

Benutzerhandbuch – FEMS App Lese- /Schreibzugriff für Heim-, Gewerbe- und Industriespeicher

Version 2023.6.1

Inhalt

1. Einleitung	2
2. Installation der App	2
3. Modbus/TCP - Lese-/Schreibzugriff	2
3.1. Voraussetzungen	2
3.2. Grundlagen Modbus/TCP	2
3.3. Lesezugriff	3
3.4. Modbus-Tabelle	3
3.5. Beispiel 1: Lesezugriff Batterieladezustand mit QModMaster	11
3.6. Schreibzugriff	13
3.7. Beispiel 2: Schreibzugriff auf EssActivePower mit QModMaster	14
4. REST/JSON - Lese-/Schreibzugriff	15
4.1. Voraussetzungen	15
4.2. Grundlagen REST/JSON	16
4.3. Lesezugriff	16
4.4. Schreibzugriff	21
5. Kontakt	25

1. Einleitung

1. Einleitung

Sehr geehrte Kundin, sehr geehrter Kunde,

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

2. Installation der App

Mit der Bestellung der »FEMS App Lese-/Schreibzugriff für Heim-, Gewerbe- und Industriespeicher« haben Sie einen 16-stelligen Lizenzschlüssel erhalten. Mittels diesem Lizenzschlüssel können Sie die App eigenständig im FEMS App Center einlösen.

Eine Anleitung zur Vorgehensweise finden Sie [hier](#).



Die Schnittstelle für den Lesezugriff ist bereits ab Werk inklusive und vorinstalliert.

3. Modbus/TCP - Lese-/Schreibzugriff

Diese Anleitung dient der Beschreibung des Lese-/Schreibzugriffs auf ein FENECON Stromspeichersystem mittels Modbus/TCP API. Zunächst werden Grundlagen zum Protokoll beschrieben. Anschließend wird die Funktionsweise der Schnittstelle erklärt.

3.1. Voraussetzungen

Das auf das Stromspeichersystem zugreifende Gerät (z.B. Notebook/PC) muss direkten Zugriff auf die IP-Adresse des FEMS haben – also z. B. im gleichen physischen Netzwerk angeschlossen sein.

3.2. Grundlagen Modbus/TCP

Das Modbus-Protokoll ist ein Kommunikationsprotokoll, das auf einer Client/Server-Architektur basiert. Es wurde 1979 von Gould-Modicon für die Kommunikation mit seinen speicherprogrammierbaren Steuerungen ins Leben gerufen. In der Industrie hat sich der Modbus zu einem De-facto-Standard entwickelt, da es sich um ein offenes Protokoll handelt. Seit 2007 ist die Version Modbus TCP Teil der Norm IEC 61158.

[Wikipedia: Modbus/TCP](#)

Mittels Modbus können ein Client (z. B. ein PC/EMS) und mehrere Server (z. B. Mess- und Regelsysteme, Batteriespeicher, PV-Anlage, Ladestation E-Auto) verbunden werden. Es gibt zwei Versionen: Eine für die serielle Schnittstelle (EIA-232 und EIA-485) und eine für Ethernet. In dieser Anleitung wird die Version für Ethernet beschrieben. Hierbei werden TCP/IP-Pakete verwendet, um die Daten zu übermitteln.

Lese- und Schreibzugriffe sind auf folgende Objekttypen möglich:

Objekttyp	Zugriff	Größe	Funktionscode
Einzelner Ein-/Ausgang „Coil“	Lesen & Schreiben	1-bit	01 / 05 / 15

Einzelner Eingang „Discrete Input“	nur Lesen	1-bit	02
(analoge) Eingänge „Input Register“	nur Lesen	16-bits	04
(analoge) Ein-/Ausgänge „Holding Register“	Lesen & Schreiben	16-bits	03 / 06 / 16

3.3. Lesezugriff



Diese App ist im Standard-Lieferumfang enthalten.

Die Modbus-Schnittstelle ist folgendermaßen konfiguriert:

Geräteadresse	<i>IP-Adresse des EMS (z.B. 192.168.0.20)</i>
Port	502
Unit-ID	1
Function-Codes	03 (Read Holding Registers)
	04 (Read Input Registers)

Table 1. Parameter für Lesezugriff

Die Schnittstelle ermöglicht standardmäßig Zugriff auf die Kanäle der Komponente *_sum*. Der Zugriff auf weitere Komponenten wird projektspezifisch freigegeben – um z. B. ansteuerbare Stromspeichersysteme oder Ladesäulen über die Schnittstelle freizugeben.

3.4. Modbus-Tabelle

Die individuelle Modbus-Tabelle für Ihr System können Sie bequem über das Online-Monitoring als Excel-Datei wie folgt herunterladen:

3.4. Modbus-Tabelle

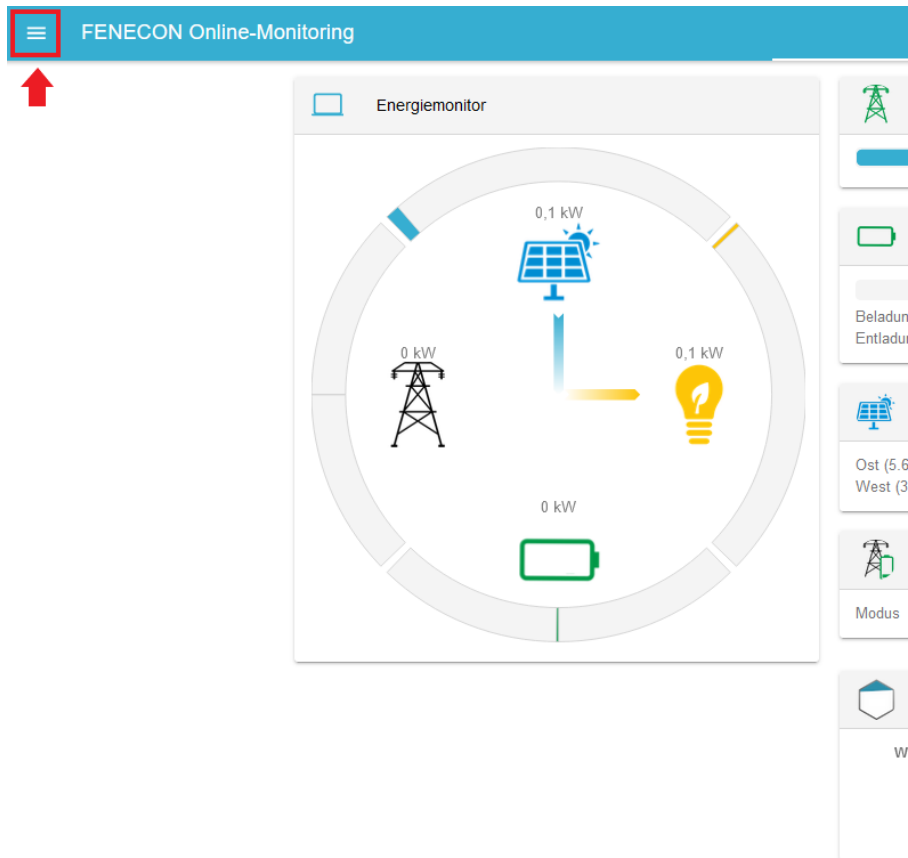


Abbildung 1. Reiter links oben im Online-Monitoring öffnen

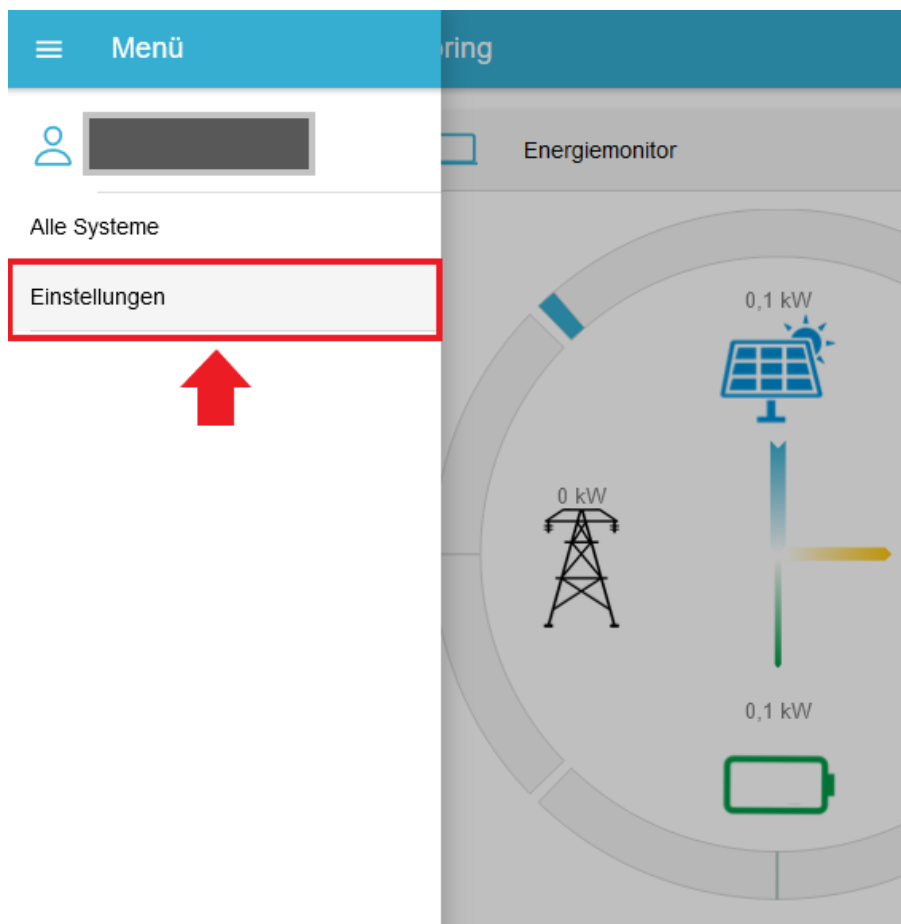


Abbildung 2. Reiter "Einstellungen" öffnen

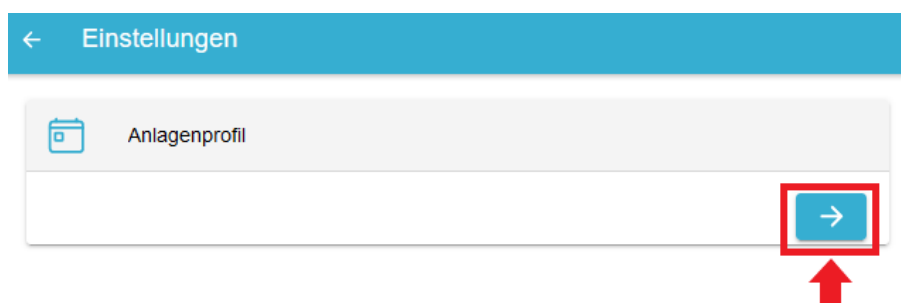


Abbildung 3. Anlagenprofil öffnen

3.4. Modbus-Tabelle

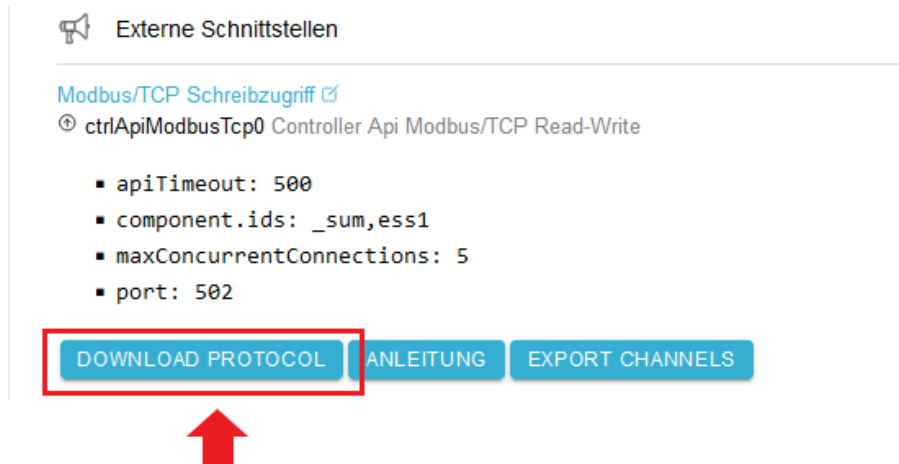


Abbildung 4. ctrlApiModbusTcp öffnen und auf "Download Protocol" klicken

Die wichtigsten Datenpunkte finden Sie auch hier in der Schnellübersicht:

Address (Adresse)	Name (Name)	Type (Typ)	Value/Description (Wert/Beschreibung)	Unit (Einheit)	Access (Zugang)
Header					
0	Hash of "OpenEMS"	uint16	0x6201		RO
1	Length of block "_meta"	uint16	199		RO
2	OpenEMS Version Major	uint16	2020		RO
3	OpenEMS Version Minor	uint16	26		RO
4	OpenEMS Version Patch	uint16	1		RO
5	Manufacturer	string16	FENECON GmbH		RO
21	Manufacturer Model	string16	OpenEMS		RO
37	Manufacturer Options	string16			RO
53	Manufacturer Version	string16			RO
69	Manufacturer Serial Number	string16			RO
85	Manufacturer EMS Serial Number	string16			RO
Sum					
200	Component-ID	string16	_sum		RO

216	Length of block "_sum"	uint16	300		RO
220	Hash of "OpenemsCompon ent"	uint16	0xb3dc		RO
221	Length of block "OpenemsCompon ent"	uint16	80		RO
222	_sum/State	enum16	0:Ok, 1:Info, 2:Warning, 3:Fault		RO
300	Hash of "Sum"	uint16	0x462b		RO
301	Length of block "Sum"	uint16	220		RO
302	_sum/EssSoc	uint16		Percent [%]	RO
303	_sum/EssActivePo wer	float32	AC-side power of Energy Storage System. Includes excess DC-PV production for hybrid inverters. Negative values for charge; positive for discharge	Watt [W]	RO
305	Reserved	float32			RO
307	Reserved	float32			RO
309	_sum/EssReactiveP ower	float32		VoltAmpereReactiv e [var]	RO
311	Reserved	float32			RO
313	Reserved	float32			RO
315	_sum/GridActivePo wer	float32	Grid exchange power. Negative values for sell-to- grid; positive for buy-from-grid	Watt [W]	RO
317	_sum/GridMinActiv ePower	float32		Watt [W]	RO
319	_sum/GridMaxActi vePower	float32		Watt [W]	RO
321	Reserved	float32			RO
323	Reserved	float32			RO
325	Reserved	float32			RO
327	_sum/ProductionA ctivePower	float32	Total production; always positive	Watt [W]	RO

3.4. Modbus-Tabelle

329	_sum/ProductionMaxActivePower	float32		Watt [W]	RO
331	_sum/ProductionAcActivePower	float32	Production from AC source	Watt [W]	RO
333	_sum/ProductionMaxAcActivePower	float32		Watt [W]	RO
335	Reserved	float32			RO
337	Reserved	float32			RO
339	_sum/ProductionDcActualPower	float32	Production from DC source	Watt [W]	RO
341	_sum/ProductionMaxDcActualPower	float32		Watt [W]	RO
343	_sum/ConsumptionActivePower	float32		Watt [W]	RO
345	_sum/ConsumptionMaxActivePower	float32		Watt [W]	RO
347	Reserved	float32			RO
349	Reserved	float32			RO
351	_sum/EssActiveChargeEnergy	float64		WattHours [Wh]	RO
355	_sum/EssActiveDischargeEnergy	float64		WattHours [Wh]	RO
359	_sum/GridBuyActiveEnergy	float64		WattHours [Wh]	RO
363	_sum/GridSellActiveEnergy	float64		WattHours [Wh]	RO
367	_sum/ProductionActiveEnergy	float64		WattHours [Wh]	RO
371	_sum/ProductionAcActiveEnergy	float64		WattHours [Wh]	RO
375	_sum/ProductionDcActiveEnergy	float64		WattHours [Wh]	RO
379	_sum/ConsumptionActiveEnergy	float64		WattHours [Wh]	RO
383	_sum/EssDcChargeEnergy	float64		WattHours [Wh]	RO
387	_sum/EssDcDischargeEnergy	float64		WattHours [Wh]	RO

391	_sum/EssActivePowerL1	float32	AC-side power of Energy Storage System on phase L1. Includes excess DC-PV production for hybrid inverters. Negative values for charge; positive for discharge	Watt [W]	RO
393	_sum/EssActivePowerL2	float32	AC-side power of Energy Storage System on phase L2. Includes excess DC-PV production for hybrid inverters. Negative values for charge; positive for discharge	Watt [W]	RO
395	_sum/EssActivePowerL3	float32	AC-side power of Energy Storage System on phase L3. Includes excess DC-PV production for hybrid inverters. Negative values for charge; positive for discharge	Watt [W]	RO
397	_sum/GridActivePowerL1	float32	Grid exchange power on phase L1. Negative values for sell-to-grid; positive for buy-from-grid	Watt [W]	RO
399	_sum/GridActivePowerL2	float32	Grid exchange power on phase L2. Negative values for sell-to-grid; positive for buy-from-grid	Watt [W]	RO
401	_sum/GridActivePowerL3	float32	Grid exchange power on phase L3. Negative values for sell-to-grid; positive for buy-from-grid	Watt [W]	RO

3.4. Modbus-Tabelle

403	_sum/ProductionAcActivePowerL1	float32	Production from AC source on phase L1	Watt [W]	RO
405	_sum/ProductionAcActivePowerL2	float32	Production from AC source on phase L2	Watt [W]	RO
407	_sum/ProductionAcActivePowerL3	float32	Production from AC source on phase L3	Watt [W]	RO
409	_sum/ConsumptionActivePowerL1	float32		Watt [W]	RO
411	_sum/ConsumptionActivePowerL2	float32		Watt [W]	RO
413	_sum/ConsumptionActivePowerL3	float32		Watt [W]	RO
415	_sum/EssDischarge Power	float32	Actual AC-side battery discharge power of Energy Storage System. Negative values for charge; positive for discharge	Watt [W]	RO
417	_sum/GridMode	enum16	1:On-Grid, 2:Off-Grid		RO
ess0					
500	Component-ID	string16	ess0		RO
516	Length of block "ess0"	uint16	580		RO
520	Hash of "OpenemsComponent"	uint16	0xb3dc		RO
521	Length of block "OpenemsComponent"	uint16	80		RO
522	ess0/State	enum16	0:Ok, 1:Info, 2:Warning, 3:Fault		RO
600	Hash of "SymmetricEss"	uint16	0x42ee		RO
601	Length of block "SymmetricEss"	uint16	100		RO
602	ess0/Soc	uint16		Percent [%]	RO
603	ess0/GridMode	enum16	1:On-Grid, 2:Off-Grid		RO
604	ess0/ActivePower	float32		Watt [W]	RO
700	Hash of "ManagedSymmetricEss"	uint16	0xa3ed		RO

701	Length of block "ManagedSymmetricEss"	uint16	100		RO
702	ess0/AllowedChargePower	float32		Watt [W]	RO
704	ess0/AllowedDischargePower	float32		Watt [W]	RO
706	ess0/SetActivePowerEquals	float32		Watt [W]	WO
708	ess0/SetReactivePowerEquals	float32		VoltAmpereReactive [var]	WO
710	ess0/SetActivePowerLessOrEquals	float32		Watt [W]	WO
712	ess0/SetReactivePowerLessOrEquals	float32		VoltAmpere [VA]	WO
714	ess0/SetActivePowerGreaterOrEquals	float32		Watt [W]	WO
716	ess0/SetReactivePowerGreaterOrEquals	float32		Watt [W]	WO
800	Hash of "EssSymmetric"	uint16	0x1352		RO
801	Length of block "EssSymmetric"	uint16	300		RO

3.5. Beispiel 1: Lesezugriff Batterieladezustand mit QModMaster

Im Folgenden soll der Lesezugriff auf den Ladezustand (SoC) der Batterie mittels des kostenlosen Tools *QModMaster* exemplarisch gezeigt werden.

Das Tool kann unter folgendem Link heruntergeladen werden:

Online: <https://sourceforge.net/projects/qmodmaster/>

Der Wert des Ladezustands ist wie folgt hinterlegt (s. oben):

Address	Name	Type	Value/Description	Unit	Access
302	_sum/EssSoc	uint16		Percent [%]	RO

Table 2. Registeradresse für den Ladezustand der Batterie

Standardmäßig wird in QModbusMaster die *Base Address* auf 1 gesetzt. Dieser Wert ist auf 0 zu ändern. Anderenfalls sind die Registeradressen aus dem Anlagenprofil um 1 verschoben.

3.5. Beispiel 1: Lesezugriff Batterieladezustand mit QModMaster

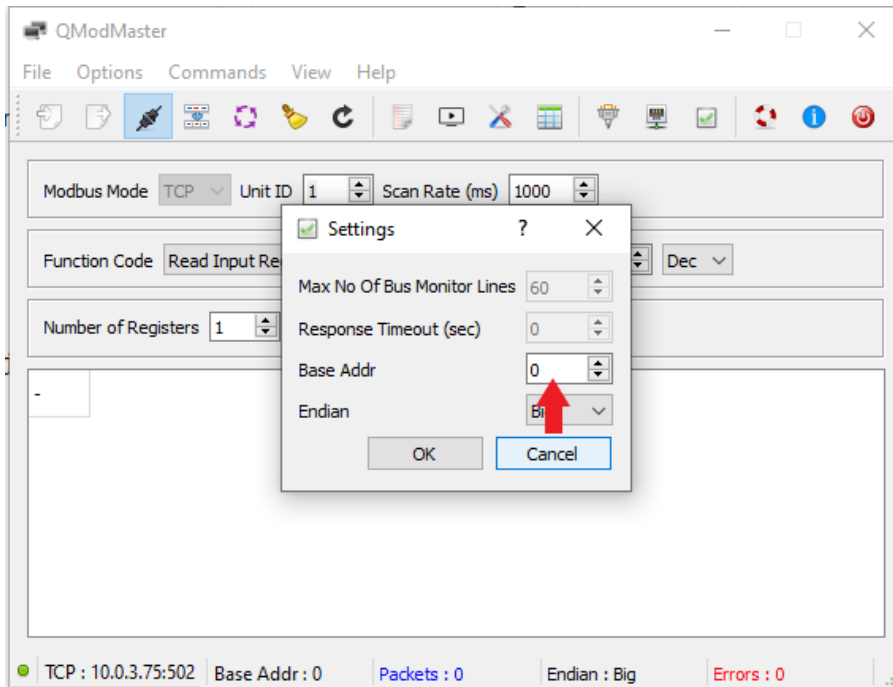


Abbildung 5. Einstellungen

Unter *Modbus TCP Settings* müssen *Slave IP* und *TCP Port* richtig konfiguriert sein.

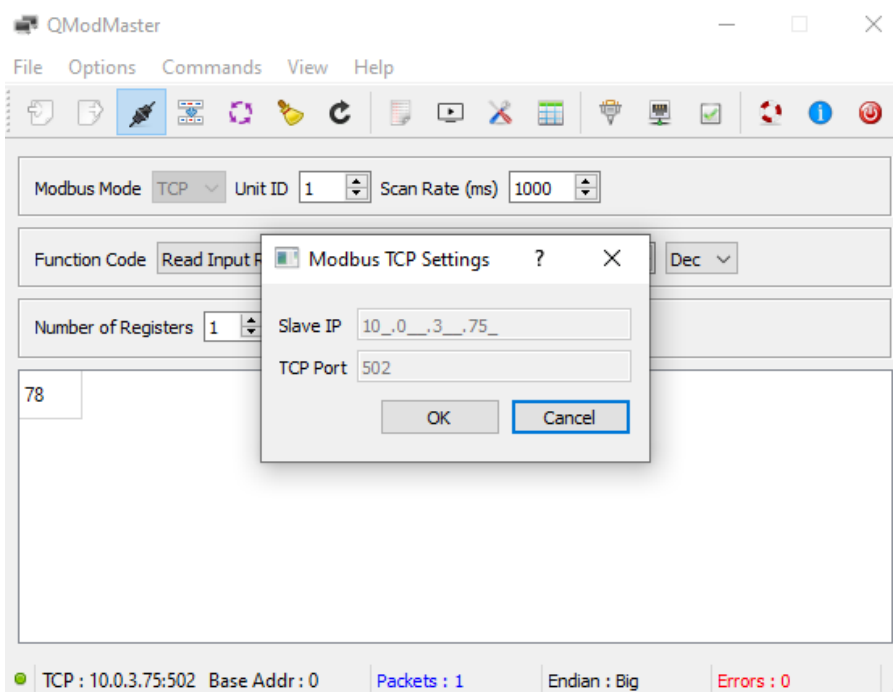


Abbildung 6. Modbus TCP Einstellungen

Da es sich um einen *unit16* handelt, muss ein 16-bit Wort, also ein Register, ausgelesen werden. Nach Setzen der Werte auf den Menüpunkt "Read/Write" klicken. Der gelesene Wert erscheint unten.

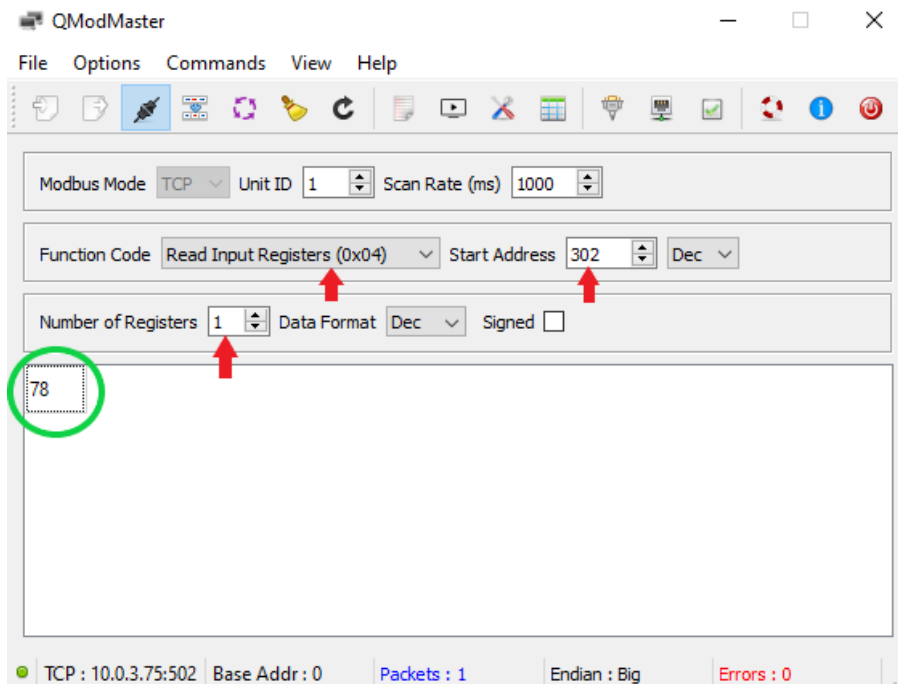


Abbildung 7. Wert lesen

Der Abgleich mit dem Online-Monitoring bestätigt die Korrektheit des gelesenen Wertes.

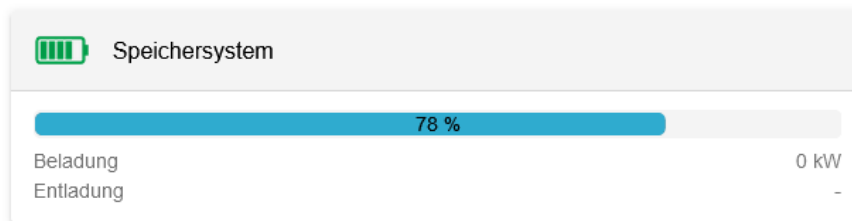


Abbildung 8. Vergleich mit Online-Monitoring

Die Durchführung anderer Leseoperationen erfolgt analog.

3.6. Schreibzugriff



Diese App ist nicht im Standard-Lieferumfang enthalten. Sie kann jedoch nachträglich jederzeit nachgerüstet werden.

Die Modbus-Schnittstelle ist folgendermaßen konfiguriert:

Geräteadresse	IP-Adresse des EMS (z.B. 192.168.0.20)
Port	502
Unit-ID	1
Function-Codes	03 (Read Holding Registers)
	04 (Read Input Registers)
	06 (Write Single Holding Register)
	16 (Write Multiple Holding Registers)

3.7. Beispiel 2: Schreibzugriff auf EssActivePower mit QModMaster

Table 3. Parameter für Schreibzugriff

3.7. Beispiel 2: Schreibzugriff auf EssActivePower mit QModMaster

Im Folgenden soll der Schreibzugriff für das Setzen der *EssActivePower* mittels des kostenlosen Tools *QModMaster* exemplarisch gezeigt werden. Hierdurch kann die Funktion des *Controller Fix Active Power Symmetric* simuliert werden.

Der Wert ist wie folgt hinterlegt (s. oben):

Address	Name	Type	Value/Description	Unit	Access
706	ess0/SetActivePowerEquals	float32		Watt [W]	WO

Table 4. Registeradresse für das Setzen der *EssActivePower* des Speichers

	Zusätzlich zur Überprüfung der <i>Base Address</i> auf 0 muss sichergestellt werden, dass unter <i>Endian</i> die Einstellung <i>Big</i> ausgewählt ist.
---	--

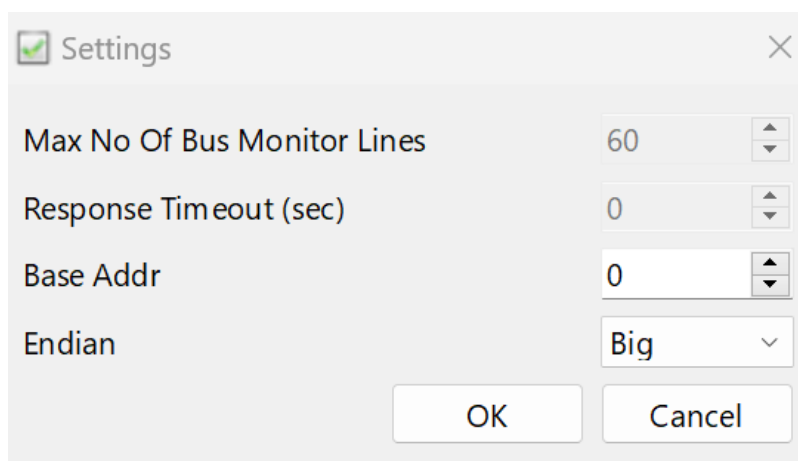


Abbildung 9. Einstellungen

Da es sich um einen *float32* handelt, müssen zwei 16-bit Wörter, also zwei Register, geschrieben werden. In diesem Beispiel soll der Speicher mit 4000 (4E+03) Watt entladen werden. Der Wert kann direkt als Dezimalzahl in das Register eingegeben werden, wobei das Data Format *Float* zu wählen ist. Nach Setzen des Wertes auf den Menüpunkt "Read/Write" klicken, um die Schreiboperation durchzuführen.

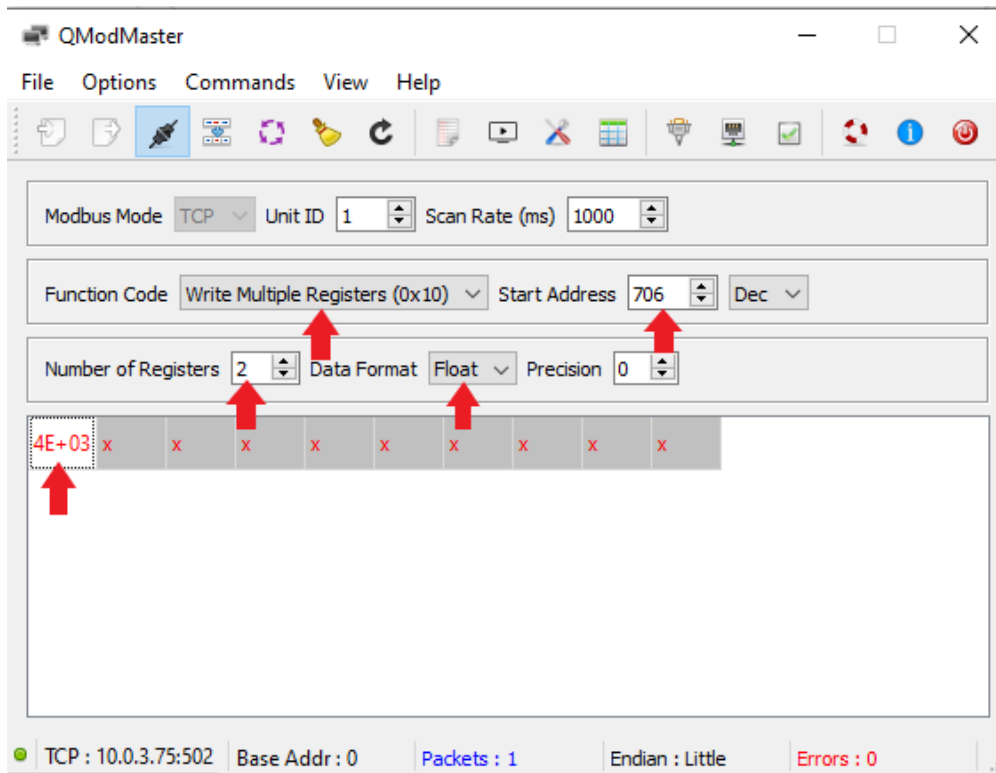


Abbildung 10. Wert schreiben

Der Abgleich mit dem Online-Monitoring bestätigt die Korrektheit des geschriebenen Wertes.

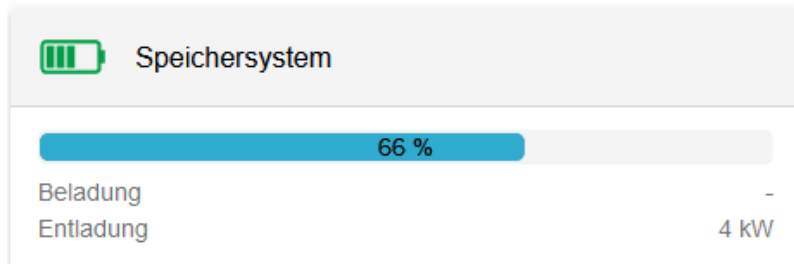


Abbildung 11. Vergleich mit Online-Monitoring



Die Schreiboperation wird nur für die Dauer des *Api-Timeout* durchgeführt. Diese beträgt in der Standardkonfiguration 60 Sekunden, kann aber durch den Service angepasst werden.

Die Durchführung anderer Schreiboperationen erfolgt analog.

4. REST/JSON - Lese-/Schreibzugriff

Diese Anleitung dient der Beschreibung des Lese-/Schreibzugriffs auf ein FENECON Stromspeichersystem mittels REST/JSON API. Anschließend wird die Funktionsweise der Schnittstelle erklärt.

4.1. Voraussetzungen

Das auf das Stromspeichersystem zugreifende Gerät (z.B. Notebook/PC) muss direkten Zugriff auf die IP-Adresse des FEMS haben – also z. B. im gleichen physischen Netzwerk angeschlossen sein.

4.2. Grundlagen REST/JSON

Die REST/JSON Schnittstelle ermöglicht den Zugriff auf das FEMS im lokalen Netzwerk über eine an REST angelehnte Schnittstelle.

4.3. Lesezugriff

Diese App stellt eine an REST angelehnte Schnittstelle zur Verfügung, mit der Datenpunkte im System ausgelesen werden können.



Diese App ist im Standard-Lieferumfang des FEMS 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 FEMS hat das Passwort "user"
- **<IP>** ist die IP-Adresse des FEMS
- **80** ist der Port für die REST/JSON-API (optional)

Wenn Ihr FEMS 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. Eine Abfrage direkt über den Browser (ohne Erweiterung) ist folglich nicht möglich. In diesem Fall erscheint eine Fehlermeldung, vgl. Abbildung [Fehlermeldung bei Simple Authentication Requests](#). Für eine Abfrage im Browser, nutzen Sie bitte die Erweiterung Talend API Tester wie [hier](#) beschrieben.

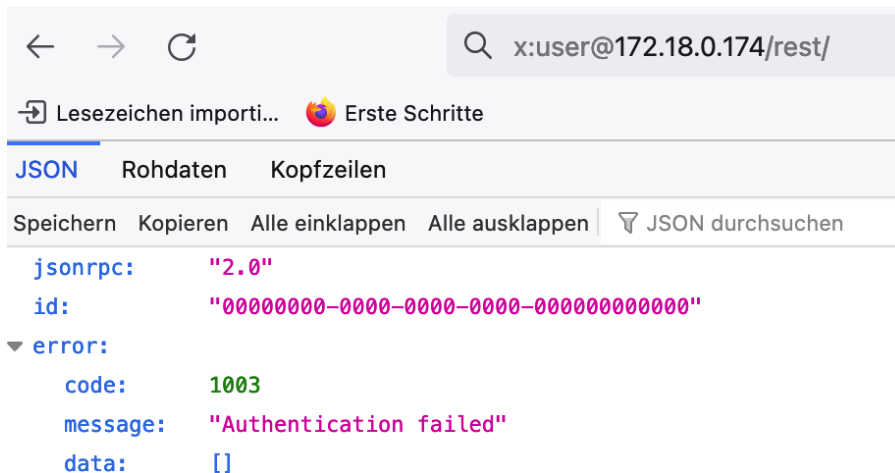


Abbildung 12. Fehlermeldung bei Simple Authentication Requests

4.3.1. /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

4.3.2. Datenpunkte

Die folgenden Datenpunkte können ausgelesen werden:

Datenpunkt	Beschreibung	Einheit
<code>_sum/State</code>	Zustand des Systems (0: Ok, 1:Info, 2:Warning, 3:Fault)	
<code>_sum/EssSoc</code>	Ladezustand des Speichers	Prozent [%]
<code>_sum/EssActivePower</code>	Wirkleistung des Speichers	Watt [W]
<code>_sum/GridActivePower</code>	Wirkleistung am Netzanschlusspunkt	Watt [W]
<code>_sum/ProductionActivePower</code>	Wirkleistung Erzeuger	Watt [W]
<code>_sum/ProductionAcActivePower</code>	Wirkleistung AC Erzeuger	Watt [W]
<code>_sum/ProductionDcActualPower</code>	Wirkleistung DC Erzeuger	Watt [W]
<code>_sum/ConsumptionActivePower</code>	Wirkleistung Verbraucher	Watt [W]
<code>_sum/EssActiveChargeEnergy</code>	Energie Speicherbeladung	WattHours [Wh]
<code>_sum/EssActiveDischargeEnergy</code>	Energie Speicherentladung	WattHours [Wh]
<code>_sum/GridBuyActiveEnergy</code>	Energie Netzbezug	WattHours [Wh]
<code>_sum/GridSellActiveEnergy</code>	Energie Netzeinspeisung	WattHours [Wh]

4.3. Lesezugriff

<code>_sum/ProductionActiveEnergy</code>	Energie Erzeugung	WattHours [Wh]
<code>_sum/ProductionAcActiveEnergy</code>	Energie AC Erzeugung	WattHours [Wh]
<code>_sum/ProductionDcActiveEnergy</code>	Energie DC Erzeugung	WattHours [Wh]
<code>_sum/ConsumptionActiveEnergy</code>	Energie Verbraucher	WattHours [Wh]
<code>_sum/EssActivePowerL1</code>	Wirkleistung Phase 1 Speicher	Watt [W]
<code>_sum/EssActivePowerL2</code>	Wirkleistung Phase 2 Speicher	Watt [W]
<code>_sum/EssActivePowerL3</code>	Wirkleistung Phase 3 Speicher	Watt [W]
<code>_sum/GridActivePowerL1</code>	Wirkleistung Phase 1 Netz	Watt [W]
<code>_sum/GridActivePowerL2</code>	Wirkleistung Phase 2 Netz	Watt [W]
<code>_sum/GridActivePowerL3</code>	Wirkleistung Phase 3 Netz	Watt [W]
<code>_sum/ProductionAcActivePowerL1</code>	Wirkleistung Phase 1 Erzeuger	Watt [W]
<code>_sum/ProductionAcActivePowerL2</code>	Wirkleistung Phase 2 Erzeuger	Watt [W]
<code>_sum/ProductionAcActivePowerL3</code>	Wirkleistung Phase 3 Erzeuger	Watt [W]
<code>_sum/ConsumptionActivePowerL1</code>	Wirkleistung Phase 1 Verbraucher	Watt [W]
<code>_sum/ConsumptionActivePowerL2</code>	Wirkleistung Phase 2 Verbraucher	Watt [W]
<code>_sum/ConsumptionActivePowerL3</code>	Wirkleistung Phase 3 Verbraucher	Watt [W]

4.3.3. 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": "R0", "text": "", "unit": "%", "value": 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": "R0", "text": "", "unit": "%", "value": 99}
```

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

4.3.4. 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:

4.3. Lesezugriff

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

Ausgabe:

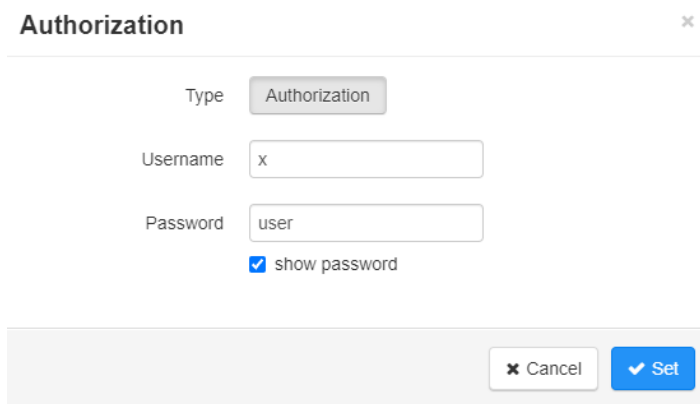
```
{ "address": "_sum/EssSoc", "type": "INTEGER", "accessMode": "RO", "text": "", "unit": "%", "value": 99 }
```

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

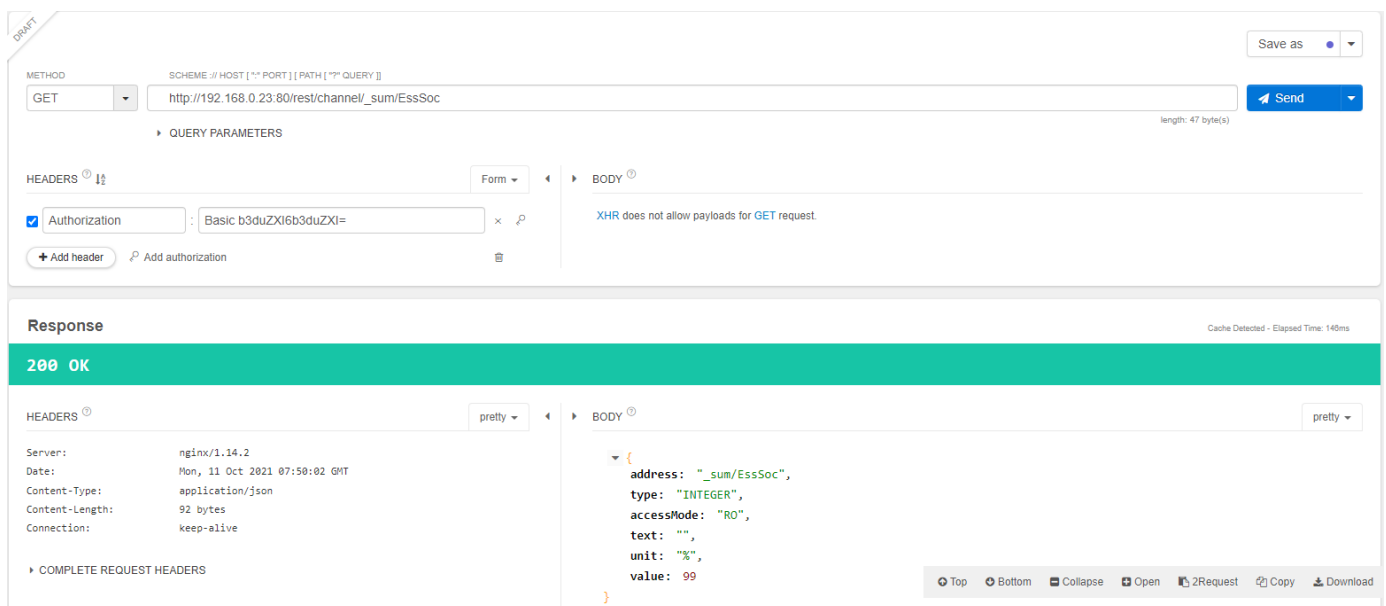
4.3.5. Beispiel 3 - Abfrage des Ladezustands: Talend API Tester

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.



The screenshot shows the Talend API Tester interface. The top section displays the request details: METHOD is GET, URL is `http://192.168.0.23:80/rest/channel/_sum/EssSoc`, and the HEADERS section shows an 'Authorization' header with the value 'Basic b3duZXI6b3duZXI='. The bottom section shows the response: Status is 200 OK, and the BODY is a JSON object: `{ "address": "_sum/EssSoc", "type": "INTEGER", "accessMode": "RO", "text": "", "unit": "%", "value": 99 }`.

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

4.4. Schreibzugriff

Diese App 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 FEMS 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 Nutzernamen kann wie beim Lesezugriff beliebig gewählt werden.

Sämtliche Schreibzugriffe müssen als **POST**-Requests gesendet werden.

4.4.1. Timeout

Diese App 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.

4.4.2. /channel Endpunkt

Über den Endpunkt **/channel** wird 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

4.4.3. Datenpunkte

Die folgenden Datenpunkte können beschrieben werden:

Datenpunkt	Beschreibung	Einheit
<code>ess0/SetActivePowerEquals</code>	Wirkleistungsvorgabe	Watt [W]
<code>ess0/SetReactivePowerEquals</code>	Blindleistungsvorgabe	VoltAmpereReactive [VAR]
<code>ess0/SetActivePowerLessOrEquals</code>	(Maximale) Wirkleistungsvorgabe	Watt [W]
<code>ess0/SetReactivePowerLessOrEquals</code>	(Maximale) Blindleistungsvorgabe	VoltAmpereReactive [VAR]
<code>ess0/SetActivePowerGreaterOrEquals</code>	(Minimale) Wirkleistungsvorgabe	Watt [W]

4.4. Schreibzugriff

ess0/SetReactivePowerGreaterOrEquals	(Minimale) Wirkleistungsvorgabe	VoltAmpereReactive [VAR]
--------------------------------------	---------------------------------	--------------------------



Mehr Informationen zum Channel `SetActivePowerEquals` und anderen Channels zur Leistungsvorgabe finden Sie im [Glossar](#)

4.4.4. 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'

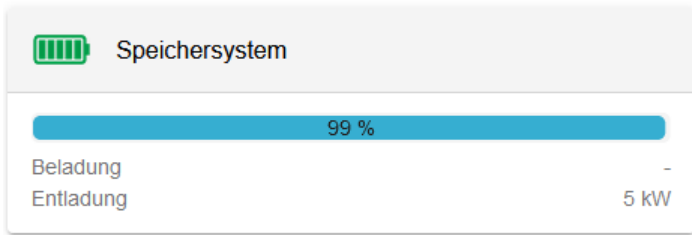
user = 'x'
password = 'owner'

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

data = {"value": 5000}

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

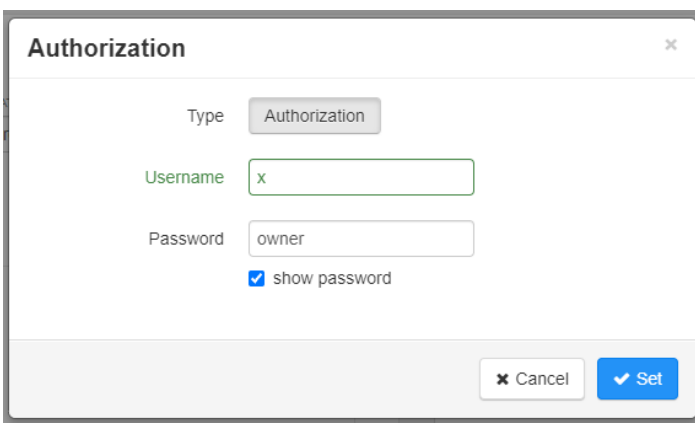
Die korrekte Durchführung des Requests kann über einen anschließenden **GET**-Request oder das Online-Monitoring (s. unten) überprüft werden.



4.4.5. 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:



Authorization

Type: Authorization

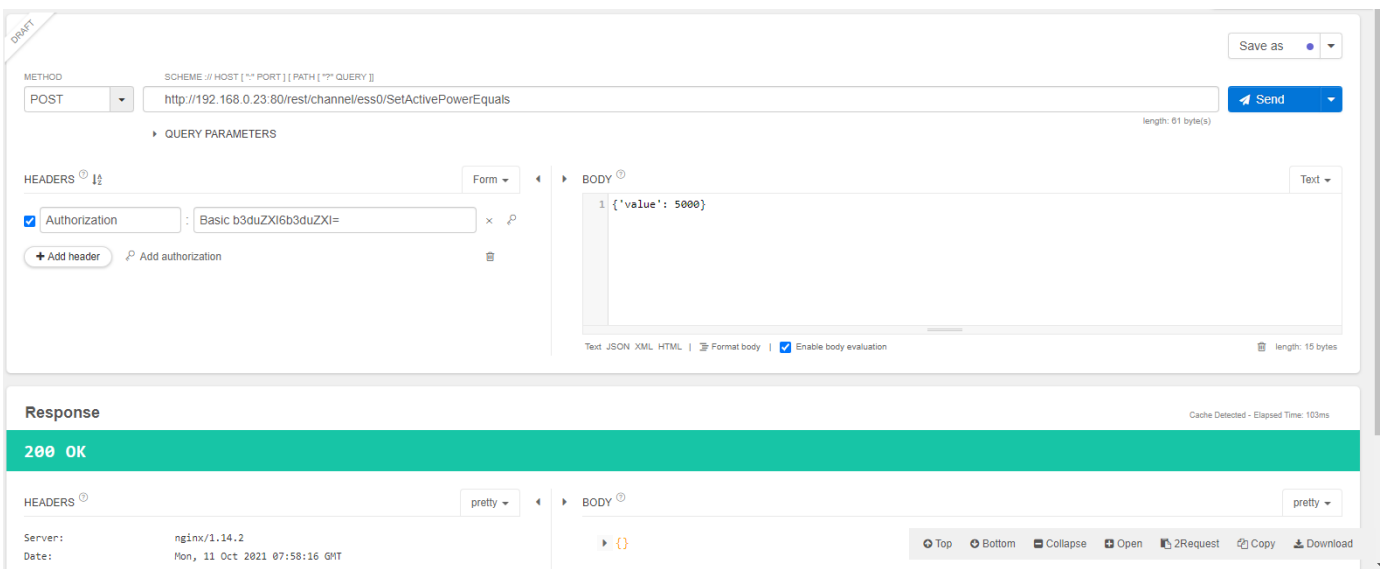
Username: x

Password: owner

☒ show password

Cancel Set

Anschließend kann der **POST**-Request ausgeführt werden.



Request

METHOD: POST

SCHEME // HOST [: PORT] [PATH [? QUERY]]

http://192.168.0.23:80/rest/channel/ess0/SetActivePowerEquals

length: 61 byte(s)

Send

HEADERS

☒ Authorization: Basic b3duZXI6b3duZXI=

Body: 1 { 'value': 5000 }

length: 15 bytes

Response

200 OK

Cache Detected - Elapsed Time: 103ms

HEADERS

Server: nginx/1.14.2

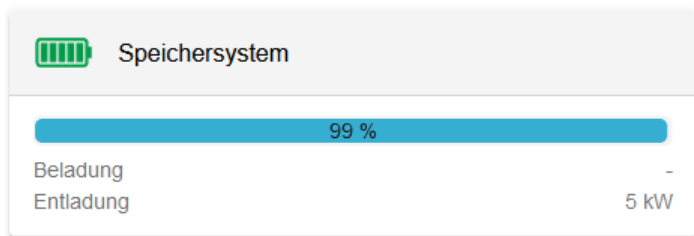
Date: Mon, 11 Oct 2021 07:58:16 GMT

Body: {}

Top Bottom Collapse Open 2Request Copy Download

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

4.4. Schreibzugriff



5. Kontakt

Für Unterstützung wenden Sie sich bitte an:

FENECON GmbH

Brunnwiesenstr. 4

94469 Deggendorf

Telefon Service: 0991-648800-33

E-Mail Service: service@fenecon.de