 0 oder 1 sein. 0 = Alles in Ordnung, 1 = Fehler.

<error> </error> Im Fehlerfall steht hier ein Text, mit dem der Fehler näher beschrieben wird.

Es gibt auf dem Raspberry auch noch eine LOG Datei: /var/www/log/api.log

Erster Test der API Schnittstelle, mit einer XML Datei:

Zuerst in der Influx Datenbank eine Datenbank „test" anlegen!

Influx
create database test
quit

dann in das Unterverzeichnis api springen:
cd /var/www/html/api

dann
./send.sh api_test.xml
aufrufen.
Damit wird mit einem HTTP POST Request die xml Datei api_test.xml an die API Schnittstelle gesendet. In der influx Datenbank „test" muss es jetzt die beiden Measurement api und Batterie mit verschiedenen Werten geben.

Nach dem Testen kann die Datenbank „test" wieder gelöscht werden: „drop database test"

Daten aus der Datenbank lesen:

Das Herzstück ist die XML Datei, die per HTTP POST Request übertragen werden muss.

```
<?xml version="1.0" encoding="UTF-8" ?>
<solaranzeige>
  <version>1.0</version>
  <in_out>out</in_out>
  <database></database>
</solaranzeige>
```

Wenn nur das TAG Element <database> ohne ein Attribut „name" gesendet wird, kommen alle vorhandenen Datenbanknamen zurück. Auf Groß und Kleinschreibung der Datenbanknamen ist zu achten!

```
<?xml version="1.0" encoding="UTF-8" ?>
<solaranzeige>
  <version>1.0</version>
  <in_out>out</in_out>
  <database name="test">
   <measurement></measurement>
  </database>
</solaranzeige>
```

Beinhaltet das TAG Element <database> ein Attribut „name" mit einem vorhandenen Datanbanknamen und ist das Element <measurement> ohne Attribut „name" vorhanden, werden alle Measurement Namen zurück gegeben.

```
<?xml version="1.0" encoding="UTF-8" ?>
<solaranzeige>
  <version>1.0</version>
  <in_out>out</in_out>
  <database name="test">
   <measurement name="api">
    <fieldname></fieldname>
   </measurement>
  </database>
</solaranzeige>
```

Werden die Elemente <database> und <measurement> jeweils mit Attribut „name" und das Element <fieldname> ohne Attribut „name" übermittelt, werden alle Variablennamen (Feldnamen) ohne Werte ausgegeben.

```
<?xml version="1.0" encoding="UTF-8" ?>
<solaranzeige>
  <version>1.0</version>
  <in_out>out</in_out>
  <database name="test">
   <measurement name="api">
    <fieldname name="*"></fieldname>
   </measurement>
  </database>
</solaranzeige>
```

Enthält das Attribut name des Elementes <feldname> den Wert „*", dann werden alle Variablen mit den aktuellen Werten, des letzten gespeicherten Eintrags, ausgegeben.

```
<?xml version="1.0" encoding="UTF-8" ?>
<solaranzeige>
  <version>1.0</version>
  <in_out>out</in_out>
  <database name="solaranzeige">
   <measurement name="Info">
    <fieldname name="Datum"></fieldname>
    <fieldname name="Firmware"></fieldname>
   </measurement>
  </database>
</solaranzeige>
```

Enthält das Element <fieldname name="Batterie_Strom"> ein Attribut „name" dann wird nur dieser Wert, der zuletzt gespeichert wurde, zurück gegeben.

Das Element <fieldname> mit Attribut „name" kann mehrfach enthalten sein, wenn man nur bestimmte Werte auslesen möchte.

```
<?xml version="1.0" encoding="UTF-8" ?>
<solaranzeige>
  <version>1.0</version>
  <in_out>out</in_out>
  <database name="solaranzeige">
   <measurement name="Info">
    <fieldname name="*"></fieldname>
   </measurement>
   <measurement name="Service">
    <fieldname name="*"></fieldname>
   </measurement>
  </database>
</solaranzeige>
```

Auf die gleiche Weise können auch mehrere Measurements aus einer Datenbank oder mehreren Datenbanken ausgelesen werden.

Auslesen der Datenbank, mit der XML Datei api_lesen.xml:

In das Unterverzeichnis api springen:
cd /var/www/html/api

dann
./send.sh api_lesen.xml
aufrufen.
Damit wird mit einem HTTP POST Request die xml Datei api_lesen.xml an die API Schnittstelle gesendet.

Die Ausgabe sollte so in etwa aussehen.

```
pi@solaranzeige:/var/www/html/api # ./send.sh api_lesen.xml

<?xml version="1.0" encoding="UTF-8"?>
<solaranzeige>
  <version>1.0</version>
  <in_out>out</in_out>
  <database name="solaranzeige">
    <measurement name="Info">
      <fieldname name="Datum">20.01.2021</fieldname>
      <fieldname name="Firmware">1.19</fieldname>
      <fieldname name="Objekt">DemoFinca</fieldname>
      <fieldname name="Produkt">A052</fieldname>
    </measurement>
  </database>
</solaranzeige>
```

Installation:

Vor Image Version 4.7.7 bitte folgendes machen:

strg + F1 drücken oder sich mit PUTTY einloggen.

Auf der Konsole in das Home Verzeichnis von pi springen und folgendes eingeben:

„cd /home/pi“

„sudo wget -N http://solaranzeige.de/api_install“

„sudo php api_install“

Dann 1 x „w“ drücken, dann sollte die Installation fertig sein.

Jetzt gibt es ein neues Verzeichnis /var/www/html/api

Haben Sie ein Image nach 4.7.7 dann ist alles schon installiert.

Update:

Möchte man die aktuellste Version erhalten, nachdem man schon das API installiert hat bitte folgendes machen:

strg + F1 drücken oder sich mit PUTTY einloggen.

Auf der Konsole in das Home Verzeichnis von pi springen und folgendes eingeben:

„cd /home/pi“

„sudo php api_install“

Ab einschließlich Image Version 4.7.7 oder später ist keine Installation bzw. Update mehr nötig. Dann ist alles schon im Image enthalten und wird auch automatisch auf dem neuesten Stand gehalten.