Heckert Solar
Die Energiekompetenz

Benutzerhandbuch – App REST/JSON Lese-/Schreibzugriff

Version 2021.2

Inhalt

1. Einleitung

Sehr geehrte Kundin, sehr geehrter Kunde,

vielen Dank, dass Sie sich für die »App REST/JSON Lese-/Schreibzugriff« entschieden haben. Gerne können Sie uns Ihre Anregungen mitteilen, damit wir die Qualität unserer Produkte noch weiter verbessern können.

2. App REST/JSON Lese-/Schreibzugriff

Die Apps REST/JSON Lese- und Schreibzugriff ermöglichen den Zugriff auf das EMS im lokalen Netzwerk über eine an REST W angelehnte Schnittstelle. Es ist dafür notwendig, dass das zugreifende Gerät direkten Zugriff auf die IP-Adresse des EMS hat – also z. B. im gleichen physischen Netzwerk angeschlossen ist.

2.1. Lesezugriff

Die App REST/JSON Lese-/Schreibzugriff stellt eine an REST angelehnte Schnittstelle zur Verfügung, mit der Datenpunkte im System ausgelesen werden können.

Diese App ist im Standard-Lieferumfang des EMS enthalten.

Die Basis-Adresse für die REST-Zugriffe lautet `http://<BENUTZER>:<PASSWORT>@<IP>:80/rest`

- `http` ist das Protokoll
- `<BENUTZER>` ist der Benutzername. Da die Authentifizierung lediglich über das Passwort erfolgt, kann hier ein beliebiger Wert (z. B. "x") angegeben werden
- `<PASSWORT>` ist das Passwort des Benutzers. Der Standard "Gast"-Benutzer im EMS hat das Passwort "**user**"
- `<IP>` ist die IP-Adresse des EMS
- `80` ist der Port für die REST/JSON-Api (optional)

Wenn Ihr EMS also die lokale IP-Adresse '192.168.0.23' hat, lautet die Basis-Adresse für REST-Zugriffe `http://x:user@192.168.0.23:80/rest`

Aus Sicherheitsgründen werden *Simple Authentication Requests* nicht unterstützt, da bei dieser Variante Passwörter über die URL übertragen werden. Für REST Calls muss explizit Header Authentication verwendet werden.

2.1.1. /channel Endpunkt

Der `/channel` Endpunkt ermöglicht den Zugriff auf einzelne Datenpunkte, sogenannte "Channels", im System.

Die vollständige Adresse des Endpunkts lautet:

http://x:<PASSWORT>@<IP>:80/rest/channel/<KOMPONENTE>/<KANAL>

- <KOMPONENTE> ist die ID der Komponente
- <KANAL> ist die ID des Kanals

2.1.2. Datenpunkte

Die folgenden Datenpunkte können ausgelesen werden:

Datenpunkt	Beschreibung	Einheit
_sum/State	Zustand des Systems (0: Ok, 1:Info, 2:Warning, 3:Fault)	
_sum/EssSoc	Ladezustand des Speichers	Prozent [%]
_sum/EssActivePower	Wirkleistung des Speichers	Watt [W]
_sum/GridActivePower	Wirkleistung am Netzanschlusspunkt	Watt [W]
_sum/ProductionActivePower	Wirkleistung Erzeuger	Watt [W]
_sum/ProductionAcActivePower	Wirkleistung AC Erzeuger	Watt [W]
_sum/ProductionDcActualPower	Wirkleistung DC Erzeuger	Watt [W]
_sum/ConsumptionActivePower	Wirkleistung Verbraucher	Watt [W]
_sum/EssActiveChargeEnergy	Energie Speicherbeladung	WattHours [Wh]
_sum/EssActiveDischargeEnergy	Energie Speicherentladung	WattHours [Wh]
_sum/GridBuyActiveEnergy	Energie Netzbezug	WattHours [Wh]
_sum/GridSellActiveEnergy	Energie Netzeinspeisung	WattHours [Wh]
_sum/ProductionActiveEnergy	Energie Erzeugung	WattHours [Wh]
_sum/ProductionAcActiveEnergy	Energie AC Erzeugung	WattHours [Wh]
_sum/ProductionDcActiveEnergy	Energie DC Erzeugung	WattHours [Wh]
_sum/ConsumptionActiveEnergy	Energie Verbraucher	WattHours [Wh]
_sum/EssActivePowerL1	Wirkleistung Phase 1 Speicher	Watt [W]
_sum/EssActivePowerL2	Wirkleistung Phase 2 Speicher	Watt [W]
_sum/EssActivePowerL3	Wirkleistung Phase 3 Speicher	Watt [W]
_sum/GridActivePowerL1	Wirkleistung Phase 1 Netz	Watt [W]
_sum/GridActivePowerL2	Wirkleistung Phase 2 Netz	Watt [W]
_sum/GridActivePowerL3	Wirkleistung Phase 3 Netz	Watt [W]
_sum/ProductionAcActivePowerL1	Wirkleistung Phase 1 Erzeuger	Watt [W]
_sum/ProductionAcActivePowerL2	Wirkleistung Phase 2 Erzeuger	Watt [W]
_sum/ProductionAcActivePowerL3	Wirkleistung Phase 3 Erzeuger	Watt [W]

_sum/ConsumptionActivePowerL1	Wirkleistung Phase 1 Verbraucher	Watt [W]
_sum/ConsumptionActivePowerL2	Wirkleistung Phase 2 Verbraucher	Watt [W]
_sum/ConsumptionActivePowerL3	Wirkleistung Phase 3 Verbraucher	Watt [W]

2.1.3. Beispiel 1 - Abfrage des Ladezustands: cURL

Das Kommandozeilen-Programm cURL ist sowohl unter Windows und Linux vorinstalliert.

Um den Ladezustand des Stromspeichers auszulesen, senden Sie einen `GET`-Request an die Adresse: http://x:user@192.168.0.23:80/rest/channel/_sum/EssSoc

Sie erhalten eine Antwort im JSON-Format:

Windows

Der folgende Befehl speichert die Antwort im JSON-Format in die Datei *out.json*

```
>curl -o out.json http://x:user@192.168.0.23:80/rest/channel/_sum/EssSoc
```

Um den Inhalt der Datei auszugeben, nutzen Sie:

```
>type out.json
```

Ausgabe:

```
{"address":"_sum/EssSoc","type":"INTEGER","accessMode":"RO","text":"","unit":"%","valu
e":99}
```

Den Wert des Ladezustands finden Sie unter *value*. Im Beispiel oben beträgt er 99 %.

Linux

Der folgende Befehl speichert die Antwort im JSON-Format in die Datei *out.json*

```
$curl -o out.json http://x:user@192.168.0.23:80/rest/channel/_sum/EssSoc
```

Um den Inhalt der Datei auszugeben, nutzen Sie:

```
>cat out.json
```

Ausgabe:

```
{"address":"_sum/EssSoc","type":"INTEGER","accessMode":"RO","text":"","unit":"%","valu
e":99}
```

Den Wert des Ladezustands finden Sie unter *value*. Im Beispiel oben beträgt er 99 %.

2.1.4. Beispiel 2 - Abfrage des Ladezustands: Python

Python Versionen für Windows und Linux erhalten Sie hier: https://www.python.org/downloads/

Um den Ladezustand des Stromspeichers auszulesen, muss ebenfalls ein `GET`-Request an die Adresse:
`http://x:user@192.168.0.23:80/rest/channel/_sum/EssSoc` gesendet werden.

Hierfür kann die *requests* Bibliothek genutzt werden, die zu Beginn importiert werden muss:

```
import requests

url = 'http://192.168.0.23:80/rest/channel/_sum/EssSoc'

user = 'x'
password = 'user'

session = requests.Session()
session.auth = (user, password)

response = session.get(url)
response.raise_for_status()
```

Der Befehl liefert eine Antwort im JSON-Format. Diese kann mit dem folgenden Befehl ausgegeben werden:

```
print(response.text)
```

Ausgabe:

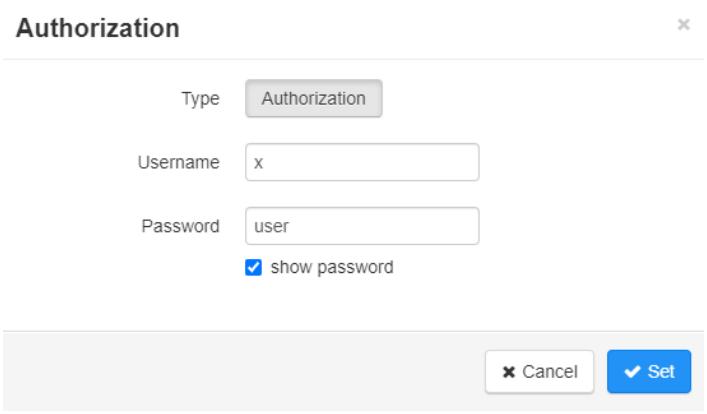
```
{"address":"_sum/EssSoc","type":"INTEGER","accessMode":"RO","text":"","unit":"%","valu
e":99}
```

Den Wert des Ladezustands finden Sie unter *value*. Im Beispiel oben beträgt er 99 %.

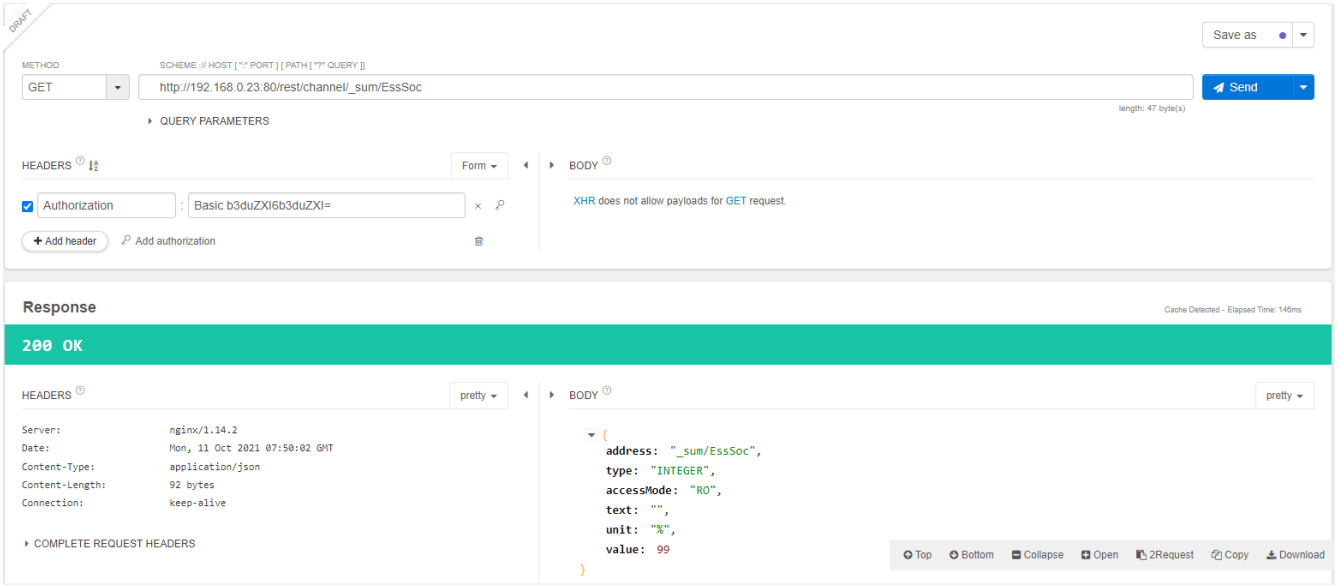
2.1.5. Beispiel 3 - Abfrage des Ladezustands: Talend API Tester

Talend API Tester ist eine Erweiterung für Google Chrome, die es ermöglicht, REST APIs zu testen.

Zunächst muss ein *Authorization* Header hinzugefügt werden:



Anschließend kann der GET-Request ausgeführt werden.



Den Wert des Ladezustands finden Sie unter *value*. Im Beispiel oben beträgt er 99 %.

2.2. Schreibzugriff

Die App REST/JSON Lese-/Schreibzugriff stellt eine an REST W angelehnte Schnittstelle zur Verfügung, mit der Datenpunkte im System beschrieben werden können.

Diese App ist **nicht** im Standard-Lieferumfang des EMS enthalten. Sie kann jedoch nachträglich jederzeit nachgerüstet werden. Kontaktieren Sie uns hierfür!

Die Verwendung des Schreibzugriffs ist nicht über den Gast-Zugang möglich. Stattdessen ist ein gesonderter Kundenzugang notwendig. Hierfür ist das Passwort "owner" zu verwenden. Der Nutzername kann wie beim Lesezugriff beliebig gewählt werden.

Sämtliche Schreibzugriffe müssen als POST-Requests gesendet werden.

2.2.1. Timeout

Die App REST/JSON Lese-/Schreibzugriff verfügt über einen konfigurierbaren Timeout. Im Standard

ist dieser auf 60 Sekunden konfiguriert. Er sorgt dafür, dass ein Vorgabewert 60 Sekunden lang aktiv bleibt. Sobald ein neuer Vorgabewert geschrieben wird, wird der neue Wert verwendet. Erfolgt kein neuer Vorgabewert innerhalb von 60 Sekunden, fällt die Steuerung auf den nachrangig priorisierten Controller zurück - z. B. Vorgabe einer "0"-Leistung oder Eigenverbrauchsoptimierung.

2.2.2. /channel Endpunkt

Genau wie bei der App REST/JSON Lese-/Schreibzugriff, wird über den Endpunkt `/channel` der Zugriff auf einzelne Datenpunkte, sogenannte "Channels", im System ermöglicht.

Die vollständige Adresse des Endpunkts lautet:

http://x:<PASSWORT>@<IP>:80/rest/channel/<KOMPONENTE>/<KANAL>

- `<KOMPONENTE>` ist die ID der Komponente.
- `<KANAL>` ist die ID des Kanals

2.2.3. Datenpunkte

Die folgenden Datenpunkte können beschrieben werden:

Datenpunkt	Beschreibung	Einheit
`_ess0/SetActivePowerEquals`	Wirkleistungsvorgabe	Watt [W]
`_ess0/SetReactivePowerEquals`	Blindleistungsvorgabe	VoltAmpereReactive [VAR]
`_ess0/SetActivePowerLessOrEquals`	(Maximale) Wirkleistungsvorgabe	Watt [W]
`_ess0/SetReactivePowerLessOrEquals`	(Maximale) Blindleistungsvorgabe	VoltAmpereReactive [VAR]
`_ess0/SetActivePowerGreaterOrEquals`	(Minimale) Wirkleistungsvorgabe	Watt [W]
`_ess0/SetReactivePowerGreaterOrEquals`	(Minimale) Wirkleistungsvorgabe	VoltAmpereReactive [VAR]

Mehr Informationen zum Channel `SetActivePowerEquals` und anderen Channels zur Leistungsvorgabe finden Sie im Glossar

2.2.4. Beispiel 1 - Wirkleistungsvorgabe: Python

Um z. B. dem ersten Stromspeichersystem (bzw. Stromspeicher-Cluster) eine Entladeleistung von 5 kW vorzugeben, senden Sie einen `POST`-Request an die Adresse `http://192.168.0.23:80/rest/channel/ess0/SetActivePowerEquals` mit der Leistungsvorgabe im *JSON* Format

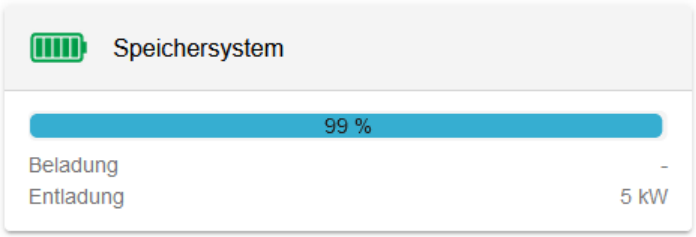
```
{
  "value": 5000
}
```

Eine Entladung des Speichers wird durch einen positiven, eine Beladung durch einen negativen Wert umgesetzt.

Hierfür kann die *requests* Bibliothek genutzt werden, die zu Beginn importiert werden muss:

```
import requests

url = 'http://192.168.0.23:80/rest/channel/ess0/SetActivePowerEquals'

user = 'x'
password = 'owner'

session = requests.Session()
session.auth = (user, password)

data = {"value": 5000}

response = session.post(url, json = data)
response.raise_for_status()
```

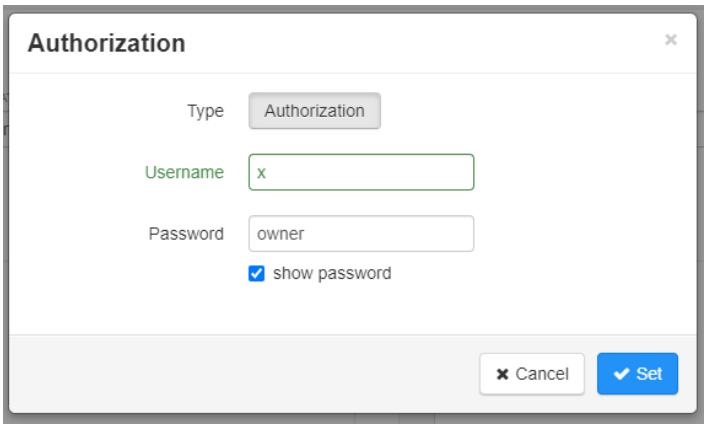
Die korrekte Durchführung des Requests kann über einen anschließenden `GET`-Request oder das Online-Monitoring (s. unten) überprüft werden.



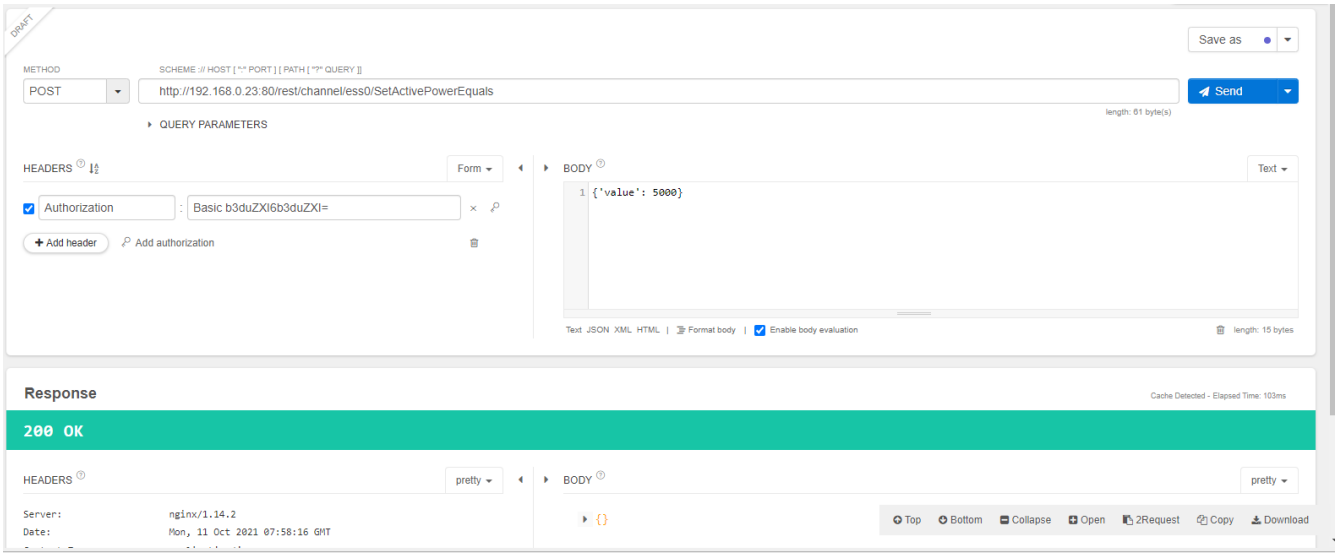
2.2.5. Beispiel 2 - Wirkleistungsvorgabe: Talend API Tester

Talend API Tester ist eine Erweiterung für Google Chrome, die es ermöglicht, REST APIs zu testen.

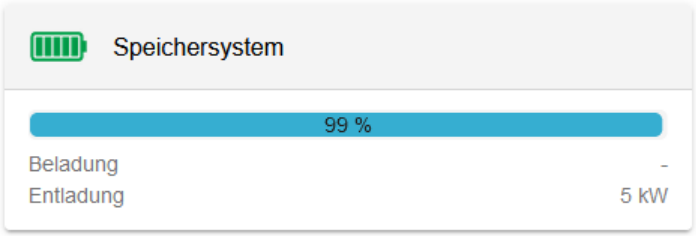
Zunächst muss ein *Authorization* Header hinzugefügt werden:



Anschließend kann der `POST`-Request ausgeführt werden.



Die korrekte Durchführung des requests kann über einen anschließenden `GET`-Request oder das Online-Monitoring (s. unten) überprüft werden.



3. Aktivierung der App

Falls Sie die App direkt mit Ihrem Speicher bestellt haben, wurde sie bereits vorkonfiguriert und ist sofort aktiv. Falls Sie die App nachrüsten, muss das EMS noch per Fernwartung konfiguriert werden. Kontaktieren Sie hierzu bitte unseren Service und geben Sie Ihre EMS-Nr. (z. B. „fems123“) an, sowie um welche App es sich handelt.

4. Kontakt

Für Unterstützung wenden Sie sich bitte an:

Symphon-E Service

Telefon Service: +49 (0) 371 45 85 68 – 100

E-Mail Service: symphon-e@heckert-solar.com