

IOT-GW30_4G

Quick Start Guide - SMS via IOT-GW30_4G

Abstract:

This document describes how to send and receive SMS via IOT-GW30_4G. This is achieved via Node-RED which controls a so-called modem manager cli (mmcli) running on the system.

Hardware reference

No.	Component name	Article No.	Hardware / Firmware version
1	IOT-GW30_4G	2682630000	Revision 6 / 1.16.0
2			
3			

Software reference

No.	Software name	Article No.	Software version
1	Node-RED	-	v. 1.1.2
2	mmcli	-	v. 1.10.8
3			

File reference

No.	Name	Description	Version
1	QSG0045-IOT-GW30 Quick Start for SMS via IOT-GW30.zip	The file contains a Node-RED flow related to this quick start guide	-
2			
3			

Contact

Weidmüller Interface GmbH & Co. KG
Klingenbergstraße 26
32758 Detmold, Germany
www.weidmueller.com

For any further support please contact your local sales representative:
<https://www.weidmueller.com/countries>

Content

1	Warning and Disclaimer.....	4
2	Preparation.....	5
2.1	Prerequisites.....	5
2.2	Software Setup.....	5
3	Application.....	6
3.1	Init	6
3.2	Send SMS	6
3.3	List SMS.....	8
3.4	Read SMS	9
3.5	Delete SMS	10
3.6	Additional hints	10

1 Warning and Disclaimer

Warning

Controls may fail in unsafe operating conditions, causing uncontrolled operation of the controlled devices. Such hazardous events can result in death and / or serious injury and / or property damage. Therefore, there must be safety equipment provided / electrical safety design or other redundant safety features that are independent from the automation system.

Disclaimer

This Application Note / Quick Start Guide / Example Program does not relieve you of the obligation to handle it safely during use, installation, operation and maintenance. Each user is responsible for the correct operation of his control system. By using this Application Note / Quick Start Guide / Example Program prepared by Weidmüller, you accept that Weidmüller cannot be held liable for any damage to property and / or personal injury that may occur because of the use.

Note

The given descriptions and examples do not represent any customer-specific solutions, they are simply intended to help for typical tasks. The user is responsible for the proper operation of the described products. Application notes / Quick Start Guides / Example Programs are not binding and do not claim to be complete in terms of configuration as well as any contingencies. By using this Application Note / Quick Start Guide / Example Program, you acknowledge that we cannot be held liable for any damages beyond the described liability regime. We reserve the right to make changes to this application note / quick start guide / example at any time without notice. In case of discrepancies between the proposals Application Notes / Quick Start Guides / Program Examples and other Weidmüller publications, like manuals, such contents have always more priority to the examples. We assume no liability for the information contained in this document. Our liability, for whatever legal reason, for damages caused using the examples, instructions, programs, project planning and performance data, etc. described in this Application Note / Quick Start Guide / Example is excluded.

Security notes

In order to protect equipment, systems, machines and networks against cyber threats, it is necessary to implement (and maintain) a complete state-of-the-art industrial security concept. The customer is responsible for preventing unauthorized access to his equipment, systems, machines and networks. Systems, machines and components should only be connected to the corporate network or the Internet if necessary and appropriate safeguards (such as firewalls and network segmentation) have been taken.

2 Preparation

2.1 Prerequisites

SIM card:



SMS functionality must be available and enabled on the SIM card used.

IOT-GW30_4G:



Suitable antennas and the SIM card must be installed before the IOT-GW30_4G is started.

2.2 Software Setup

Make sure the cellular settings are configured and the state is "online".

Cellular settings	
PIN	<input type="text" value=""/>
Provider APN	<input type="text" value="internet.telekom (example)"/>
Username	<input type="text" value=""/>
Password	<input type="password" value=""/>
<input type="button" value="Apply"/>	
<hr/>	
State	<input type="text" value="Online"/>
Signal strength	<input type="text" value="100"/>
Mobile operator name	<input type="text" value=""/>
Mobile Network Type	<input type="text" value="LTE"/>

Open the Node-Red Editor and deploy the attached application. No additional open-source nodes are required.

3 Application

This application example is based on a so-called “modem manager cli” (mmcli) daemon running on the system. This application includes only a subset of all available commands, that are required to send, receive, and delete a SMS.

3.1 Init

On startup, all available modems are listed. Since the IOT-GW30_4G has only one modem available, the modem id should always be “0”. If the modem id cannot be extracted from the command response, there is most likely an issue with the SIM card or the cellular settings.

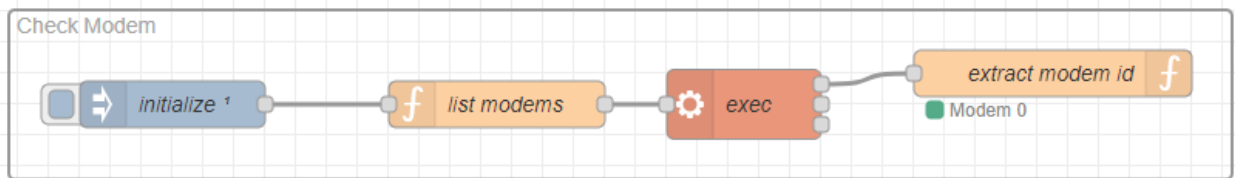


Figure 1 Check modem

3.2 Send SMS

Sending a SMS via mmcli requires multiple steps.

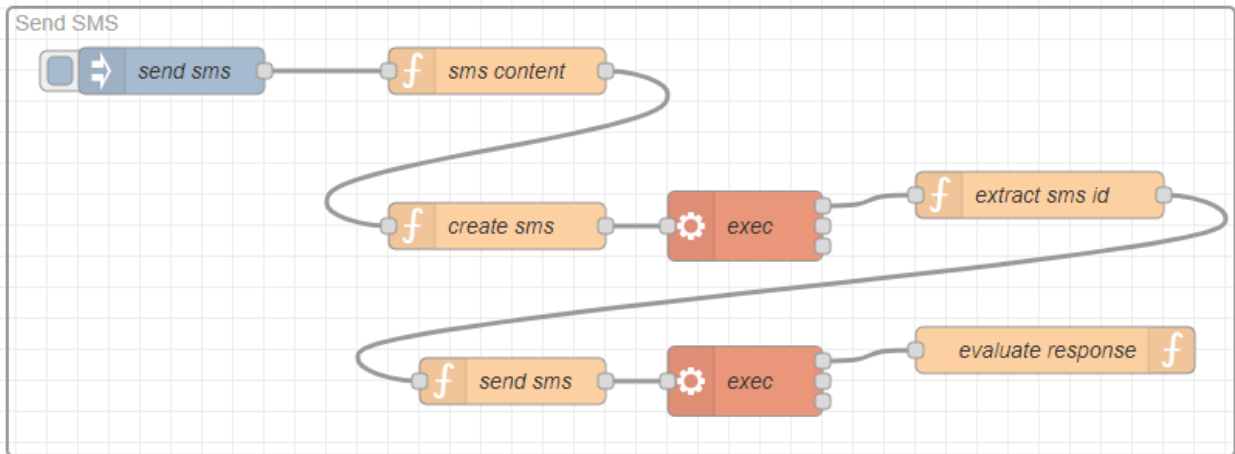


Figure 2 Send SMS

First, the phone number of the recipient as well as the SMS text must be changed (Figure 3).

```

1 msg.number = '+4912345678910';
2
3 //Example for a multiline SMS using template strings
4 msg.text = `Hello world,
5
6 this is an example for a multiline SMS.
7
8
9 Best regards
10
11 IoT-GW30-4G
12 `;
13
14 return msg;

```

Figure 3 SMS content

The content is handed over to the “create sms”-function (Figure 4). Some characters like for example German “Umlauts” (äöü), or Chinese symbols (你翻譯的) are not supported by the mmcli daemon, so regular expressions are used to remove untested characters from the text. This ensures that the message can be processed. *Read comments in the code for details.* The mmcli command creates a SMS containing additional meta data which is required to send the SMS.

```

1 /*
2 ** Some chars cannot be used via mmcli and must be removed. Otherwise the SMS cannot be send.
3 ** Supported (and tested) unicode character:
4 ** !"#%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMN O PQRSTU VWXYZ[\]^_`abcdefghijklmnopqrstuvwxyz{|}~
5 ** additional chars: U+000A (new line), U+0020 (space)
6 ** unicode char U+0027 (single quotation marks) is replaced by U+0022 (double quotation marks)
7 ^
8
9 msg.text = msg.text.replace(/[\\u0000-\\u0009\\u000B-\\u001F\\u007F-\\u00FF\\u007F-\\uFFFF]/g, ''); //remove untested characters
10 msg.text = msg.text.replace(/\\u0027/g, '\\u0022'); //replace all ' by "
11
12 msg.payload = `mmcli -m ${global.get('cellular.modemId')} --messaging-create-sms="text='${msg.text}',number='${msg.number}'`
13
14 return msg;

```

Figure 4 Create SMS

If the creation is successful there is a response containing an id which is assigned to this SMS. Finally, this id is used to send the SMS.

3.3 List SMS

Another command is used to get a list of all SMS on the device. The response includes SMS of the type “received”, “sent”, “unknown” and their corresponding ids.

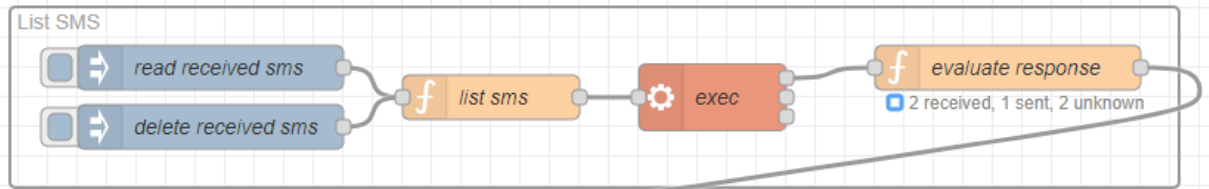


Figure 5 List SMS

The SMS ids become ordered and assigned to the different SMS types and stored to the global context. On this basis, various scenarios such as "read the last received SMS" or "delete all unknown SMS" can be implemented.

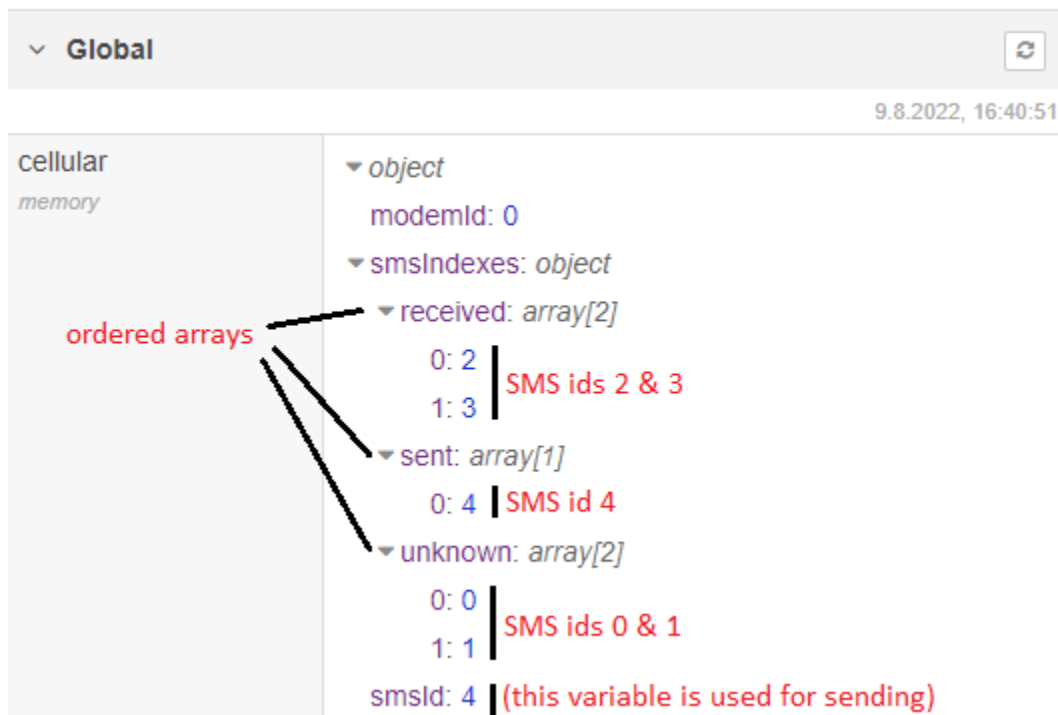


Figure 6 Cellular context variable

This document exemplary describes how to “read all received SMS” (Chapter 3.4) and how to “delete all received SMS” (Chapter 3.5) on this basis.

3.4 Read SMS

There are as many mmcli read commands as received SMS based on the Chapter 3.3.

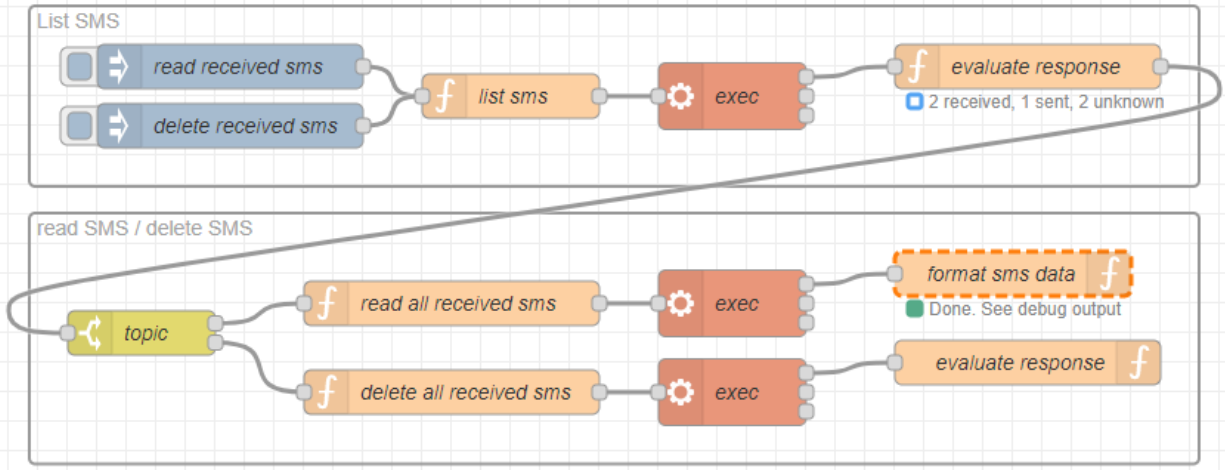


Figure 7 Read SMS

The response is a human readable string and is formatted to JavaScript object in order to enable easy further processing.



Figure 8 Format SMS

3.5 Delete SMS

There are as many mmcli delete commands as received SMS based on the Chapter 3.3.

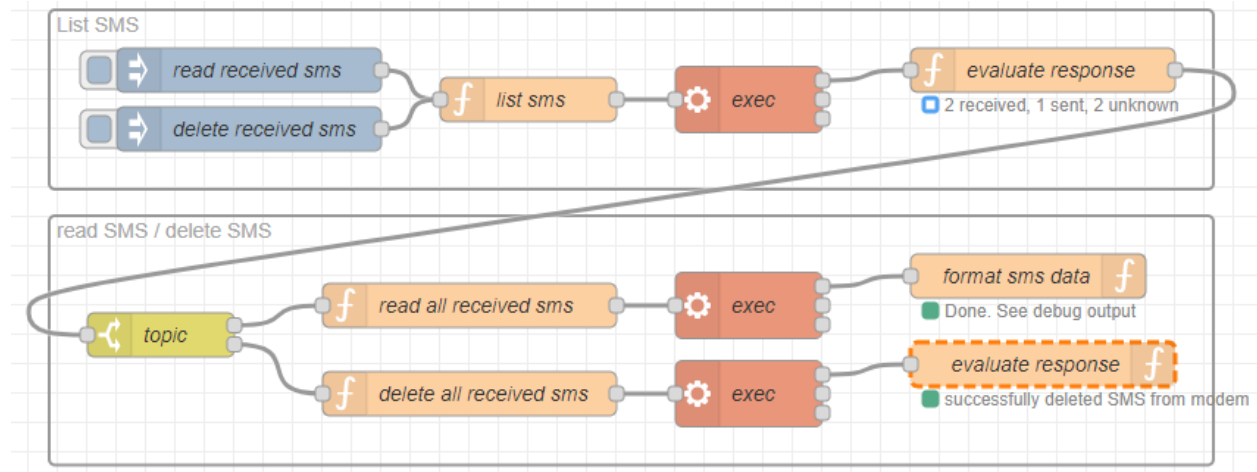


Figure 9 Delete SMS

3.6 Additional hints

- SMS of the type “sent” and “unknown” are deleted after a reboot.
- Example for a SMS of the type “unknown”: SMS is created but not sent.
- Received SMS are saved until they are deleted manually.
- The behavior of having hundreds of SMS is not tested. So, it is recommended to clear received SMS somehow in the program.
- Received SMS cannot be read event based. Therefore, the SMS list must be checked for changes cyclically in order to detect newly received SMS.
- The mmcli daemon responses are in a string format. That’s why some regular expression and other string operations are required to evaluate the responses from the “exec”-nodes.