Panasonic

PAN9420

Firmware Version 1.1.x.x

Command Specification

Rev. 1.2

Wireless Modules

By purchase of any of the products described in this document the customer accepts the document's validity and declares their agreement and understanding of its contents and recommendations. Panasonic Industrial Devices Europe GmbH (Panasonic) reserves the right to make changes as required at any time without notification.

This Command Specifcation does not lodge the claim to be complete and free of mistakes.

Table of Contents

1 About This Document

1.1 Purpose and Audience

This Command Specification provides details on the communication via UART and HTTP/JSON for the various features of the PAN9420 module. It is intended for software engineers. The product is referred to as “the PAN9420” or “the module” within this document.

1.2 Revision History

Revision	Date	Modifications/Remarks
0.1	05.07.2017	1st preliminary version.
1.0	04.08.2017	- Layout changes - Reviewed commands - Spell checking
1.1	03.04.2018	- Added fwu mode, server and resource commands - Added MQTT - Added general status codes - Added UDP status codes - Added info event messages - Updated binary UART read and write examples - Updated power save mode command description and examples - Added range and unit for the RSSI command
1.2	20.04.2018	- Editorial changes - Added info message structure to commands - Updated description of the commands “get wlan ssid”, “set wlan cfg”, “get wlan list” and “get wlan bssid”

1.3 Use of Symbols

Symbol	Description
	**Note** Indicates important information for the proper use of the product. Non-observance can lead to errors.
	**Attention** Indicates important notes that, if not observed, can put the product's functionality at risk.
	**Tip** Indicates useful information designed to facilitate working with the software.
⇨ [chapter number] [chapter title]	**Cross reference** Indicates cross references within the document. Example: Description of the symbols used in this document ⇨ 1.3 Use of Symbols.
`This font`	**File names, messages, user input** Indicates file names or messages and information displayed on the screen or to be selected or entered by the user. **Examples:** Pan9420.c contains the actual module initialization. The message Failed to save your data is displayed. Enter the value Product 123.

1.4 Related Documents

Please refer to the Panasonic website for related documents ⇨ 6.2 Product Information.

2 Overview

The PAN9420 is a 2.4 GHz 802.11 b/g/n embedded Wi-Fi module with integrated stack and API that minimizes firmware development and includes a full security suite. The module is specifically designed for highly integrated and cost-effective applications. The module includes a fully shielded case, integrated crystal oscillators, and a chip antenna.

The module combines a high-performance CPU, high-sensitivity wireless radio, baseband processor, medium access controller, encryption unit, boot ROM with patching capability, internal SRAM, and in-system programmable flash memory. The module's integrated memory is available to the application for storing web content such as HTML pages or image data.

Parallel support of access point and infrastructure mode allows easy setup of simultaneous Wi-Fi connections from the module to smart devices and home network routers.

The pre-programmed Wi-Fi SoC firmware enables client (STA) and micro access point (µAP) applications. With the transparent mode, raw data can be sent from the UART to the air interface to smart devices, web servers, or PC applications.

Please refer to the Panasonic website for more information ⇨ 6.2 Product Information.

3 Commands

3.1 General

There are four types of commands:

- Request
- Response
- Error
- Info

The format is ASCII-based and ending with “CR-LF”.

„Request“ Command

Structure

Type	Module	Variable	Parameters

Definition	
Type:	The command type, which is either “get” or “set”.
Module:	The functional software module.
Variable:	The subset of the selected software module.
Parameters:	The set of parameters depending on the module and variable.

Each request triggers a „Response“ of the following structure:

„Response“ Command

Structure

Type	Module	Variable	Return Code	Parameters

Definition	
Type:	The command type, which is either “get” or “set”.
Module:	The functional software module.
Variable:	The subset of the selected software module.
Return Code:	The return code of the execution of the request ⇨ 5.1 Telegram Return Code.
Parameters:	The set of parameters depending on the module and variable.

„Error“ Message

Structure

„Error“	Error Code

Definition	
Error Code:	The error code to a request command ⇨ 5.1 Telegram Return Code.

„Info“ Message

Structure

“info”	Module	Variable	Parameter

Definition	
Module:	The functional software module.
Variable:	The subset of the selected software module.
Parameters:	The set of parameters depending on the module and variable.

3.2 Command Templates and Examples

Templates	
HTTP/JSON Request	`["Command","Module","Variable"]`
HTTP/JSON Response	`["Command","Module","Variable","Return-Code","Parameter"]`
CMD-UART Request	`Command Module Variable\x0d\x0a`
CMD-UART Response	`Command Module Variable Return-Code Parameter\x0d\x0a`

Template for the HTTP/JSON Interface

HTTP/JSON Request	`{"FileName":"cfgpars.json","CmdArr"[["Command","Module","Variable"]]}`
HTTP/JSON Response	`{"FileName":"cfgpars.json","CmdArr"[["Command","Module","Variable","Return-Code","Parameter"]]}`

Example of JSON-Request (with HTTP-Header)

```
POST /ajax/cfgpars.json?rauth=0x00 HTTP/1.1
Host: 192.168.1.1
Connection: keep-alive
Content-Length: 221
Accept: text/html, */*; q=0.01
Origin: http://192.168.1.1
X-Requested-With: XMLHttpRequest
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/52.0.2743.116 Safari/537.36
Content-Type: application/json; charset=UTF-8
Referer: http://192.168.1.1/webdesktop/index.html
Accept-Encoding: gzip, deflate
Accept-Language: de-DE,de;q=0.8,en-US;q=0.6,en;q=0.4

{“FileName”:”cfgpars.json”,”CmdArr”:[[“get”,”system”,”version”,””],[“get”,”system”,”firm
ware”,””],[“get”,”system”,”serialnum”,””],[“get”,”system”,”macaddr”,””],[“get”,”system”,
”hwrev”,””]]}
```

Example JSON-Response (with HTTP-Header)

```
HTTP/1.1 200 OK
Connection: keep-alive
Content-Type: application/json
Keep-Alive: timeout=8, max=65535
Transfer-Encoding: chunked

{“FileName”:”cfgpars.json”,”CmdArr”:[[“get”,”system”,”version”,”0”,”1.0”],[“get”,”system
”,”firmware”,”0”,”1.1.0.0”],[“get”,”system”,”serialnum”,”0”,”00000001”],[“get”,”system”,
”macaddr”,”0”,”00:01:02:03:04:05”],[“get”,”system”,”hwrev”,”0”,”03”]]}
```

4 Software Modules

4.1 WLAN

4.1.1 Infrastructure SSID (variable: ssid)

Definition		
Command Option:	Get	
Parameter 1:	**<interface>** **Default** **Range**	Identifier to select the WLAN interface. - [ap, sta] ap: Access-Point sta: Station
Parameter 2:	**<ssid>** **Default value:** **Range:**	SSID of a WLAN interface. **-** String with ASCII printable characters with a length of (1-32). SSIDs containing spaces will be put in quotes, quotes in an SSID will be marked with an additional quote in front of it and non-ASCII characters will be displayed with its UTF-8 code and a “\u”-escape sequence in the front.
Description:	Returns the network SSID of a WLAN interface as an ASCII-encoded string with a maximum length of 32 characters.	
GET Rights:	0x00	

Examples	
http/JSON Get-Request	`["get","wlan","ssid","sta"]`
http/JSON Get-Response	`["get","wlan","ssid,","0","sta","testnetwork"]`
CMD-UART Get-Request	`get wlan ssid sta\x0d\x0a`
CMD-UART Get-Response	`get wlan ssid 0 sta testnetwork\x0d\x0a`

4.1.2 Infrastructure State (variable: state)

Definition		
Command option:	Get / Set	
Parameter 1:	**<interface>** **Default** **Range**	Identifier to select the WLAN interface. - [ap, sta] ap: Access-Point sta: Station
Parameter 2:	**<state>** **Default value** **Range**	Enables/Disables a WLAN interface. 'ap' on, 'sta' off [on, off] on: interface enabled off: interface disabled
Description:	GET: Returns the state of a WLAN interface. SET: Sets the state of a WLAN interface.	
GET-Rights:	0x00	
SET-Rights:	0x02	

Examples	
HTTP/JSON Get-Request	`["get","wlan","state","sta"]`
HTTP/JSON Get-Response	`["get","wlan","state","0","sta","on"]`
HTTP/JSON Set-Request	`["set","wlan","state","sta","off"]`
HTTP/JSON Set-Response	`["set","wlan","state","0","sta","off"]`
CMD-UART Get-Request	`get wlan state sta\x0d\x0a`
CMD-UART Get-Response	`get wlan state 0 sta on\x0d\x0a`
CMD-UART Set-Request	`set wlan state sta off\x0d\x0a`
CMD-UART Set-Response	`set wlan state 0 sta off\x0d\x0a`

4.1.3 Infrastructure Status (variable: status)

Definition		
Command option:	Get	
Parameter 1:	**<interface>** **Default** **Range**	Identifier to select the WLAN interface. - [ap, sta] ap: Access-Point sta: Station
Parameter 2	**<status>** **Default value** **Range**	Status of the infrastructure connection. 0 See list ⇨ 5.2 Wi-Fi Parameter.
Description:	Returns the status of the infrastructure connection.	
GET-Rights:	0x00	

Examples	
HTTP/JSON Get-Request	`["get","wlan","status","sta"]`
HTTP/JSON Get-Response	`["get","wlan","status","0","sta","4"]`
CMD-UART Get-Request	`get wlan status sta\x0d\x0a`
CMD-UART Get-Response	`get wlan status 0 sta 4\x0d\x0a`

4.1.4 Infrastructure Security (variable: sec)

Definition		
Command option:	Get	
Parameter 1:	**<interface>** **Default** **Range**	Identifier to select the WLAN interface. - [ap, sta] ap: Access-Point sta: Station
Parameter 2:	**<security>** **Default value** **Range** **Range**	Security setting of the infrastructure network. - [1,4] for Access-Point 1: OPEN 4: WPA2 [1...5] for Station 1: OPEN 2: WEP 3: WPA 4: WPA2 5: WPA2_MIX
Description:	Returns the security setting of the infrastructure network to which the application is connected.	
GET-Rights:	0x00	

Examples	
HTTP/JSON Get-Request	`["get","wlan","sec","sta"]`
HTTP/JSON Get-Response	`["get","wlan","sec","0","sta","2"]`
CMD-UART Get-Request	`get wlan sec sta\x0d\x0a`
CMD-UART Get-Response	`get wlan sec 0 sta 2\x0d\x0a`

4.1.5 WLAN Network Configuration (variable: cfg)

Definition		
Command option:	Set	
Parameter 1:	**<interface>** **Default** **Range**	Identifier to select the WLAN interface. - [ap, sta] ap: Access-Point sta: Station
Parameter 2:	**<ssid>** **Default value** **Range**	SSID for the WLAN network. - String with ASCII printable characters with a length of (1-32). SSIDs containing spaces have to be put in quotes and quotes in an SSID have to be marked with an additional quote in front.
Parameter 3:	**<psk>** - **Default value** **Range**	Pass phrase of the WLAN network. - For the access-point configuration the PSK must be presented as a string with ASCII printable characters with a length of (8-63). For a WEP security the password needs to be 5,10,13 or 26 chars.
Parameter 4	**<security>** **Default value** **Station- Range**	Security setting of the WLAN network. - [1..5] 1: OPEN 2: WEP 3: WPA 4: WPA2 5: WPA2_MIX Access-Point-Range [1 or 4] 1: OPEN 2: WEP (not supported) 3: WPA (not supported) 4: WPA2 5: WPA2_MIX (not supported)
Parameter 5:	**<radio-channel>** **Default value**	Optional parameter for setting a specific radio channel. -
Description:	**Station**: **Access-Point**:	With the variable, a connection to a chosen station network could be established. The pass phrase and the security level must match the settings of the infrastructure network. Configuration of the own access-point.
SET-Rights:	0x02	

Examples	
HTTP/JSON Set-Request	`["set","wlan","cfg","sta","testnetwork","password","4"]`
HTTP/JSON Set-Response	`["set","wlan","cfg","0","sta","testnetwork","password","4"]`
CMD-UART Set-Request	`set wlan cfg sta testnetwork password 4\x0d\x0a`
CMD-UART Set-Response	`set wlan cfg 0 sta testnetwork password 4\x0d\x0a`

4.1.6 Access Point Station List (variable: stlist)

Definition		
Command option:	Get	
Parameter 1:	**<interface>** **Default** **Range**	Identifier to select the WLAN interface. - [ap] ap: Access-Point
Parameter 2:	**<stlist>**	List of connected devices (MAC-Address).
Description:	With this parameter a list of connected devices can be requested. The MAC-Address of each device is listed.	
GET-Rights:	0x00	

Examples	
HTTP/JSON Set-Request	`["get","wlan","stlist","ap"]`
HTTP/JSON Set-Response	`["get","wlan","stlist","0","ap","01:02:03:04:05:06","11:12:13:14:15:16"]`
CMD-UART Set-Request	`get wlan stlist ap\x0d\x0a`
CMD-UART Set-Response	`get wlan stlist 0 ap 01:02:03:04:05:06 11:12:13:14:15:16\x0d\x0a`

4.1.7 Wireless Network Scan (variable: scan)

Definition	
Command option:	Set
Description:	Triggers a scan for available Wireless networks.
SET-Rights:	0x02

Examples	
HTTP/JSON Set-Request	`["set","wlan","scan"]`
HTTP/JSON Set-Response	`["set","wlan","scan","0"]`
CMD-UART Set-Request	`set wlan scan\x0d\x0a`
CMD-UART Set-Response	`set wlan scan 0\x0d\x0a`

4.1.8 Available Wireless Networks (variable: list)

Definition		
Command option:	Get	
Parameter 1:	**<ssid>**	SSID of the discovered wireless networks.
	Default value	-
	Range	String with ASCII printable characters with a length of (1-32).
	SSIDs containing spaces will be put in quotes, quotes in an SSID will be marked with an additional quote in front of it and non-ASCII characters will be displayed with its UTF-8 code and a "\u"-escape sequence in the front.	
Parameter 2:	**<security>**	Security level of the found wireless network.
	Default value	-
	Range	[1..5] 1: OPEN 2: WEP 3: WPA 4: WPA2 5: WPA2_MIX
Parameter 3:	**<signal>**	Signal strength of the found wireless network.
	Default value	-
	Range	[0…100]
Parameter 4:	**<radio channel>**	Radio channel of the WLAN network.
	Range	[0...13]
Description:	Returns a list of available wireless networks which were found by the network scan. All found networks up to 8 will be returned in a single Get-Respond. Each found network is described with its SSID, security type signal strength and channel. The possible first three parameters (1-4) describe the first network in the list. The possible next three parameters (5-8) describe the second network in the list and so on.	
GET-Rights:	0x00	

Examples	
HTTP/JSON Get-Request	`["get","wlan","list"]`
HTTP/JSON Get-Response	`["get","wlan","list","0","net1","2","50","6","net2","1","80","4"]`
CMD-UART Get-Request	`get wlan list\x0d\x0a`
CMD-UART Get-Response	`get wlan list 0 net1 2 50 6 net2 1 80 4\x0d\x0a`

4.1.9 WLAN Region Code (variable: rcode)

Definition		
Command option:	Get	
Parameter 1:	**<code>**	Region code of this module.
	Default value	EU
	Range	[EU, US]
Description:	Region code for the radio chip.	
GET-Rights:	0x00	

Examples	
HTTP/JSON Get-Request	`["get","wlan","rcode"]`
HTTP/JSON Get-Response	`["get","wlan","rcode","0","EU"]`
CMD-UART Get-Request	`get wlan rcode\x0d\x0a`
CMD-UART Get-Response	`get wlan rcode 0 EU\x0d\x0a`

4.1.10 WLAN BSSID (variable: bssid)

Definition		
Command option:	Get	
Parameter 1:	**<interface>**	Identifier to select the WLAN interface.
	Default	-
	Range	[ap, sta] ap: Access-Point sta: Station
Parameter 2:	**<bssid>**	Current BSSID.
	Default value	-
	SSIDs containing spaces will be put in quotes and quotes in an SSID will be marked with an additional quote in front.	
Description:	BSSID of the connected Wi-Fi network.	
GET-Rights:	0x00	

Examples	
HTTP/JSON Get-Request	`["get","wlan","bssid","sta"]`
HTTP/JSON Get-Response	`["get","wlan","bssid","0","sta","80:70:60:50:40:30"]`
CMD-UART Get-Request	`get wlan bssid sta\x0d\x0a`
CMD-UART Get-Response	`get wlan bssid 0 sta 80:70:60:50:40:30\x0d\x0a`

4.1.11 WLAN Channel (variable: channel)

Definition		
Command option:	Get	
Parameter 1:	**<interface>** **Default** **Range**	Identifier to select the WLAN interface. - [ap, sta] ap: Access-Point sta: Station
Parameter 2:	**<channel>** **Default value**	Current radio channel. -
Description:	Radio channel of the WLAN access point or the connected station network.	
GET-Rights:	0x00	

Examples	
HTTP/JSON Get-Request	`["get","wlan","channel","ap"]`
HTTP/JSON Get-Response	`["get","wlan","channel","0","ap","5"]`
CMD-UART Get-Request	`get wlan channel ap\x0d\x0a`
CMD-UART Get-Response	`get wlan channel 0 ap 5\x0d\x0a`

4.1.12 WLAN RSSI (variable: rssi)

Definition		
Command option:	Get	
Parameter 1:	**<interface>** **Default** **Range**	Identifier to select the WLAN interface. - [ap, sta] ap: Access-Point sta: Station
Parameter 2:	**<rssi>** **Default value** **Range**	Current RSSI. - [0…100] in %
Description:	RSSI of the WLAN interface.	
GET-Rights:	0x00	

Examples	
HTTP/JSON Get-Request	`["get","wlan","rssi","sta"]`
HTTP/JSON Get-Response	`["get","wlan","rssi","0","sta","80"]`
CMD-UART Get-Request	`get wlan rssi sta\x0d\x0a`
CMD-UART Get-Response	`get wlan rssi 0 sta 80\x0d\x0a`

4.2 Name

4.2.1 Device Name (variable: name_all)

Definition		
Command option:	Set	
Parameter 1:	**<name>**	Device name.
	Default value –	
	Range	String with maximum of 15 characters (see description for valid characters).
Description:	The device name is an universal name and is taken for the following services: - MDNS-Domain - MDNS-Serv1-Inst-Name - MDNS-Serv2-Inst-Name - NBNS-Name The device name is limited to twenty-six letters, ten digits and the hyphen character. It is not allowed to use spaces, underscores or other punctuation. A get command is not useful, because the set device name can differ, if one or more name services detect name conflicts. Nonetheless it is possible to get and also set the service name separately from each name service. See ⇨ 4.2.4 MDNS Server 1/2 (variable: mdns_serv1 || mdns_serv2) and 4.2.5 NetBIOS Name Service Name (variable: nbns_name).	
SET-Rights:	0x02	

Examples	
HTTP/JSON Set-Request	`["set","name","name_all","pan9420"]`
HTTP/JSON Set-Response	`["set","name","name_all","0","pan9420"]`
CMD-UART Set-Request	`set name name_all pan9420\x0d\x0a`
CMD-UART Set-Response	`set name name_all 0 pan9420\x0d\x0a`

4.2.2 MDNS Domain (variable: mdns_name)

Definition		
Command option:	Get / Set	
Parameter 1:	**<name>**	MDNS name.
	Default value	pan9420
	Range	Limited to a maximum of 31 characters. See description for valid characters.
Description:	The name is for the MDNS service. The MDNS name is limited to twenty-six letters, ten digits and the hyphen character. It is not allowed to use spaces, underscores or other punctuation.	
GET-Rights:	0x00	
SET-Rights:	0x04	

Examples	
HTTP/JSON Get-Request	`["get","name","mdns_name"]`
HTTP/JSON Get-Response	`["get","name","mdns_name","0","pan9420"]`
HTTP/JSON Set-Request	`["set","name","mdns_name","pan9420"]`
HTTP/JSON Set-Response	`["set","name","mdns_name","0","pan9420"]`
CMD-UART Get-Request	`get name mdns_name\x0d\x0a`
CMD-UART Get-Response	`get name mdns_name 0 pan9420\x0d\x0a`
CMD-UART Set-Request	`set name mdns_name pan9420\x0d\x0a`
CMD-UART Set-Response	`set name mdns_name 0 pan9420\x0d\x0a`

4.2.3 MDNS State (variable: mdns_state)

Definition		
Command option:	Get / Set	
Parameter 1:	**<state>**	Enables/Disables MDNS state.
	Default value	on
	Range	[on, off] on: state enabled off: state disabled
Description:	The MDNS state is to switch the MDNS service on or off.	
GET-Rights:	0x01	
SET-Rights:	0x04	

Examples	
HTTP/JSON Get-Request	`["get","name","mdns_state"]`
HTTP/JSON Get-Response	`["get","name","mdns_state","0","on"]`
HTTP/JSON Set-Request	`["set","name","mdns_state","on"]`
HTTP/JSON Set-Response	`["set","name","mdns_state","0","on"]`
CMD-UART Get-Request	`get name mdns_state\x0d\x0a`
CMD-UART Get-Response	`get name mdns_state 0 on\x0d\x0a`
CMD-UART Set-Request	`set name mdns_state on\x0d\x0a`
CMD-UART Set-Response	`set name mdns_state 0 on\x0d\x0a`

4.2.4 MDNS Server 1/2 (variable: mdns_serv1 || mdns_serv2)

Definition		
Command option:	Get / Set	
Parameter 1:	**<dev_name>**	Device name.
	Default value	pan9420
	Range	String with maximum of 31 characters.
	The device name is limited to twenty-six letters, ten digits and the hyphen character. It is not allowed to use spaces or other punctuation.	
Parameter 2:	**<prot_name>**	Protocol name.
	Default value **Default value**	for mdns_serv1: _http._tcp for mdns_serv2: _iot._tcp
	Range	String with maximum of 31 characters.
Parameter 3:	**<port>**	Port of the service.
	Default value **Default value**	for mdns_serv1: 80 for mdns_serv2: 80
	Range	[0…65535]
Parameter 4:	**<ttl>**	Time to live of the service.
	Default value **Default value**	for mdns_serv1: 120 for mdns_serv2: 120
	Range	[0…4294967294]
Parameter 5:	**<txt>**	Additional information of the service.
	Default value **Default value**	for mdns_serv1: dev=pan9420 for mdns_serv2: ver=1.1.0.6
	Range	String with maximum of 31 characters.
Description:	The MDNS service can be configured for individual use. By setting the <prot_name> every device can propagate its own service in the network.	
GET-Rights:	0x01	
SET-Rights:	0x04	

Examples	
HTTP/JSON Get-Request	`["get","name","mdns_serv1"]`
HTTP/JSON Get-Response	`["get","name","mdns_serv1","0","pan9420","_http._tcp","80","120","PAN9420text"]`
HTTP/JSON Set-Request	`["set","name","mdns_serv1","pan9420","_http._tcp","80","120","PAN9420text"]`
HTTP/JSON Set-Response	`["set","name","mdns_serv1","0","pan9420","_http._tcp","80","120","PAN9420text"]`
CMD-UART Get-Request	`get name mdns_serv1\x0d\x0a`
CMD-UART Get-Response	`get name mdns_serv1 0 pan9420 _http._tcp 80 120 PAN9420text\x0d\x0a`
CMD-UART Set-Request	`set name mdns_serv1 pan9420 _http._tcp 80 120 PAN9420text\x0d\x0a`
CMD-UART Set-Response	`set name mdns_serv1 0 pan9420 _http._tcp 80 120 PAN9420text\x0d\x0a`

4.2.5 NetBIOS Name Service Name (variable: nbns_name)

Definition		
Command option:	Get / Set	
Parameter 1:	**<name>**	NetBIOS Name Service name.
	Default value	pan9420
	Range	String with a maximum size of 15 characters.
Description:	The NetBIOS Name Service Name is limited to 15 characters.	
GET-Rights:	0x00	
SET-Rights:	0x04	

Examples	
HTTP/JSON Get-Request	`["get","name","nbns_name"]`
HTTP/JSON Get-Response	`["get","name","nbns_name","0","pan9420"]`
HTTP/JSON Set-Request	`["set","name","nbns_name","pan9420"]`
HTTP/JSON Set-Response	`["set","name","nbns_name","0","pan9420"]`
CMD-UART Get-Request	`get name nbns_name\x0d\x0a`
CMD-UART Get-Response	`get name nbns_name 0 pan9420\x0d\x0a`
CMD-UART Set-Request	`set name nbns_name pan9420\x0d\x0a`
CMD-UART Set-Response	`set name nbns_name 0 pan9420\x0d\x0a`

4.2.6 NetBIOS State (variable: nbns_state)

Definition		
Command option:	Get / Set	
Parameter 1:	**<state>**	Enables/Disables NBNS state.
	Default value	on
	Range	[on, off] on: state enabled off: state disabled
Description:	The NBNS state is to switch the NBNS service on or off.	
GET-Rights:	0x01	
SET-Rights:	0x04	

Examples	
HTTP/JSON Get-Request	`["get","name","nbns_state"]`
HTTP/JSON Get-Response	`["get","name","nbns_state","0","on"]`
HTTP/JSON Set-Request	`["set","name","nbns_state","on"]`
HTTP/JSON Set-Response	`["set","name","nbns_state","0","on"]`
CMD-UART Get-Request	`get name nbns_state\x0d\x0a`
CMD-UART Get-Response	`get name nbns_state 0 on\x0d\x0a`
CMD-UART Set-Request	`set name nbns_state on\x0d\x0a`
CMD-UART Set-Response	`set name nbns_state 0 on\x0d\x0a`

4.3 NET

4.3.1 IP-Configuration (variable: cfg)

Definition		
Command option:	Get / Set	
Parameter 1:	**<interface>** **Default** **Range**	Identifier to select the WLAN interface. - [ap, sta] ap: Access-Point sta: Station
Parameter 2:	**<mode>** **Default value** **Range**	IP Mode. ap: manual (fix IP) sta: auto (DHCP) ap: [manual] sta: [auto, manual]
Parameter 2:	**<local_ip>** **Default value** **Range**	IP address. - -
Parameter 3:	**<subnet_ip>** **Default value** **Range**	Subnet mask. - -
Parameter 4:	**<gateway_ip>** **Default value** **Range**	Gateway address. - -
Description:	IP configuration of an interface. The get-request requires only the interface and returns the IP-configuration of this interface. To set one interface to an automatic IP-configuration only the mode is required. To set one interface to a manual IP-configuration all parameters are required.	
GET-Rights:	0x00	
SET-Rights:	0x04	

Examples	
HTTP/JSON Get-Request	`["get","net","cfg","sta"]`
HTTP/JSON Get-Response	`["get","net","cfg","sta","auto","192.168.130.30","255.255.255.0","192.168.130.1"]`
HTTP/JSON Set-Request	`["set","net","cfg","sta",manual","192.168.130.30","255.255.255.0","192.168.130.1"]`
HTTP/JSON Set-Response	`["set","net","cfg","0","sta","manual","192.168.130.30","255.255.255.0","192.168.130.1"]`
CMD-UART Get-Request	`get net cfg sta\x0d\x0a`
CMD-UART Get-Response	`get net cfg 0 sta auto 192.168.130.30 255.255.255.0 192.168.130.1\x0d\x0a`
CMD-UART Set-Request	`set net cfg sta auto\x0d\x0a`
CMD-UART Set-Response	`set net cfg 0 sta auto\x0d\x0a`

4.3.2 DHCP Server Configuration (variable: dhcpserv)

Definition		
Command option:	Get / Set	
Parameter 1:	**<interface>** **Default** **Range**	Identifier to select the WLAN interface. - [ap] ap: Access-Point
Parameter 2:	**<state>** **Default value** **Range**	The state of the module. on [on, off]
Parameter 2:	**<start_ip>** **Default value** **Range**	IP address. 192.168.1.100 -
Parameter 3:	**<lease-time>** **Default value** **Range**	Lease time. 120 seconds [0…65535]
Parameter 4:	**<max-clients>** **Default value** **Range**	Maximum number of clients. 8 [0…8]
Description:	DHCP server Configuration for the access-point.	
GET-Rights:	0x01	
SET-Rights:	0x04	

Examples	
HTTP/JSON Get-Request	`["get","net","dhcpserv","ap"]`
HTTP/JSON Get-Response	`["get","net","dhcpserv","0","ap","on","192.168.1.100","120","8"]`
HTTP/JSON Set-Request	`["set","net","dhcpserv","ap","on","192.168.1.100","120","8"]`
HTTP/JSON Set-Response	`["set","net","dhcpserv","0","ap","on","192.168.1.100","120","8"]`
CMD-UART Get-Request	`get net dhcpserv ap\x0d\x0a`
CMD-UART Get-Response	`get net dhcpserv 0 ap on 192.168.1.100 120 8\x0d\x0a`
CMD-UART Set-Request	`set net dhcpserv ap on 192.168.1.100 120 8\x0d\x0a`
CMD-UART Set-Response	`set net dhcpserv 0 ap on 192.168.1.100 120 8\x0d\x0a`

4.3.3 Local Captive Portal (variable: captive)

Definition		
Command option:	Get / Set	
Parameter 1:	**<state>**	The state of the module.
	Default value	off
	Range	[on, off]
Description:	Captive Portal configuration.	
GET-Rights:	0x01	
SET-Rights:	0x04	

Examples	
HTTP/JSON Get-Request	`["get","net","captive"]`
HTTP/JSON Get-Response	`["get","net","captive","0","on"]`
HTTP/JSON Set-Request	`["set","net","captive","on"]`
HTTP/JSON Set-Response	`["set","net","captive","0","on"]`
CMD-UART Get-Request	`get net captive\x0d\x0a`
CMD-UART Get-Response	`get net captive 0 on\x0d\x0a`
CMD-UART Set-Request	`set net captive on\x0d\x0a`
CMD-UART Set-Response	`set net captive 0 on\x0d\x0a`

4.3.4 DNS-Server Configuration (variable: dnsserv)

Definition		
Command option:	Get / Set	
Parameter 1:	**<interface>** **Default** **Range**	Identifier to select the WLAN interface. - [ap, sta] ap: Access-Point sta: Station
Parameter 2:	**<index>** **Default value** **Range**	Server-Index. - [0,1]
Parameter 2:	**<dnsserv_ip>** **Default value** **Range**	IP-Address of the DNS-Server. - -
Description:	DNS-Server configuration.	
GET-Rights:	0x01	
SET-Rights:	0x04	

Examples	
HTTP/JSON Get-Request	`["get","net","dnsserv","sta","0"]`
HTTP/JSON Get-Response	`["get","net","dnsserv","0","sta","0","192.168.130.1"]`
HTTP/JSON Set-Request	`["set","net","dnsserv","sta","0","192.168.130.1"]`
HTTP/JSON Set-Response	`["set","net","dnsserv","0","sta","0","192.168.130.1"]`
CMD-UART Get-Request	`get net dnsserv sta 0\x0d\x0a`
CMD-UART Get-Response	`get net dnsserv 0 sta 0 192.168.130.1\x0d\x0a`
CMD-UART Set-Request	`set net dnsserv sta 0 192.168.130.1\x0d\x0a`
CMD-UART Set-Response	`set net dnsserv 0 sta 0 192.168.130.1\x0d\x0a`

4.3.5 Request PING (variable: ping)

Definition		
Command option:	Set	
Parameter 1:	**<ip-address>**	Destination IP Address.
	Default value	-
	Range	-
Parameter 2:	**<random-id>**	Random ID for this request (only used in response). This ID must be used for requesting the status.
	Default value	-
	Range	[0...65535]
Description:	The Ping request is only the command to trigger a ping to an IP address. The status of this request must be requested separately.	

Examples	
HTTP/JSON Set-Request	`["set","net","ping","192.168.1.1"]`
HTTP/JSON Set-Response	`["set","net","ping","0","192.168.1.1","12345"]`
CMD-UART Set-Request	`set net ping 192.168.1.1\x0d\x0a`
CMD-UART Set-Response	`set net ping 0 192.168.1.1 12345\x0d\x0a`

4.3.6 Response PING (variable: ping)

Definition		
Command option:	Get	
Parameter 1:	**<random-id>**	Request ID from ping request.
	Default value	-
	Range	[0...65535]
Parameter 2:	**<ip-address>**	Destination IP Address.
	Default value	-
	Range	-
Parameter 3:	**<status>**	Status of the ping request.
	Default value	-
	Range	[0…3] 0: Idle 1: Ping sending 2: Ping Success 3: Ping Error
Parameter 4:	**<time>**	Response time for the ping request in [ms].
	Default value	-
	Range	- Note: Value is only valid when status is ‘Ping Success’!
Description:	Status for a ping request.	

Examples	
HTTP/JSON Get-Request	`["get","net","ping","12345"]`
HTTP/JSON Get-Response	`["get","net","ping","0","12345","192.168.1.1","2","65"]`
CMD-UART Get-Request	`get net ping 12345\x0d\x0a`
CMD-UART Get-Response	`get net ping 0 12345 192.168.1.1 2 65\x0d\x0a`

4.4 Email

4.4.1 Mail Module Status

Status Code	Description
0	Not configured
1	Configuration ok – Ready to send
2	Mail is sending
3	Failed to connect to server
4	Authentication failed

4.4.2 API-Commands (module: mail)

4.4.2.1 User Mail Address (variable: from)

Definition		
Command option:	Get / Set	
Parameter 1:	**<address>**	Email address of the sender.
	Default value	-
	Range	String with maximum of 60 characters.
Description:	Mail address of the sender.	
GET-Rights:	0x01	
SET-Rights:	0x04	

Examples	
HTTP/JSON Get-Request	`["get","mail","from"]`
HTTP/JSON Get-Response	`["get","mail","from","0","maxmuster@muster.de"]`
HTTP/JSON Set-Request	`["set","mail","from","maxmuster@muster.de"]`
HTTP/JSON Set-Response	`["set","mail","from","0","maxmuster@muster.de"]`
CMD-UART Get-Request	`get mail from\x0d\x0a`
CMD-UART Get-Response	`get mail from 0 maxmuster@muster.de\x0d\x0a`
CMD-UART Set-Request	`set mail from maxmuster@muster.de\x0d\x0a`
CMD-UART Set-Response	`set mail from 0 maxmuster@muster.de\x0d\x0a`

4.4.2.2 Server Configuration (variable: server)

Definition		
Command option:	Get / Set	
Parameter 1:	**<server_name>**	Name of the mail server.
	Default value	-
	Range	String with maximum of 31 characters and a minimum of 4 characters (underscores not allowed).
Parameter 2:	**<port>**	Port of the server.
	Default value	587
	Range	[0…65535]
Parameter 3:	**<login>**	Login for the server.
	Default value	-
	Range	String with maximum of 31 characters.
Parameter 4:	**<password>**	Password for the login.
	Default value	-
	Range	String with maximum of 15 characters.
Description:	Server configuration for mail.	
GET-Rights:	0x01	
SET-Rights:	0x04	

Examples	
HTTP/JSON Get-Request	`["get","mail","server"]`
HTTP/JSON Get-Response	`["get","mail","server","0","MyServ","25","MyLogin"]`
HTTP/JSON Set-Request	`["set","mail","server","MyServ","25","MyLogin","MyPassword"]`
HTTP/JSON Set-Response	`["set","mail","server","0","MyServ","25","MyLogin","MyPassword"]`
CMD-UART Get-Request	`get mail server\x0d\x0a`
CMD-UART Get-Response	`get mail server 0 MyServ 25 MyLogin\x0d\x0a`
CMD-UART Set-Request	`set mail server MyServ 25 MyLogin MyPassword\x0d\x0a`
CMD-UART Set-Response	`set mail server 0 MyServ 25 MyLogin MyPassword\x0d\x0a`

4.4.2.3 Mail Sending (variable: send)

Definition		
Command option:	Set	
Parameter 1:	**<recipient>**	Recipient for the email.
	Default value	-
	Range	String with maximum of 48 characters.
Parameter 2:	**<subject>**	Subject of the email.
	Default value	-
	Range	String with maximum of 60 characters.
Parameter 3:	**<text>**	Text of the email.
	Default value	-
	Range	String with maximum of 250 characters.
Description:	Server configuration for email. The mail text only supports printable ASCII characters. A carriage return new line command is not supported.	
SET-Rights:	0x04	

Examples	
HTTP/JSON Set-Request	`["set","mail","send","receiver@web.de","subject","mail"]`
HTTP/JSON Set-Response	`["set","mail","send","0","receiver@web.de","subject","mail"]`
CMD-UART Set-Request	`set mail send receiver@web.de subject mail\x0d\x0a`
CMD-UART Set-Response	`set mail send 0 receiver@web.de subject mail\x0d\x0a`

4.4.2.4 Mail Status (variable: status)

Definition		
Command option:	Get	
Parameter:	**<status>**	Status of the mail module.
	Default value	0
	Range	See list ⇨ 4.4.1 Mail Module Status.
Description:	Status of the mail module.	
GET-Rights:	0x01	

Examples	
HTTP/JSON Get-Request	`["get","mail","status"]`
HTTP/JSON Get-Response	`["get","mail","status","0","1"]`
CMD-UART Get-Request	`get mail status\x0d\x0a`
CMD-UART Get-Response	`get mail status 0 1\x0d\x0a`

4.4.2.5 Mail Error (variable: error)

Definition		
Command option:	Get	
Parameter 1:	**<value>**	Value of the error.
	Default value	0
	Range	[0-1] 0: No error 1: Error
Description:	Error value of the email module.	
GET-Rights:	0x01	

Examples	
HTTP/JSON Get-Request	`["get","mail","error"]`
HTTP/JSON Get-Response	`["get","mail","error","0","0"]`
CMD-UART Get-Request	`get mail error\x0d\x0a`
CMD-UART Get-Response	`get mail error 0 0\x0d\x0a`

4.5 System

4.5.1 API-Commands (module: system)

4.5.1.1 Version (variable: version)

Definition		
Command option:	Get	
Parameter 1:	**<version>**	Software version.
	Default value	0.0.0.0
	Range	String with maximum of 12 characters.
Description:	Returns the software version.	
GET-Rights:	0x00	

Examples	
HTTP/JSON Get-Request	`["get","system","version"]`
HTTP/JSON Get-Response	`["get","system","version","0","0.0.0.0"]`
CMD-UART Get-Request	`get system version\x0d\x0a`
CMD-UART Get-Response	`get system version 0 0.0.0.1\x0d\x0a`

4.5.1.2 Firmware Version (variable: firmware)

Definition		
Command option:	Get	
Parameter 1:	**<firmware>**	WiFi-Firmware version.
	Default value	-
	Range	-
Description:	Returns the WiFi-firmware version.	
GET-Rights:	0x00	

Examples	
HTTP/JSON Get-Request	`["get","system","firmware"]`
HTTP/JSON Get-Response	`["get","system","firmware","0","V_01_01_02"]`
CMD-UART Get-Request	`get system firmware\x0d\x0a`
CMD-UART Get-Response	`get system firmware 0 V_01_01_02\x0d\x0a`

4.5.1.3 MAC Address (variable: macaddr)

Definition		
Command option:	get	
Parameter 1:	**<macaddr>**	MAC Address.
	Default value	-
	Range	-
Description:	Returns the MAC address.	
GET-Rights:	0x00	

Examples	
HTTP/JSON Get-Request	`["get","system","macaddr"]`
HTTP/JSON Get-Response	`["get","system","macaddr","0","0f:0e:03:02:01:00"]`
CMD-UART Get-Request	`get system macaddr\x0d\x0a`
CMD-UART Get-Response	`get system macaddr 0 0f:0e:03:02:01:00\x0d\x0a`

4.5.1.4 Serial Number (variable: serialnum)

Definition		
Command option:	Get	
Parameter 1:	**<serialnum >**	The serial number of the product.
	Default value	-
	Range	-
Description:	Returns the serial number.	
GET-Rights:	0x00	

Examples	
HTTP/JSON Get-Request	`["get","system","serialnum"]`
HTTP/JSON Get-Response	`["get","system","serialnum","0","Ser00001"]`
CMD-UART Get-Request	`get system serialnum\x0d\x0a`
CMD-UART Get-Response	`get system serialnum 0 Ser00001\x0d\x0a`

4.5.1.5 Radio Firmware Version (variable: radio_ver)

Definition		
Command option:	Get	
Parameter 1:	**<radio_ver >**	The radio version of the module.
	Default value	-
	Range	-
Description:	Returns the firmware version of the Radio module.	
GET-Rights:	0x00	

Examples	
HTTP/JSON Get-Request	`["get","system","radio_ver"]`
HTTP/JSON Get-Response	`["get","system","radio_ver","0","V_03_02_01"]`
CMD-UART Get-Request	`get system radio_ver\x0d\x0a`
CMD-UART Get-Response	`get system radio_ver 0 V_03_02_01\x0d\x0a`

4.5.1.6 Bootloader Version (variable: bootl_ver)

Definition		
Command option:	Get	
Parameter 1:	**<bootl_ver >**	The bootloader version of the module.
	Default value	-
	Range	-
Description:	Returns the bootloader version of the Wi-Fi module.	
GET-Rights:	0x00	

Examples	
HTTP/JSON Get-Request	`["get","system","bootl_ver"]`
HTTP/JSON Get-Response	`["get","system","bootl_ver","0","V_01_02_01"]`
CMD-UART Get-Request	`get system bootl_ver\x0d\x0a`
CMD-UART Get-Response	`get system bootl_ver 0 V_01_02_01\x0d\x0a`

4.5.1.7 Hardware Revision (variable: hwrev)

Definition		
Command option:	Get	
Parameter 1:	**<hwrev >**	The hardware revision of the module.
	Default value	-
	Range	-
Description:	Returns the hardware revision number	
GET-Rights:	0x00	

Examples	
HTTP/JSON Get-Request	`["get","system","hwrev"]`
HTTP/JSON Get-Response	`["get","system","hwrev","0","03"]`
CMD-UART Get-Request	`get system hwrev\x0d\x0a`
CMD-UART Get-Response	`get system hwrev 0 03\x0d\x0a`

4.5.1.8 Restart (variable: restart)

Definition	
Command option:	Set
Description:	This parameter triggers a restart of the Hardware. Furthermore the host-controller will be informed that the restart was executed. A restart can also be triggered by hardware and the same info message will be generated.
SET-Rights:	0x04

Examples	
HTTP/JSON Set-Request	`["set","system","restart"]`
HTTP/JSON Set-Response	`["set","system","restart","0"]`
CMD-UART Set-Request	`set system restart\x0d\x0a`
CMD-UART Set-Response	`set system restart 0\x0d\x0a`
CMD-UART Notification	`info system restart\x0d\x0a`

4.5.1.9 Reset (variable: factory)

Definition	
Command option:	Set
Description:	This parameter initiates a factory reset of the software. First the host controller will be informed that the device will be set back to factory settings, then the configuration will be set back lastly the device will be restarted. An info message will inform the host controller about the reset. This can also be triggered by hardware and the same info message will be generated.
SET-Rights:	0x04

Examples	
HTTP/JSON Set-Request	`["set","system","factory"]`
HTTP/JSON Set-Response	`["set","system","factory","0"]`
CMD-UART Set-Request	`set system factory\x0d\x0a`
CMD-UART Set-Response	`set system factory 0\x0d\x0a`
CMD-UART Notification	`info system factory\x0d\x0a`

4.5.1.10 Save Mode (variable: savemode)

Definition		
Command option:	Set	
Parameter 1:	**<savemode>**	The save mode can set to manual or automatic.
	Default value	auto
	Range	[auto, manual]
Description:	Manual means the user has to trigger a save of the current configuration. Automatic means, the configuration will be saved automatically after every change.	
SET-Rights:	0x04	

Examples	
HTTP/JSON Set-Request	`["set","system","savemode","auto"]`
HTTP/JSON Set-Response	`["set","system","savemode","0","auto"]`
HTTP/JSON Get-Request	`["get","system","savemode"]`
HTTP/JSON Get-Response	`["get","system","savemode","0","auto"]`
CMD-UART Set-Request	`set system savemode auto\x0d\x0a`
CMD-UART Set-Response	`set system savemode 0 auto\x0d\x0a`
CMD-UART Get-Request	`get system savemode\x0d\x0a`
CMD-UART Get-Response	`get system savemode 0 auto\x0d\x0a`

4.5.1.11 Save the Configuration (variable: savecfg)

Definition	
Command option:	Set
Description:	This command triggers a save of the configuration.
SET-Rights:	0x02

Examples	
HTTP/JSON Set-Request	`["set","system","savecfg"]`
HTTP/JSON Set-Response	`["set","system","savecfg","0"]`
CMD-UART Set-Request	`set system savecfg\x0d\x0a`
CMD-UART Set-Response	`set system savecfg 0\x0d\x0a`

4.5.1.12 Power Save Mode (variable: psm)

Definition		
Command option:	Set	
Parameter 1:	**<mode>**	Power save mode.
	Default value	-
	Range	[0,1,2] 0 = Disable power save mode 1 = IEEE power save modus (only if AP is disabled) 2 = Power down
Parameter 2: **(only for mode 1)**	**<dtims>**	Multiple dtims (16Bit).
	Default value	-
	Range	[0…65535] 65534: CLOSEST_DTIM_TO_LISTEN_INTERVAL)
Parameter 3: **(only for mode 1)**	**<BeaconTimeout>**	Beacon miss timeout (16Bit).
	Default value	-
	Range	[0…65535]
Parameter 4: **(only for mode 1)**	**<ListInt>**	Local listen interval (16Bit).
	Default value	-
	Range	[0…65535]
Parameter 5: **(only for mode 1)**	**<AdhocWakePeriod>**	Adhoc wake period (16Bit).
	Default value	-
	Range	[0…65535]
Parameter 6: **(only for mode 1)**	**<delay>**	Delay to powersave (16Bit).
	Default value	-
	Range	[0…65535]
Parameter 7: **(only for mode 1)**	**<delay>**	Null packet interval (32Bit).
	Default value	-
	Range	[0…4294967295]
Description:	Sets the PAN9420 into power save mode. The IEEE power save mode can only be configured if AP is disabled. It is also important to deactivate the STA interface, when configuring the IEEE power save mode. If the STA WLAN interface is connected, the IEEE power save mode will not work.	
SET-Rights:	0x04	

Examples	
HTTP/JSON Set-Request	`["set","system","psm","1","65534","5","0","0","0","30"]`
HTTP/JSON Set-Response	`["set","system","psm","0","1","65534","5","0","0","0","30"]`
HTTP/JSON Set-Request	`["set","system","psm","2"]`
HTTP/JSON Set-Response	`["set","system","psm","0","2"]`
CMD-UART Set-Request	`set system psm 1 65534 5 0 0 0 30\x0d\x0a`
CMD-UART Set-Response	`set system psm 0 1 65534 5 0 0 0 30\x0d\x0a`
CMD-UART Set-Request	`set system psm 2\x0d\x0a`
CMD-UART Set-Response	`set system psm 0 2\x0d\x0a`

4.5.1.13 Device Information (variable: device)

Definition		
Command option:	Get	
Parameter 1:	**<class>**	Device class.
	Default value	-
	Range	-
Parameter 2:	**<type>**	Device type.
	Default value	-
	Range	-
Parameter 3:	**<name>**	Device name.
	Default value	-
	Range	-
Description:	Information about the device.	
GET-Rights:	0x00	

Examples	
HTTP/JSON Get-Request	`["get","system","device"]`
HTTP/JSON Get-Response	`["get","system","device","0","class","type","name"]`
CMD-UART Get-Request	`get system device\x0d\x0a`
CMD-UART Get-Response	`get system device 0 class type name\x0d\x0a`

4.5.1.14 Device Description (variable: dev_dec)

Definition	
Command option:	Get
Parameter 1:	**<description>** Device description. **Default value** - **Range** -
Description:	Returns the device description.
GET-Rights:	0x00

Examples	
HTTP/JSON Get-Request	`["get","system","dev_dec"]`
HTTP/JSON Get-Response	`["get","system","dev_dec","0","device description"]`
CMD-UART Get-Request	`get system dev_dec\x0d\x0a`
CMD-UART Get-Response	`get system dev_dec 0 "device description"\x0d\x0a`

4.5.1.15 Manufacturer ID (variable: id_mfd)

Definition	
Command option:	Get
Parameter 1:	**<id>** Manufacturer ID. **Default value** - **Range** -
Description:	Returns the manufacturer ID.
GET-Rights:	0x00

Examples	
HTTP/JSON Get-Request	`["get","system","id_mfd"]`
HTTP/JSON Get-Response	`["get","system","id_mfd","0","123456789"]`
CMD-UART Get-Request	`get system id_mfd\x0d\x0a`
CMD-UART Get-Response	`get system id_mfd 0 123456789\x0d\x0a`

4.5.1.16 Product ID (variable: id_product)

Definition		
Command option:	Get	
Parameter 1:	**<id>**	Product ID.
	Default value	-
	Range	-
Description:	Returns the product ID.	
GET-Rights:	0x00	

Examples	
HTTP/JSON Get-Request	`["get","system","id_product"]`
HTTP/JSON Get-Response	`["get","system","id_product","0","123456789"]`
CMD-UART Get-Request	`get system id_product\x0d\x0a`
CMD-UART Get-Response	`get system id_ product 0 123456789\x0d\x0a`

4.5.1.17 Serial ID (variable: id_serial)

Definition		
Command option:	Get	
Parameter 1:	**<id>**	Serial ID.
	Default value	-
	Range	-
Description:	Returns the serial ID number.	
GET-Rights:	0x00	

Examples	
HTTP/JSON Get-Request	`["get","system","id_serial"]`
HTTP/JSON Get-Response	`["get","system","id_serial","0","123456789"]`
CMD-UART Get-Request	`get system id_serial\x0d\x0a`
CMD-UART Get-Response	`get system id_serial 0 123456789\x0d\x0a`

4.5.1.18 URL Device (variable: url_device)

Definition		
Command option:	Get	
Parameter 1:	**<url>**	URL for the device information.
	Default value	-
	Range	-
Description:	Returns a URL for the device.	
GET-Rights:	0x00	

Examples	
HTTP/JSON Get-Request	`["get","system","url_device"]`
HTTP/JSON Get-Response	`["get","system","url_device","0","http://device.de"]`
CMD-UART Get-Request	`get system url_device\x0d\x0a`
CMD-UART Get-Response	`get system url_device 0 http://device.de\x0d\x0a`

4.5.1.19 URL Shop (variable: url_shop)

Definition		
Command option:	Get	
Parameter 1:	**<url>**	URL for a shop for the device.
	Default value	-
	Range	-
Description:	Returns a shop URL for the device.	
GET-Rights:	0x00	

Examples	
HTTP/JSON Get-Request	`["get","system","url_shop"]`
HTTP/JSON Get-Response	`["get","system","url_shop","0","http://deviceshop.de"]`
CMD-UART Get-Request	`get system url_shop\x0d\x0a`
CMD-UART Get-Response	`get system url_shop 0 http://deviceshop.de\x0d\x0a`

4.5.1.20 URL Help (variable: url_help)

Definition		
Command option:	Get	
Parameter 1:	**<url>**	URL for a help page for the device.
	Default value	-
	Range	-
Description:	Returns a help URL for the device.	
GET-Rights:	0x00	

Examples	
HTTP/JSON Get-Request	`["get","system","url_help"]`
HTTP/JSON Get-Response	`["get","system","url_help","0","http://devicehelp.de"]`
CMD-UART Get-Request	`get system url_help\x0d\x0a`
CMD-UART Get-Response	`get system url_help 0 http://devicehelp.de\x0d\x0a`

4.5.1.21 Company (variable: company)

Definition		
Command option:	Get	
Parameter 1:	**<company>**	Company name.
	Default value	-
	Range	-
Description:	Returns the company name.	
GET-Rights:	0x00	

Examples	
HTTP/JSON Get-Request	`["get","system","company"]`
HTTP/JSON Get-Response	`["get","system","company","0","name"]`
CMD-UART Get-Request	`get system company\x0d\x0a`
CMD-UART Get-Response	`get system company 0 name\x0d\x0a`

4.5.1.22 API (variable: api)

Definition		
Command option:	Get	
Parameter 1:	**<api>**	API version used.
	Default value	-
	Range	-
Description:	Returns the API version of module.	
GET-Rights:	0x00	

Examples	
HTTP/JSON Get-Request	`["get","system","api"]`
HTTP/JSON Get-Response	`["get","system","api","0","iot.1 1.0.0"]`
CMD-UART Get-Request	`get system api\x0d\x0a`
CMD-UART Get-Response	`get system api 0 iot.1 1.0.0\x0d\x0a`

4.5.1.23 Uptime (variable: uptime)

Definition		
Command option:	Get	
Parameter 1:	**<uptime>**	Time since startup.
	Default value	-
	Range	-
Description:	Returns the Time since startup.	
GET-Rights:	0x00	

Examples	
HTTP/JSON Get-Request	`["get","system","uptime"]`
HTTP/JSON Get-Response	`["get","system","uptime","0","0:00:36:52"]`
CMD-UART Get-Request	`get system uptime\x0d\x0a`
CMD-UART Get-Response	`get system uptime 0 0:00:36:52\x0d\x0a`

4.6 User Management

4.6.1 User-Rights

Rights	Description
0x80	Rights to change the user configuration update
0x40	Rights to do a firmware update
0x20	Rights to SET/GET data via BINUART
0x10	Rights to SET/GET data via CMDUART HIGH
0x08	Rights to SET/GET data via CMDUART LOW
0x04	Rights to SET parameters with a high priority
0x02	Rights to SET parameters with a low priority
0x01	Rights to GET parameters

4.6.2 API-Commands (module: user)

4.6.2.1 User Edit/Add (variable: edit)

Definition		
Command option:	Set	
Parameter 1:	**<index#>**	User index number.
	Default value	-
	Range	[0…4]
Parameter 2:	**<name>**	Login name.
	Default value	-
	Range	String with maximum of 30 characters
Parameter 3:	**<password>**	Password for login.
	Default value	-
	Range	String with maximum of 30 characters
Parameter 4:	**<rights>**	Rights of the user.
	Default value	-
	Range	[0…255]
		See list ⇨ 4.6.1 User-Rights.
Description:	Sets the user with login name, password and rights.	
SET-Rights:	0x80	

Examples	
HTTP/JSON Set-Request	`["set","user","edit","1","Max","password","61"]`
HTTP/JSON Set-Response	`["set","user","edit","0","1","Max","password","61"]`
CMD-UART Set-Request	`set user edit 1 Max password 61\x0d\x0a`
CMD-UART Set-Response	`set user edit 0 1 Max password 61\x0d\x0a`

4.6.2.2 User Name (variable: name)

Definition		
Command option:	Get	
Parameter 1:	**<index#>**	User index number.
	Default value	-
	Range	[0…4]
Parameter 2:	**<name#>**	User name.
	Default value	-
	Range	-
Parameter 3:	**<rights#>**	User rights.
	Default value	-
	Range	[0…255]
		See list ⇨ 4.6.1 User-Rights.
Description:	Returns the user name for the login.	
GET-Rights:	0x01	

Examples	
HTTP/JSON Get-Request	`["get","user","name","1"]`
HTTP/JSON Get-Response	`["get","user","name","0","1","Max","61"]`
CMD-UART Get-Request	`get user name 1\x0d\x0a`
CMD-UART Get-Response	`get user name 0 1 Max 61\x0d\x0a`

4.6.2.3 User Delete (variable: delete)

Definition		
Command option:	Set	
Parameter:	**<index#>**	User index number.
	Default value	-
	Range	[1…4]
Description:	Deletes a user. The user 0 cannot be deleted.	
SET-Rights:	0x80	

Examples	
HTTP/JSON Set-Request	`["set","user","delete","1"]`
HTTP/JSON Set-Response	`["set","user","delete","0","1"]`
CMD-UART Set-Request	`set user delete 1\x0d\x0a`
CMD-UART Set-Response	`set user delete 0 1\x0d\x0a`

4.6.2.4 Default Login Active (variable: dlogin)

Definition	
Command option:	Get
Description:	Returns '1' for default login is still set. Returns '0' for login has been changed.
GET-Rights:	0x01

Examples	
HTTP/JSON Get-Request	`["get","user","dlogin"]`
HTTP/JSON Get-Response	`["get","user","dlogin","0","1"]`
CMD-UART Get-Request	`get user dlogin\x0d\x0a`
CMD-UART Get-Response	`get user dlogin 0 1\x0d\x0a`

4.6.2.5 Default Rights (variable: drights)

Definition		
Command option:	Get / Set	
Parameter 1:	**<drights>**	Parameter default rights if no authentication was requested.
	Default value	0
	Range	[0…255]
Description:	This parameter is used for default rights, if no authentication is needed for the files.	
GET-Rights:	0x01	
SET-Rights:	0x80	

Examples	
HTTP/JSON Get-Request	`["get","user","drights"]`
HTTP/JSON Get-Response	`["get","user","drights","0","0"]`
HTTP/JSON Set-Request	`["set","user","drights","1"]`
HTTP/JSON Set-Response	`["set","user","drights","0","1"]`
CMD-UART Get-Request	`get user drights\x0d\x0a`
CMD-UART Get-Response	`get user drights 0 0\x0d\x0a`
CMD-UART Set-Request	`set user drights 1\x0d\x0a`
CMD-UART Set-Response	`set user drights 0 1\x0d\x0a`

4.6.2.6 My Rights (variable: myrights)

Definition		
Command option:	Get	
Parameter 1:	**<myrights>**	The current user rights.
	Default value	0 (if no user logged in) 255 (for user 0)
	Range	[0…255]
Description:	This parameter is used to display the current rights for a logged in user.	
GET-Rights:	0x00	

Examples	
HTTP/JSON Get-Request	`["get","user","myrights"]`
HTTP/JSON Get-Response	`["get","user","myrights","0","3"]`
CMD-UART Get-Request	`get user myrights\x0d\x0a`
CMD-UART Get-Response	`get user myrights 0 3\x0d\x0a`

4.6.2.7 Maximum User (variable: maxuser)

Definition		
Command option:	Get	
Parameter 1:	**<maxuser>**	The number of users on the system.
	Default value	5
Description:	This parameter returns the current number of user for the software.	
GET-Rights:	0x01	

Examples	
HTTP/JSON Get-Request	`["get","user","maxuser"]`
HTTP/JSON Get-Response	`["get","user","maxuser","0","5"]`
CMD-UART Get-Request	`get user maxuser\x0d\x0a`
CMD-UART Get-Response	`get user maxuser 0 5\x0d\x0a`

4.6.2.8 Authentication (variable: auth)

Definition		
Command option:	Get / Set	
Parameter 1:	**<auth>**	Authentication for telnet enabled/disabled.
	Default value	on
	Range	[on, off]
Description:	With this parameter the telnet authentication can be enabled or disabled.	
GET-Rights:	0x01	
SET-Rights:	0x80	

Examples	
HTTP/JSON Get-Request	`["get","telnet","auth"]`
HTTP/JSON Get-Response	`["get","telnet","auth","0","on"]`
HTTP/JSON Set-Request	`["set","telnet","auth","off"]`
HTTP/JSON Set-Response	`["set","telnet","auth","0","off"]`
CMD-UART Get-Request	`get telnet auth\x0d\x0a`
CMD-UART Get-Response	`get telnet auth 0 on\x0d\x0a`
CMD-UART Set-Request	`set telnet auth off\x0d\x0a`
CMD-UART Set-Response	`set telnet auth 0 off\x0d\x0a`

4.7 Firmware Update

The PAN9420 enables to perform software updates over the air as well as over UART.

The updates can address various targets such as the PAN9420

- firmware,
- flash images (certificates, websites, etc.) and
- configuration as well as
- host controller firmware (only over the air).

4.7.1 Firmware Update Status

Status Code	Description
0	Ready for Update
1	Update active
2	Checking for new firmware
3	Update done
4	Update failed

4.7.2 API-Commands (module: fwu)

4.7.2.1 Progress (variable: progress)

Definition		
Command option:	Get	
Parameter 1:	**<progress>**	The firmware update progress.
	Default value	-
	Range	[0…100] in %
Description:	The command returns the progress of the firmware update process.	
GET-Rights:	0x01	

Examples	
HTTP/JSON Get-Request	`["get","fwu","progress"]`
HTTP/JSON Get-Response	`["get","fwu","progress","0","50"]`
CMD-UART Get-Request	`get fwu progress\x0d\x0a`
CMD-UART Get-Response	`get fwu progress 0 50\x0d\x0a`

4.7.2.2 Status (variable: status)

Definition		
Command option:	Get	
Parameter 1:	**<status>**	The firmware update status.
	Default value	0
	Range	See list ⇨ 4.7.1 Firmware Update Status.
Description:	Returns the status of the firmware update process.	
GET-Rights:	0x01	

Examples	
HTTP/JSON Get-Request	`["get","fwu","status"]`
HTTP/JSON Get-Response	`["get","fwu","status","0","0"]`
CMD-UART Get-Request	`get fwu status\x0d\x0a`
CMD-UART Get-Response	`get fwu status 0 0\x0d\x0a`

4.7.2.3 Mode (variable: mode)

Definition		
Command option:	Set	
Parameter 1:	**<mode>**	A mode will trigger a functionality of the firmware update.
	Default value	-
	Range	[check, update, uart, uart off]
		check: Requests the HTTP server for updates.
		update: Initiates a firmware update using the binuart.
		uart: Enables the UART firmware update using the binuart.
		uart off: Disables the UART firmware update mode.
	Note: Turning the UART update mode off cannot be reactivated.	
Parameter 2:	**<version>**	Optional parameter for the mode 'update'. Version number for the update. If this parameter is empty, the latest version will be installed.
	Default value	-
	Range	-
Description:	Setting the mode to 'check' will trigger a request to get the current firmware version on the server. Setting the mode to 'update' will trigger an execution of the automatic firmware update. Setting the mode to 'uart' will set the firmware update module to receive an update file from the host controller. After sending the command 'uart off' the feature will be deactivated.	
SET-Rights:	0x80	

Examples	
HTTP/JSON Set-Request	`["set","fwu","mode","uart"]`
HTTP/JSON Set-Response	`["set","fwu","mode","0","uart"]`
CMD-UART Set-Request	`set fwu mode uart\x0d\x0a`
CMD-UART Set-Response	`set fwu 0 mode uart\x0d\x0a`
CMD-UART Notification	`info fwu uart ready\x0d\x0a`
CMD-UART Notification	`info fwu uart timeout\x0d\x0a`
CMD-UART Notification	`info fwu uart error\x0d\x0a`

4.7.2.4 Info (variable: info)

Definition		
Command option:	Get	
Parameter 1:	**<timestamp>**	Information about the time since the last firmware update ‘check’.
	Default value	-
	Range	-
Parameter 2:	**<last-version>**	Information about the version on the server.
	Default value	-
	Range	-
Description:	Returns the information about the last firmware update check, with the time since the last check in seconds and the version number returns from the server.	
GET-Rights:	0x01	

Examples	
HTTP/JSON Get-Request	`["get","fwu","info"]`
HTTP/JSON Get-Response	`["get","fwu","info","0",55112","V_01_02_03"]`
CMD-UART Get-Request	`get fwu info\x0d\x0a`
CMD-UART Get-Response	`get fwu info 0 55112 V_01_02_03\x0d\x0a`

4.7.2.5 Server (variable: server)

Definition		
Command option:	Get / Set	
Parameter 1:	**<server_name>**	Server address for the firmware update server.
	Default value	-
	Range	-
Parameter 2:	**<port>**	Server port for the firmware update server.
	Default value	443
	Range	-
Parameter 3:	**<security>**	Firmware update server security.
	Default value	auto
	Range	[off, on, auto]
Description:	Server configuration for the firmware update server.	
GET-Rights:	0x01	
SET-Rights:	0x04	

Examples	
HTTP/JSON Get-Request	`["get","fwu","server"]`
HTTP/JSON Get-Response	`["get","fwu","server","0","update.de","443","auto"]`
HTTP/JSON Set-Request	`["set","fwu","server","update.de","443","auto"]`
HTTP/JSON Set-Response	`["set","fwu","server","0","update.de","443","auto"]`
CMD-UART Get-Request	`get fwu server\x0d\x0a`
CMD-UART Get-Response	`get fwu server 0 update.de 443 auto\x0d\x0a`
CMD-UART Set-Request	`set fwu server update.de 443 auto\x0d\x0a`
CMD-UART Set-Response	`set fwu server 0 update.de 443 auto\x0d\x0a`

4.7.2.6 Resource (variable: resource)

Definition		
Command option:	Get / Set	
Parameter 1:	**<resource>**	Resource name of the file for the firmware-update.
	Default value	-
	Range	-
Description:	Resource file name for the firmware update server.	
GET-Rights:	0x01	
SET-Rights:	0x04	

Examples	
HTTP/JSON Get-Request	`["get","fwu","resource"]`
HTTP/JSON Get-Response	`["get","fwu","resource","0","firmware.fwu"]`
HTTP/JSON Set-Request	`["set","fwu","resource","firmware.fwu"]`
HTTP/JSON Set-Response	`["set","fwu","resource","0","firmware.fwu"]`
CMD-UART Get-Request	`get fwu resource\x0d\x0a`
CMD-UART Get-Response	`get fwu resource 0 firmware.fwu\x0d\x0a`
CMD-UART Set-Request	`set fwu resource firmware.fwu\x0d\x0a`
CMD-UART Set-Response	`set fwu resource 0 firmware.fwu\x0d\x0a`

4.8 Command UART

The command UART (CMD UART) is used for host controller communication. By default, this interface is mapped to the UART 0 port and the communication needs to be string based. Every byte received on the CMD UART will be parsed and recognized commands will be executed. Service applications can also send commands through the CMD UART to the host controller (read and write commands). For this, the host controller will be informed with unsolicited messages (info commands) from the PAN9420.

4.8.1 Default CMD UART Configuration

The following table shows the default settings of the CMD UART.

Setting	Value
Baud rate	115200
Data bits	8
Parity	0
Stop bits	1
Flow Control	None

4.8.2 Info Messages

In some cases it is necessary to inform the host controller about an event or an upcoming process of the module. The PAN9420 implements such an information mechanism, so that the host controller has the chance to implement functionalities to handle these events.

At the moment the following unsolicited commands are supported by the PAN9420:

Definition	
Commands	**Description**
“info system restart”	Used to inform the host controller that the device will be restarted now. This can be triggered by software and hardware.
“info system factory”	Used to inform the host controller that the device will be reset back to factory settings now. This can be triggered by software and hardware.
“info fwu file <type> <id> <version> <size>”	Used to inform the host controller that a firmware update file was received which will be forwarded to the host controller using the binary UART after 5 seconds.
“info fwu uart ready”	Informs the user that all data for the firmware update via the host controller has been successfully transmitted.
“info fwu uart error”	Informs that the file was not correct or another error occurred during the firmware update process.
“info fwu uart timeout”	Informs that a timeout occurred during the update process.
“info udp recvmsg <portcfg-id> <ip> <port> <size>”	Used to inform the host controller that the PAN9420 has received UDP data which will be forwarded using the BIN UART to the host controller after this message. **<portcfg-id>** The entry in the UDP port config table. **<ip>** The IP address of the message source. **<port>** The port of the message source. **<size>** The size of the received message.

Definition	
“info httpc header <status-code> <mode> <datasize>”	This info is generated (when using the send-command) to inform the host controller about the received answer from the HTTP server (HTTP status code) and the mode which will be used for the transfer (normal or chunked). If the mode is chunked no data size will be received, because it was not delivered from the server. **<status-code>** Value from HTTP header. **<mode>** The transfer mode. 0: normal 1: chunked **<datasize>** Expected data size in bytes (when not chunked). Note: This message will only be sent if an answer from the server was received!
“info httpc status <status> <datasize>”	This info will be received after the HTTP client transfer (using the send-command) has finished successfully, but also when there was an error. **<status>** See ⇨ 4.10.1 HTTP Client Status Values . **<datasize>** Received bytes via HTTP.
“info mqttc connect OK”	Informs that the MQTT client has connected successfully.
“info mqttc connection ERROR”	Informs that the MQTT client discovered an error while connecting to a broker.
“info mqttc publish OK”	Informs that the MQTT client has successfully published a topic.
“info mqttc publish ERROR”	Informs that the MQTT client discovered an error publishing a topic.
“info mqttc subscribe OK”	Informs that the MQTT client successfully subscribed to a topic.
“info mqttc subscribe ERROR”	Informs that the MQTT client discovered an error subscribing to a topic.
“info mqttc unsubscribe OK”	Informs that the MQTT client successfully unsubscribed from a topic.
“info mqttc unsubscribe ERROR”	Informs that the MQTT client discovered an error unsubscribing from a topic.
“info mqttc recvmsg <QoS> <topic> <retain> <duplicate> <totalSize> <size>”	Informs that the MQTT client received a message to be continued. **<QoS>** Quality of Service level **<topic>** Topic **<retain>** Retain flag **<duplicate>** Duplicate flag **<totalSize>** Total size of the message **<size>** Size of the payload

Definition	
"info mqttc recvmsg <QoS> <topic> <retain> <duplicate> <totalSize> <size> OK"	Informs that the MQTT client received the last chunk of a message. **<QoS>** Quality of Service level **<topic>** Topic **<retain>** Retain flag **<duplicate>** Duplicate flag **<totalSize>** Total size of the message **<size>** Size of the payload
"info mqttc recvmsg ERROR"	Informs that the MQTT client discovered an error receiving a message.
"info mqttc internal ERROR"	Informs that the MQTT client discovered an internal error.
"info mqttc disconnect OK"	Informs that the MQTT client successfully disconnected from a broker.

Note: If a command could not be sent, because the CMD-UART is blocked it will be tried again until timeout. Some commands will be executed in any case, also when the command could not been sent using the CMD-UART. For example the 'system restart' command and the 'system factory'-command.

4.8.3 Additional User Rights

With the command ⇨ 4.8.5.4 User Rights via UART (variable: auth_state) an application can decide to send the current user rights using a CMD UART command.

A '#' and the user rights will be added in front of the data string separated by a space from the original data.

Example:

Original data (binary)

```
54 65 73 74
```

Original data (ASCII)

```
Test
```

Added user rights to the data (binary)

```
23 66 66 20 54 65 73 74
```

Added user rights to the data (ASCII)

```
#ff Test
```

4.8.4 Supported Baud Rates

The following baud rates are supported:

300, 600, 1200, 2400, 4800, 9600, 14400, 19200, 28800, 38400, 56000, 57600, 115200, 128000, 256000, 520000, 780000, 1500000.

4.8.5 API-Commands (module: CMD UART)

4.8.5.1 Write Data

Definition		
Command option:	Write	
Parameter:	**<data>**	Data to write on the CMD UART.
	Default value	-
	Range	ASCII-encoded string with a maximum of 380 characters.
Description:	The data is sent over the UART. This command is only available for a HTTP/JSON communication.	
SET-Rights:	0x08 or 0x10	

Examples	
HTTP/JSON Write-Request	`[„write“,“cmduart“,“1234567890“]`
HTTP/JSON Write-Response	`[„write“,“cmduart“,“0“,“1234567890“]`

4.8.5.2 Read Data

Definition		
Command option:	Read	
Parameter:	**<data>**	Data which is sent through the UART to request something.
	Default value	-
	Range	-
Description:	The data is sent via the UART. This command is only available for a HTTP/JSON communication.	
GET-Rights:	0x08 or 0x10	

Examples	
HTTP/JSON Read-Request	`[“read”,”cmduart”,””temp”]`
HTTP/JSON Read-Response	`[“read”,”cmduart”,”0”,”temp=100”]`

4.8.5.3 Configuration (variable: cfg)

Definition		
Command option:	Set / Get	
Parameter 1:	**<baud_rate>**	Baud rate for the UART interface.
	Default value	115200
	Range	See ⇨ 4.8.4 Supported Baud Rates.
Parameter 2:	**<data_bits>**	Data bits for the UART. This parameter cannot be set, it will always be 8.
	Default value	8
	Range	[5…8]
Parameter 3:	**<parity>**	Parity bit for the UART. This parameter cannot be set, it will always be 0.
	Default value	0
	Range	[0…4]
Parameter 4:	**<stop_bit>**	Stop bits for the UART. This parameter cannot be set, it will always be 1.
	Default value	1
	Range	[1-2]
Parameter 5	**<hw_handshake>**	Hardware handshake enabled or disabled. For the command UART the hardware handshake will be always disabled.
	Default	0
Description:	Get or set the given UART configuration (CMD UART / BIN UART).	
GET-Rights:	0x01	
SET-Rights:	0x04	

Examples	
HTTP/JSON Get-Request	`["get","cmduart","cfg"]`
HTTP/JSON Get-Response	`["get","cmduart","cfg","0","115200","8","0","1","0"]`
HTTP/JSON Set-Request	`["set","cmduart","cfg","115200","8","0","1","0"]`
HTTP/JSON Set-Response	`["set","cmduart","cfg","0","115200","8","0","1","0"]`
CMD-UART Get-Request	`get cmduart cfg\x0d\x0a`
CMD-UART Get-Response	`get cmduart cfg 0 115200 8 0 1 0\x0d\x0a`
CMD-UART Set-Request	`set cmduart cfg 115200 8 0 1 0\x0d\x0a`
CMD-UART Set-Response	`set cmduart cfg 0 115200 8 0 1 0\x0d\x0a`

4.8.5.4 User Rights via UART (variable: auth_state)

Definition		
Command option:	Set / Get	
Parameter 1:	**<state>**	Flag for sending the current user rights additionally via UART
	Default value	off
	Range	[on, off]
Description:	If this flag is on the command request the current user rights are additionally sent for the command UART.	
GET-Rights:	0x01	
SET-Rights:	0x80	

Examples	
HTTP/JSON Get-Request	`["get","cmduart","auth_state"]`
HTTP/JSON Get-Response	`["get","cmduart","auth_state","0","off"]`
HTTP/JSON Set-Request	`["set","cmduart","auth_state","off"]`
HTTP/JSON Set-Response	`["set","cmduart","auth_state","0","off"]`
CMD-UART Get-Request	`get cmduart auth_state\x0d\x0a`
CMD-UART Get-Response	`get cmduart auth_state 0 off\x0d\x0a`
CMD-UART Set-Request	`set cmduart auth_state off\x0d\x0a`
CMD-UART Set-Response	`set cmduart auth_state 0 off\x0d\x0a`

4.9 GPIO

4.9.1 Pins

The PAN9420 module offers GPIOs which can be controlled using the command API.

The following table shows all available pins.

GPIO-CMD-Index	HW-Pin Name	HW-Pin Description
4	GPIO4	Digital I/O #4
5	GPIO5	Digital I/O #5
46	GPIO46	Digital I/O #46
47	GPIO47	Digital I/O #47
48	GPIO48	Digital I/O #48
49	GPIO49	Digital I/O #49

4.9.2 API-Commands (module: gpio)

4.9.2.1 Mode (variable: mode)

Definition		
Command option:	Get / Set	
Parameter 1:	**<pin#>**	A pin of the GPIO.
	Default value	-
	Range	See list ⇨ 4.9.1 Pins.
Parameter 2:	**<mode>**	Mode of a GPIO pin.
	Default value	out
	Range	[in, out] in: input out: output
Description:	Gets/Sets the mode of a specified GPIO pin.	
GET-Rights:	0x01	
SET-Rights:	0x04	

Examples	
HTTP/JSON Get-Request	`["get","gpio","mode","4"]`
HTTP/JSON Get-Response	`["get","gpio","mode","0","4","in"]`
HTTP/JSON Set-Request	`["set","gpio","mode","4","in"]`
HTTP/JSON Set-Response	`["set","gpio","mode","0","4","in"]`
CMD-UART Get-Request	`get gpio mode 4\x0d\x0a`
CMD-UART Get-Response	`get gpio mode 0 4 in\x0d\x0a`
CMD-UART Set-Request	`set gpio mode 4 in\x0d\x0a`
CMD-UART Set-Response	`set gpio mode 0 4 in\x0d\x0a`

4.9.2.2 State (variable: state)

Definition		
Command option:	Get / Set	
Parameter 1:	**<pin#>**	A pin of the GPIO.
	Default value	-
	Range	See list ⇨ 4.9.1 Pins.
Parameter 2:	**<state>**	Enables/Disables a GPIO pin.
	Default value	off
	Range	[on, off] on: Enabled off: Disabled
Description:	Enables/Disables a specified GPIO pin.	
GET-Rights:	0x01	
SET-Rights:	0x04	

Examples	
HTTP/JSON Get-Request	`["get","gpio","state","4"]`
HTTP/JSON Get-Response	`["get","gpio","state","0","4","off"]`
HTTP/JSON Set-Request	`["set","gpio","state","4","off"]`
HTTP/JSON Set-Response	`["set","gpio","state","0","4","off"]`
CMD-UART Get-Request	`get gpio state 4\x0d\x0a`
CMD-UART Get-Response	`get gpio state 0 4 off\x0d\x0a`
CMD-UART Set-Request	`set gpio state 4 off\x0d\x0a`
CMD-UART Set-Response	`set gpio state 0 4 off\x0d\x0a`

4.10 HTTP Client

4.10.1 HTTP Client Status Values

Status Code	Description
0	Not configured
1	Ready to send data
2	Data is sending
3	Error

4.10.2 API-Commands (module: httpc)

4.10.2.1 State (variable: state)

Definition		
Command option:	Get / Set	
Parameter 1:	**<state>**	HTTP client state.
	Default value	off
	Range	[off, on]
Description:	Set and get the state of the HTTP client.	
GET-Rights:	0x01	
SET-Rights	0x04	

Examples	
HTTP/JSON Get-Request	`["get","httpc","state"]`
HTTP/JSON Get-Response	`["get","httpc","state","0","off"]`
HTTP/JSON Set-Request	`["set","httpc","state",on"]`
HTTP/JSON Set-Response	`["set","httpc","state","0","on"]`
CMD-UART Get-Request	`get httpc state\x0d\x0a`
CMD-UART Get-Response	`get httpc state 0 off\x0d\x0a`
CMD-UART Set-Request	`set httpc state on\x0d\x0a`
CMD-UART Set-Response	`set httpc state 0 on\x0d\x0a`

4.10.2.2 Status (variable: status)

Definition		
Command option:	Get	
Parameter 1:	**<status>**	HTTP client status.
	Default value	0
	Range	See ⇨ 1.1 HTTP Client Status Values.
Description:	Returns the status of the HTTP client.	
GET-Rights:	0x01	

Examples	
HTTP/JSON Get-Request	`["get","httpc","status"]`
HTTP/JSON Get-Response	`["get","httpc","status","0","1"]`
CMD-UART Get-Request	`get httpc status\x0d\x0a`
CMD-UART Get-Response	`get httpc status 0 1\x0d\x0a`

4.10.2.3 HTTP Server (variable: server)

Definition		
Command option:	Get / Set	
Parameter 1:	**<server-adr>**	HTTP-Server address.
	Default value	-
	Range	String with maximum of 40 characters (underscores not allowed).
Parameter 2:	**<port>**	HTTP-Server port number.
	Default value	-
	Range	[0…65535]
Parameter 3:	**<security>**	HTTP-Server security.
	Default value	auto
	Range	[off, on, auto]
Parameter 4:	**<username>**	HTTP-Server username.
	Default value	-
	Range	String with maximum of 30 characters.
Parameter 5:	**<password>**	HTTP-Server password.
	Default value	-
	Range	String with maximum of 64 characters.
Description:	Configuration of the HTTP server for the HTTP client.	
GET-Rights:	0x01	
SET-Rights:	0x04	

Examples	
HTTP/JSON Get-Request	`["get","httpc","server"]`
HTTP/JSON Get-Response	`["get","httpc","server","0","ServerAdr","80","auto","username"]`
HTTP/JSON Set-Request	`["set","httpc","server","ServerAdr","80","auto","username","password"]`
HTTP/JSON Set-Response	`["set","httpc","server","0","ServerAdr","80","auto","username","password"]`
CMD-UART Get-Request	`get httpc server\x0d\x0a`
CMD-UART Get-Response	`get httpc server 0 ServerAdr 80 auto username\x0d\x0a`
CMD-UART Set-Request	`set httpc server ServerAdr 80 auto username password\x0d\x0a`
CMD-UART Set-Response	`set httpc server 0 ServerAdr 80 auto username password\x0d\x0a`

4.10.2.4 HTTP Server Resource (variable: resource)

Definition		
Command option:	Get / Set	
Parameter 1:	**<resource>**	HTTP-Server resource.
	Default value	-
	Range	String with maximum of 100 characters.
Description:	Configuration of the HTTP server resource for the HTTP client.	
GET-Rights:	0x01	
SET-Rights:	0x04	

Examples	
HTTP/JSON Get-Request	`["get","httpc","resource"]`
HTTP/JSON Get-Response	`["get","httpc","resource","0","filename"]`
HTTP/JSON Set-Request	`["set","httpc","resource","filename"]`
HTTP/JSON Set-Response	`["set","httpc","resource","0","filename"]`
CMD-UART Get-Request	`get httpc resource\x0d\x0a`
CMD-UART Get-Response	`get httpc resource 0 filename\x0d\x0a`
CMD-UART Set-Request	`set httpc resource filename\x0d\x0a`
CMD-UART Set-Response	`set httpc resource 0 filename\x0d\x0a`

4.10.2.5 POST (variable: post)

Definition		
Command option:	Set	
Parameter 1:	**<data>**	Data which should be sent.
	Default value	-
	Range	String with a maximum of 300 characters.
Description:	Data which should be sent to the server. To Send the data a valid server must be configured. Also a connection to this server must be available.	
GET-Rights:	0x01	
SET-Rights:	0x04	

Examples	
HTTP/JSON Set-Request	`["set","httpc","post","data"]`
HTTP/JSON Set-Response	`["set","httpc","post","0","data"]`
CMD-UART Set-Request	`set httpc post data\x0d\x0a`
CMD-UART Set-Response	`set httpc post 0 data\x0d\x0a`

4.10.2.6 GET (variable: get)

Definition		
Command option:	Get / Set	
Parameter 1:	**<data>**	Data which should be sent. Not used with ‘set’ command.
	Default value	-
	Range	String with maximum of 299 characters.
Description:	Data which should be sent to the server	
GET-Rights:	0x01	
SET-Rights:	0x04	

Examples	
HTTP/JSON Get-Request	`["get","httpc","get"]`
HTTP/JSON Get-Response	`["get","httpc","get","0","data"]`
HTTP/JSON Set-Request	`["set","httpc","get"]`
HTTP/JSON Set-Response	`["set","httpc","get","0"]`
CMD-UART Get-Request	`get httpc get\x0d\x0a`
CMD-UART Get-Response	`get httpc get 0\x0d\x0a`
CMD-UART Set-Request	`set httpc get\x0d\x0a`
CMD-UART Set-Response	`set httpc get 0\x0d\x0a`

4.10.2.7 TOKEN (variable: token)

Definition		
Command option:	Get / Set	
Parameter 1:	**<token>**	Server token for the client.
	Default value	-
	Range	String with maximum of 200 characters.
Description:	Token from the server for the client.	
GET-Rights:	0x01	
SET-Rights:	0x04	

Examples	
HTTP/JSON Get-Request	`["get","httpc","token"]`
HTTP/JSON Get-Response	`["get","httpc","token","0","12345abc"]`
HTTP/JSON Set-Request	`["set","httpc","token","12345abc"]`
HTTP/JSON Set-Response	`["set","httpc","token","0","12345abc"]`
CMD-UART Get-Request	`get httpc token\x0d\x0a`
CMD-UART Get-Response	`get httpc token 0 12345abc\x0d\x0a`
CMD-UART Set-Request	`set httpc token 12345abc\x0d\x0a`
CMD-UART Set-Response	`set httpc token 0 12345abc\x0d\x0a`

4.10.2.8 Sending Data via BIN UART

Definition		
Command option:	Send	
Parameter 1:	**<request_type>** **Range**	POST or GET type. [post, get]
Parameter 1:	**<data_size>**	Number of bytes expected to receive over the BIN UART that should be sent as HTTP payload.
Description:	Trigger a HTTP Request and waiting HTTP payload on the BIN UART.	
GET-Rights:	0x01	
SET-Rights:	0x04	

Examples	
HTTP/JSON Set-Request	`["send","httpc","post","500"]`
HTTP/JSON Set-Response	`["send","httpc","post","0","500"]`
CMD-UART Set-Request	`send httpc post 500\x0d\x0a`
CMD-UART Set-Response	`send httpc post 0 data\x0d\x0a`

4.11 Netcat

4.11.1 Netcat Status Values

Status Code	Description
0	Not configured
1	Ready to send data
2	Data is sending
3	Error
4	Connecting

4.11.2 API-Commands (module: netcat)

4.11.2.1 State (variable: state)

Definition		
Command option:	Get / Set	
Parameter 1:	**<state>**	Starts or stops a TCP-Listen or a TCP-Connect.
	Default value	off
	Range	[on, off]
Description:	The variable can start or stop a TCP-Listen or a TCP-Connect depending on the mode parameter.	
GET-Rights:	0x01	
SET-Rights:	0x04	

Examples	
HTTP/JSON Get-Request	`["get","netcat","state"]`
HTTP/JSON Get-Response	`["get","netcat","state","0","on"]`
HTTP/JSON Set-Request	`["set","netcat","state","off"]`
HTTP/JSON Set-Response	`["set","netcat","state","0","off"]`
CMD-UART Get-Request	`get netcat state\x0d\x0a`
CMD-UART Get-Response	`get netcat state 0 on\x0d\x0a`
CMD-UART Set-Request	`set netcat state off\x0d\x0a`
CMD-UART Set-Response	`set netcat state 0 off\x0d\x0a`

4.11.2.2 Configuration (variable: cfg)

Definition		
Command option:	Get / Set	
Parameter 1:	**<mode>**	The mode can be set to “client” or “server”. In the client mode the Netcat module is connecting to a server. In the server mode the Netcat module is the server and a client can connect to establish a connection.
	Default value	client
	Range	[client, server]
Parameter 2:	**<port>**	TCP port for a TCP-Listen or a TCP-Connect.
	Default value	-
	Range	[1…65535]
Parameter 3:	**<server-ip>**	Server IP address.
	Default value	-
	Range	-
Parameter 4:	**<username>**	Username on the server for the login.
	Default value	-
	Range	String with maximum of 30 characters.
Parameter 5:	**<password>**	Password on the server for the login. The password will not be returned with a get request.
	Default value	-
	Range	String with a maximum of 30 characters.
Description:	Configuration of the Netcat module. Setting the mode to client requires all parameters to establish a connection to a server. Setting the mode to server requires only the port parameter in addition.	
GET-Rights:	0x01	
SET-Rights:	0x04	

Examples	
HTTP/JSON Get-Request	`["get","netcat","cfg"]`
HTTP/JSON Get-Response	`["get","netcat","cfg","0","client","1234","88.77.66.55","username"]`
HTTP/JSON Set-Request	`["set","netcat","cfg","client","1234","88.77.66.55","username","password"]`
HTTP/JSON Set-Response	`["set","netcat","cfg","0","client","1234","88.77.66.55","username","password"]`
CMD-UART Get-Request	`get netcat cfg\x0d\x0a`
CMD-UART Get-Response	`get netcat cfg 0 server 5555\x0d\x0a`
CMD-UART Set-Request	`set netcat cfg server 5555\x0d\x0a`
CMD-UART Set-Response	`set netcat cfg 0 server 5555\x0d\x0a`

4.11.2.3 Telnet Option (variable: telopt)

Definition		
Command option:	Set / Get	
Parameter 1:	**<state>**	The state of the Netcat telnet feature.
	Default value	off
	Range	[on, off]
Description:	Enables or disables the telnet option for Netcat.	
GET-Rights:	0x01	
SET-Rights:	0x04	

Examples	
HTTP/JSON Get-Request	`["get","netcat","telopt"]`
HTTP/JSON Get-Response	`["get","netcat","telopt","0","on"]`
HTTP/JSON Set-Request	`["set","netcat","telopt","off"]`
HTTP/JSON Set-Response	`["set","netcat","telopt,"0","off"]`
CMD-UART Get-Request	`get netcat telopt\x0d\x0a`
CMD-UART Get-Response	`get netcat telopt 0 on\x0d\x0a`
CMD-UART Set-Request	`set netcat telopt off\x0d\x0a`
CMD-UART Set-Response	`set netcat telopt 0 off\x0d\x0a`

4.11.2.4 Authentication (variable: auth)

Definition		
Command option:	Set / Get	
Parameter 1:	**<auth>**	The state of the Netcat authentication feature.
	Default value	on
	Range	[on, off]
Description:	Enables or disables the authentication for Netcat.	
GET-Rights:	0x01	
SET-Rights:	0x80	

Examples	
HTTP/JSON Get-Request	`["get","netcat","auth"]`
HTTP/JSON Get-Response	`["get","netcat","auth","0","on"]`
HTTP/JSON Set-Request	`["set","netcat","auth","off"]`
HTTP/JSON Set-Response	`["set","netcat","auth,"0","off"]`
CMD-UART Get-Request	`get netcat auth\x0d\x0a`
CMD-UART Get-Response	`get netcat auth 0 on\x0d\x0a`
CMD-UART Set-Request	`set netcat auth off\x0d\x0a`
CMD-UART Set-Response	`set netcat auth 0 off\x0d\x0a`

4.11.2.5 Status (variable: status)

Definition		
Command option:	Get	
Parameter 1:	**<status>**	Netcat status.
	Default value	0
	Range	See list ⇨ 4.11.1 Netcat Status Values.
Description:	Returns the status of the Netcat service.	
GET-Rights:	0x01	

Examples	
HTTP/JSON Get-Request	`["get","netcat","status"]`
HTTP/JSON Get-Response	`["get","netcat","status","0","1"]`
CMD-UART Get-Request	`get netcat status\x0d\x0a`
CMD-UART Get-Response	`get netcat status 0 1\x0d\x0a`

4.12 Binary UART

The binary UART (BIN UART) is used to transfer binary application data between the module and the host controller. Naturally this data is not parsed. The active service module will process or forward it. As the BIN UART can be used for different service applications (Netcat, Webcat, UDP Messaging, etc.) the BIN UART will be locked for one application. E.g. if Netcat is active, then no UDP data of the UDP-Messaging Service will be forwarded over the BIN UART.

Note: Some internal modes such as the Netcat telnet feature might modify the data.

Features:

- Can share its FIFO with the service application.
- Beside the common send and receive API, it provides an extended internal API that gives direct access to the FIFO. With this service, applications can spare separate application buffers. This can lower the RAM requirements.

- Simple management functions to share the binary UART between services (Lock IDs).
- Telnet option features. UART configuration by parsing binary command string (byte stuffing).

4.12.1 BIN UART Lock Handling

The BIN UART can be used by different BIN UART service applications. To avoid mixed up data streams the BIN UART must be locked for a given application. For this a service application requests a lock at the BIN UART which then allows commands and data transfer for the locked application. Other requests will be rejected. If the locked application is finished it should release the lock by turning off the application. Otherwise other applications will never get access to the BIN UART.

The BIN UART provides a lock command, which the user can use to get information about the current active service application.

4.12.2 BIN UART Command Overview

Command	Description
cfg	Configure the physical UART parameters.
Read	Read data from the RX FIFO.
Write	Write some data on the TX FIFO.
Clear	Clear the buffers.
Lock	Lock the BIN UART.
Dtrpin	Set the DTR PIN.
Uart_port	Configure the physical UART port index.

4.12.3 Sending Data Using the BIN UART.

Basic steps to transfer data via XBAR (from an external application to the BIN UART)

1. Get a Lock
2. Read and write with an external module (HTTP/Ajax, CMD UART, etc.)
3. Release the Lock

4.12.4 GPIOs for Single UART Mode

The UART 1 is used for sending and receiving data for BIN and CMD mode. So the pins are configured as in the normal BIN UART mode. Additional to these pins the old UART 0 pins are used to change the mode (BIN / CMD) and to read the actual state of the UART mode.

GPIO	I/O	Description
74 (UART0_TX)	Output	Shows actual mode: high: BIN mode low: CMD mode
75 (UART0_RX)	Input	Toggle pin from low to high to change the mode.

Note: The pin *74 has* to been checked to verify that the mode already changed.

4.12.5 Supported Baud Rates

The following baud rates are supported:

300, 600, 1200, 2400, 4800, 9600, 14400, 19200, 28800, 38400, 56000, 57600, 115200, 128000, 256000, 520000, 780000, 1500000.

4.12.6 API-Commands (Module: binuart)

4.12.6.1 UART Configuration (variable: cfg)

Definition		
Command option:	Set / Get	
Parameter 1:	**<lock ID>**	Lock ID.
	Default value	-
	Range	[0…65535]
Parameter 2:	**<baudrate>**	Baud rate for the UART.
	Default value	115200
	Range	See list ⇨ 4.12.5 Supported Baud Rates.
Parameter 3:	**<data_bits>**	Data bits for the UART. This parameter cannot be set, it will always be 8.
	Default value	8
	Range	[5…8]
Parameter 4:	**<parity>**	Parity bit for the UART. This parameter cannot be set, it will always be 0.
	Default value	0
	Range	[0…4] see list 3.10.1.2 Parity types
Parameter 5:	**<stopbit>**	Stop bits for the UART. This parameter cannot be set, it will always be 1.
	Default value	1
	Range	[1,2]
Parameter 6:	**<hwhandshake>**	Hardware handshake enabled or disabled.
	Default :	1
	Range	[0,1]
Description:	UART configuration for the Netcat module.	
GET-Rights:	0x01	
SET-Rights:	0x04	

Examples	
HTTP/JSON Get-Request	`["get","binuart","cfg","12345"]`
HTTP/JSON Get-Response	`["get","binuart","cfg","0","12345","115200","8","0","1","1"]`
HTTP/JSON Set-Request	`["set","binuart","cfg","12345","115200","8","0","1","1"]`
HTTP/JSON Set-Response	`["set","binuart","cfg","0","12345","115200","8","0","1","1"]`
CMD-UART Get-Request	`get binuart cfg 12345\x0d\x0a`
CMD-UART Get-Response	`get binuart cfg 0 12345 115200 8 0 1 1\x0d\x0a`
CMD-UART Set-Request	`set binuart cfg 12345 115200 8 0 1 1\x0d\x0a`
CMD-UART Set-Response	`set binuart cfg 0 12345 115200 8 0 1 1\x0d\x0a`

4.12.6.2 UART Shared (variable: shared)

Definition		
Command option:	Set / Get	
Parameter 1:	**<lock ID>**	Lock ID.
	Default value	-
	Range	[0…65535]
Parameter 2:	**<state>**	State for the shared operation.
	Default value	off
	Range	[on, off]
Parameter 3:	**<mode>**	Mode for the UART, only used with ‘get’ and if state is ‘on’
	Default value	binary (bin)
	Range	binary (BIN) or command (CMD)
Description:	The shared operation is for sharing one UART for binary commands and ASCII-encoded string commands. The parameter mode is only returned using a get-request. For setting the shared operation “on” an optional lock ID and the state are required.	
GET-Rights:	0x01	
SET-Rights:	0x04	

Examples	
HTTP/JSON Get-Request	`["get","binuart","shared"]`
HTTP/JSON Get-Response	`["get","binuart","shared","0","on","bin"]`
HTTP/JSON Set-Request	`["set","binuart","shared","on"]`
HTTP/JSON Set-Response	`["set","binuart","shared,"0","on"]`
CMD-UART Get-Request	`get binuart shared\x0d\x0a`
CMD-UART Get-Response	`get binuart shared 0 on bin\x0d\x0a`
CMD-UART Set-Request	`set binuart shared on\x0d\x0a`
CMD-UART Set-Response	`set binuart shared 0 on\x0d\x0a`

4.12.6.3 Request a Lock (variable: lock)

Definition		
Command option:	Set	
Parameter 1:	**<state>**	State for the lock.
	Default value	off
	Range	[on, off]
Parameter 2:	**<random number>**	Lock ID for the exclusive lock.
	Default value	-
	Range	[0…65535]
Description:	To lock the communication a user can request a lock ID. The lock ID must then be parsed for every operation with the BIN UART.	
GET-Rights:	0x04	

Examples	
HTTP/JSON Set-Request	`["set","binuart","lock","on"]`
HTTP/JSON Set-Response	`["set","binuart","lock","0","on","15252"]`
CMD-UART Set-Request	`set binuart lock on\x0d\x0a`
CMD-UART Set-Response	`set binuart lock 0 on 15252\x0d\x0a`

4.12.6.4 Who got the Lock (variable: lock)

Definition		
Command option:	Get	
Parameter 1:	**<module-string>**	ASCII string of the module.
	Default value	-
	Range	[CMDIF, NETCAT, UDP, WEBCAT, HTTPC]
Description:	Returns the current application which got the lock for the BIN UART.	
SET-Rights:	0x01	

Examples	
HTTP/JSON Get-Request	`["get","binuart","lock"]`
HTTP/JSON Get-Response	`["get","binuart","lock","0","netcat"]`
CMD-UART Get-Request	`get binuart lock\x0d\x0a`
CMD-UART Get-Response	`get binuart lock 0 netcat\x0d\x0a`

4.12.6.5 Get a Host Controller Lock (variable: lock)

Definition		
Command option:	Set	
Parameter 1:	**<lock_state>**	The state of the lock state of host controller.
	Default value	-
	Range	[on, off]
Description:	Locks the BIN UART for the host controller. If no data is transferred, then it will be released automatically. (20 seconds). A Lock ID is returned. It must be used for further BIN UART requests (read, write).	
SET-Rights:	0x01	

Examples	
HTTP/JSON Get-Request	`["set","binuart","lock", "on"]`
HTTP/JSON Get-Response	`["set","binuart","lock","0","1234"]`
CMD-UART Get-Request	`set binuart lock\x0d\x0a`
CMD-UART Get-Response	`set binuart lock 0 1234\x0d\x0a`

4.12.6.6 DTR-Pin (variable: dtrpin)

Definition		
Command option:	Set / Get	
Parameter 1:	**<lock ID>**	Lock ID.
	Default value	-
	Range	[0…65535]
Parameter 2:	**<pin>**	DTR pin for the BIN UART.
	Default value	-
	Range	Available pins see list ⇨ 4.9.1 Pins.
Description:	Set the DTR pin for the BIN UART. The DTR pin is a pin which can be configured via telnet-option. The DTR pin can be set to high or low via a telnet command.	
GET-Rights:	0x01	
SET-Rights:	0x04	

Examples	
HTTP/JSON Get-Request	`["get","binuart","dtrpin"]`
HTTP/JSON Get-Response	`["get","binuart","dtrpin","0","4"]`
HTTP/JSON Set-Request	`["set","binuart","dtrpin","4"]`
HTTP/JSON Set-Response	`["set","binuart","dtrpin,"0","4"]`
CMD-UART Get-Request	`get binuart dtrpin\x0d\x0a`
CMD-UART Get-Response	`get binuart dtrpin 0 4\x0d\x0a`
CMD-UART Set-Request	`set binuart dtrpin 4\x0d\x0a`
CMD-UART Set-Response	`set binuart dtrpin 0 4\x0d\x0a`

4.12.6.7 Clear (variable: clear)

Definition		
Command option:	Set	
Parameter 1:	**<lock ID>**	Lock ID.
	Default value	-
	Range	[0…65535]
Description:	Removes all data from the BIN UART buffer.	
SET-Rights:	0x04	

Examples	
HTTP/JSON Get-Request	`["set","binuart","clear","0"]`
HTTP/JSON Get-Response	`["set","binuart","clear","0","0"]`
CMD-UART Get-Request	`set binuart clear 0\x0d\x0a`
CMD-UART Get-Response	`set binuart clear 0 0\x0d\x0a`

4.12.6.8 UART port (variable: uart_port)

Definition		
Command option:	Get	
Parameter 1:	**<uart_port>**	Physical BIN UART port.
	Default value	-
	Range	[0…255]
Description:	Returns the physical UART port for the BIN UART.	
GET-Rights:	0x01	

Examples	
HTTP/JSON Get-Request	`["get","binuart","uart_port"]`
HTTP/JSON Get-Response	`["get","binuart","uart_port","0","1"]`
CMD-UART Get-Request	`get binuart uart_port\x0d\x0a`
CMD-UART Get-Response	`get binuart uart_port 0 1\x0d\x0a`

4.12.6.9 Read Data

Definition		
Command option:	Read	
Parameter 1:	**<lock ID>**	Lock ID.
	Default value	-
	Range	[0…65535]
Parameter 2:	**<data>**	Data which is received.
	Default value	-
	Range	Maximum 150 bytes (in ASCII format).
Description:	Reads the binary UART. A lock has to be requested using the lock variable. Note: The data has to be in ASCII format on the binary UART and will reach the command UART in HEX format.	
READ-Rights:	0x20	

Examples	
HTTP/JSON Read-Request	`["read","binuart","1234"]`
HTTP/JSON Read-Response	`["read","binuart","0","1234","48656C6C6F"]`
CMD-UART Read-Request	`read binuart 1234\x0d\x0a`
CMD-UART Read-Response	`read binuart 0 1234 48656C6C6F\x0d\x0a`

4.12.6.10 Write Data

Definition		
Command option:	Write	
Parameter 1:	**<lock ID>**	Handle ID.
	Default value	-
	Range	[0…65535]
Parameter 2:	**<data>**	Data to write.
	Default value	-
	Range	Maximum 300 characters (HEX Stream).
Description:	Write data to the UART. A lock has to be requested using the lock variable. Note: The data has to be a HEX stream. This stream will reach the binary UART in ASCII format.	
WRITE-Rights:	0x20	

Examples	
HTTP/JSON Write-Request	`["write","binuart","1234"," 48656C6C6F"]`
HTTP/JSON Write-Response	`["write","binuart","1234","48656C6C6F"]`
CMD-UART Write-Request	`write binuart 1234 48656C6C6F\x0d\x0a`
CMD-UART Write-Response	`write binuart 0 1234 48656C6C6F\x0d\x0a`

4.13 Time

4.13.1 API-Commands (Module: time)

4.13.1.1 UTC – GET (variable: utc)

Definition		
Command option:	Get	
Parameter 1:	**<utc>**	Date and time of the UTC.
	Default value	-
	Range	UTC
Description:	UTC with the ISO 8601 time format: Year-Month-DayTHour:Minute:Seconds 2015-08-31T11:12:13 To get a valid time, the module needs an internet connection.	
GET-Rights:	0x00	

Examples	
HTTP/JSON Get-Request	`["get","time","utc"]`
HTTP/JSON Get-Response	`["get","time","utc","0","2015-08-31T11:12:13"]`
CMD-UART Get-Request	`get time utc\x0d\x0a`
CMD-UART Get-Response	`get time utc 0 2015-08-31T11:12:13\x0d\x0a`

4.13.1.2 UTC – SET (variable: utc)

Definition	
Command option:	Set
Parameter 1:	**<year>** - Year for new date/time. **Default value** - **Range** [0…65535]
Parameter 2:	**<month>** - Month for new date/time. **Default value** - **Range** [1...12]
Parameter 3:	**<day>** - Day for new date/time. **Default value** - **Range** [1…31]
Parameter 4:	**<hour>** - Hour for new date/time. **Default value** - **Range** [0…23]
Parameter 5:	**<min>** - Minutes for new date/time. **Default value** - **Range** [0…59]
Parameter 6:	**<sec>** - Seconds for new date/time. **Default value** - **Range** [0…59]
Description:	Sets a new internal date and time.
SET-Rights:	0x02

Examples	
HTTP/JSON Set-Request	`["set","time","utc","2016","3","20","12","30","12"]`
HTTP/JSON Set-Response	`["set","time","utc","0","2016","3","20","12","30","12"]`
CMD-UART Set-Request	`set time utc 2016 3 20 12 30 12\x0d\x0a`
CMD-UART Set-Response	`set time utc 0 2016 3 20 12 30 12\x0d\x0a`

4.13.1.3 Time Sync (variable: sync)

Definition		
Command option:	Set / Get	
Parameter 1:	**<sync>**	Timeout value for NTP time update.
	Default value	3600 seconds
	Range	0 – no NTP update Minimum: 3600 seconds Maximum: 4294967294 seconds
Description:	Setting the timeout for NTP updates. Setting the value to 0 results in no updates for the time.	
GET-Rights:	0x01	
SET-Rights:	0x02	

Examples	
HTTP/JSON Get-Request	`["get","time","sync"]`
HTTP/JSON Get-Response	`["get","time","sync","0","3600"]`
HTTP/JSON Set-Request	`["set","time","sync","3600"]`
HTTP/JSON Set-Response	`["set","time","sync","0","3600"]`
CMD-UART Get-Request	`get time sync\x0d\x0a`
CMD-UART Get-Response	`get time sync 0 3600\x0d\x0a`
CMD-UART Set-Request	`set time sync 3600\x0d\x0a`
CMD-UART Set-Response	`set time sync 0 3600\x0d\x0a`

4.13.1.4 Mode (variable: mode)

Definition		
Command option:	Set / Get	
Parameter 1:	**<mode>**	Mode for the time module
	Default value	auto
	Range	[auto, manual]
Description:	The time module can get the time automatically from a configured time server or get the time manually.	
GET-Rights:	0x00	
SET-Rights:	0x02	

Examples	
HTTP/JSON Get-Request	`["get","time","mode"]`
HTTP/JSON Get-Response	`["get","time","mode","0","auto"]`
HTTP/JSON Set-Request	`["set","time","mode","manual"]`
HTTP/JSON Set-Response	`["set","time","mode","0","manual"]`
CMD-UART Get-Request	`get time mode\x0d\x0a`
CMD-UART Get-Response	`get time mode 0 auto\x0d\x0a`
CMD-UART Set-Request	`set time mode manual\x0d\x0a`
CMD-UART Set-Response	`set time mode 0 manual\x0d\x0a`

4.13.1.5 Server (variable: server)

Definition		
Command option:	Set / Get	
Parameter 1:	**<index>**	Index of the server.
	Default value	-
	Range	[0,1]
Parameter 2:	**<server>**	Server address
	Default value	-
	Range	-
Description:	This is the server for the automatic time update. The server address can be an IP-address or a domain name.	
GET-Rights:	0x00	
SET-Rights:	0x02	

Examples	
HTTP/JSON Get-Request	`["get","time","server","0"]`
HTTP/JSON Get-Response	`["get","time","server","0","0","ntp.time.de"]`
HTTP/JSON Set-Request	`["set","time","server","0","ntp.time.de"]`
HTTP/JSON Set-Response	`["set","time","server","0","0","ntp.time.de"]`
CMD-UART Get-Request	`get time server 0\x0d\x0a`
CMD-UART Get-Response	`get time server 0 0 ntp.time.de\x0d\x0a`
CMD-UART Set-Request	`set time server 0 ntp.time.de\x0d\x0a`
CMD-UART Set-Response	`set time server 0 0 ntp.time.de\x0d\x0a`

4.14 UDP Messaging

4.14.1 UDP Status Values

Status Code	Description
0	Not configured
1	Ready to send data
2	Data is sending
3	Error
4	Connecting

4.14.2 API-Commands (Module: udp)

4.14.2.1 State (variable: state)

Definition		
Command option:	Set / Get	
Parameter 1:	**<state>**	The state of the UDP messaging module
	Default value	off
	Range	[on, off]
Description:	This is the current state of the UDP messaging module.	
GET-Rights:	0x01	
SET-Rights:	0x04	

Examples	
HTTP/JSON Get-Request	`["get","udp","state"]`
HTTP/JSON Get-Response	`["get","udp","state","0","off"]`
HTTP/JSON Set-Request	`["set","udp","state","on"]`
HTTP/JSON Set-Response	`["set","udp","state","0","off"]`
CMD-UART Get-Request	`get udp state\x0d\x0a`
CMD-UART Get-Response	`get udp state 0 off\x0d\x0a`
CMD-UART Set-Request	`set udp state on\x0d\x0a`
CMD-UART Set-Response	`set udp state 0 on\x0d\x0a`

4.14.2.2 Host (variable: host)

Definition		
Command option:	Set / Get	
Parameter 1:	**<index>**	Entry in the host table.
	Default value	-
	Range	[0…4]
Parameter 2:	**<ip-addr>** -	Destination IP address.
Parameter 3:	**<port>**	Destination UDP port.
	Range	[0…65535]
Parameter 4:	**<interface>**	Identifier to select the WLAN interface.
	Default	invalid
	Range	[ap, sta] ap: Access-Point sta: Station
Description:	The UDP host table contains the peer host address and UDP port for communication.	
GET-Rights:	0x01	
SET-Rights:	0x04	

Examples	
HTTP/JSON Get-Request	`["get","udp","host","1"]`
HTTP/JSON Get-Response	`["get","udp","host","0","1","191.162.1.120","1234","sta"]`
HTTP/JSON Set-Request	`["set","udp","host","1","191.162.1.120","1234","sta"]`
HTTP/JSON Set-Response	`["set","udp","host","0","1","191.162.1.120","1234","sta"]`
CMD-UART Get-Request	`get udp host 1\x0d\x0a`
CMD-UART Get-Response	`get udp host 0 1 191.162.1.120 1234 sta\x0d\x0a`
CMD-UART Set-Request	`set udp host 1 191.162.1.120 1234 sta\x0d\x0a`
CMD-UART Set-Response	`set udp host 0 1 191.162.1.120 1234 sta\x0d\x0a`

4.14.2.3 Port Configuration (variable: portconfig)

Definition		
Command option:	Set / Get	
Parameter 1:	**<index>**	Entry in the port table
	Default value	-
	Range	[0…4]
Parameter 2:	**<port>** -	Local port.
	Range	[0…65535]
Parameter 3:	**<interface>**	Identifier to select the WLAN interface.
	Default	invalid
	Range	[ap, sta] ap: Access-Point sta: Station
Description:	The UDP port table contains the local ports for the UDP communication.	
GET-Rights:	0x01	
SET-Rights:	0x04	

Examples	
HTTP/JSON Get-Request	`["get","udp","portconfig","1"]`
HTTP/JSON Get-Response	`["get","udp","portconfig","0","1","1234","sta"]`
HTTP/JSON Set-Request	`["set","udp","portconfig","1","1234","sta"]`
HTTP/JSON Set-Response	`["set","udp","portconfig","0","1","1234","sta"]`
CMD-UART Get-Request	`get udp portconfig 1\x0d\x0a`
CMD-UART Get-Response	`get udp portconfig 0 1 1234 sta\x0d\x0a`
CMD-UART Set-Request	`set udp portconfig 1 1234 sta\x0d\x0a`
CMD-UART Set-Response	`set udp portconfig 0 1 1234 0\x0d\x0a`

4.14.2.4 Listen (variable: listen)

Definition	
Command option:	Set
Parameter 1:	**<port table entry>** Entry in the port table. **Default value** - **Range** [0…5]
Parameter 2:	**<host table entry>** Entry in the host table. **Default value** - **Range** [0…5 or all] This parameter is used as a filter parameter. The value 0 means that only packets from the host that is defined in the host table entry 0 will be forwarded through the UART. All other host will be discarded. Using the value “all” will accept UDP packets from any host.
Description:	Opens a UDP instance with the given port table entry and host table entry.
SET-Rights:	0x04

Examples	
HTTP/JSON Set-Request	`["set","udp","listen","1","0"]`
HTTP/JSON Set-Response	`["set","udp","listen","0","1","0"]`
CMD-UART Set-Request	`set udp listen 1 0\x0d\x0a`
CMD-UART Set-Response	`set udp listen 0 1 0\x0d\x0a`

4.14.2.5 Close (variable: close)

Definition	
Command option:	Set
Parameter 1:	**<port table entry>** Entry in the port table. **Default value** - **Range** [0…5]
Description:	Closes the specified UDP port.
SET-Rights:	0x04

Examples	
HTTP/JSON Set-Request	`["set","udp","close","1"]`
HTTP/JSON Set-Response	`["set","udp","close","0","1"]`
CMD-UART Set-Request	`set udp close 1\x0d\x0a`
CMD-UART Set-Response	`set udp close 0 1\x0d\x0a`

4.14.2.6 Send Message (variable: sendmsg)

Definition		
Command option:	Set	
Parameter 1:	**<port table entry>**	Entry in the port table.
	Default value	-
	Range	[0…5]
Parameter 2:	**<host table entry>**	Entry in the host table.
	Default value	-
	Range	[0…5]
Parameter 3:	**<size>**	Data size.
Description:	Sending binary data via UDP.	
SET-Rights:	0x04	

Examples	
HTTP/JSON Set-Request	`["set","udp","sendmsg","1","2","1234"]`
HTTP/JSON Set-Response	`["set","udp","sendmsg","0","1","2","1234"]`
CMD-UART Set-Request	`set udp sendmsg 1 2 1234 0\x0d\x0a`
CMD-UART Set-Response	`set udp sendmsg 0 1 2 1234 0\x0d\x0a`

4.14.2.7 Last Remote (variable: lastremote)

Definition		
Command option:	Get	
Parameter 1:	**<ip address>**	Remote IP address of the last UDP packet.
	Default value	-
Parameter 2:	**<port>**	Remote port of the last UDP packet.
	Default value	-
GET-Rights:	0x01	

Examples	
HTTP/JSON Get-Request	`["get","udp","lastremote"]`
HTTP/JSON Get-Response	`["get","udp","lastremote","0","191.161.1.120","555"]`
CMD-UART Get-Request	`get udp lastremote\x0d\x0a`
CMD-UART Get-Response	`get udp lastremote 0 192.161.1.120 555\x0d\x0a`

4.15 Webcat (websocket-client)

Webcat is a binary UART to Websocket Tunnel-Application. Data over the BIN UART can be transferred to and received from a Websocket Server. Server parameters and switching it on or off is done by defined commands over the CMD UART.

Basic supported commands for Webcat service

- server Basic server parameters (server IP and port)
- serverext Extended server parameters (extended parameters like 'origin')
- state On/off functionality
- status Status information of Webcat (e.g. websocket is in establishing state)

Basic steps for using Webcat

1. Configure the Server parameters
2. Start the Webcat module
3. Do data exchange on the BIN UART
4. Stop the Webcat module

Configure a Websocket Server to connect to

The first step is to define a server that should be used as websocket destination:

```
set webcat server <server IP/domain name> <port> <resource> <login> <password>
```

Server IP and port are mandatory parameters. The other 3 parameters are optional. If these parameters are not needed, then default is "/" for resource. Login is an empty string.

Start the Webcat – UART tunnel connection

The connection to the Server will be started as the module is turned on. This is done with following command:

```
set webcat state on
```

Shortly after executing this command the websocket connection is established. But to be sure that the connection is successful the host controller can check the connection state with:

```
get webcat status
```

Data transfer

If the Websocket connection has been established successfully, then data exchange is done by just sending and receiving data over the BIN UART.

Quit the Websocket Connection

Closing the Websocket connection is done by turning the module off.

```
Set webcat state off
```

4.15.1 Webcat Status Values

Status Code	Description
0	Not configured
1	Ready to send data
2	Data is sending
3	Error
4	Connecting

4.15.2 API-Commands (Module: webcat)

4.15.2.1 State (state)

Definition		
Command option:	Set / Get	
Parameter 1:	**<state>**	The state of the Webcat (websocket-client) module
	Default value	off
	Range	[on, off]
Description:	The Webcat (websocket client) module returns the current state and can be controlled with the parameter.	
GET-Rights:	0x01	
SET-Rights:	0x04	

Examples	
HTTP/JSON Get-Request	`["get","webcat","state"]`
HTTP/JSON Get-Response	`["get","webcat","state","0","off"]`
HTTP/JSON Set-Request	`["set","webcat","state","on"]`
HTTP/JSON Set-Response	`["set","webcat","state","0","on"]`
CMD-UART Get-Request	`get webcat state\x0d\x0a`
CMD-UART Get-Response	`get webcat state 0 off\x0d\x0a`
CMD-UART Set-Request	`set webcat state on\x0d\x0a`
CMD-UART Set-Response	`set webcat state 0 on\x0d\x0a`

4.15.2.2 Server (variable: server)

Definition		
Command option:	Get / Set	
Parameter 1:	**<server-addr>**	Server address.
	Default value	-
	Range	String with maximum of 64 characters (underscores are not allowed).
Parameter 2:	**<port>** -	Server port number.
	Default value	-
	Range	[0…65535]
Parameter 3:	**<security>**	HTTP-Server security.
	Default value	auto
	Range	[off, on, auto]
Parameter 4:	**<username>**	Server username.
	Default value	-
	Range	String with maximum of 30 characters.
Parameter 5:	**<password>**	Server password.
	Default value	-
	Range	String with maximum of 64 characters.
Description:	Configuration of the Webcat (websocket-Client) server	
GET-Rights:	0x01	
SET-Rights:	0x04	

Examples	
HTTP/JSON Get-Request	`["get","webcat","server"]`
HTTP/JSON Get-Response	`["get","webcat","server","0","ServerAdr","80","auto","username"]`
HTTP/JSON Set-Request	`["set","webcat","server","ServerAdr","80","auto","username","username", "password"]`
HTTP/JSON Set-Response	`["set","webcat","server","0","ServerAdr","80","auto","username","username", "password"]`
CMD-UART Get-Request	`get webcat server\x0d\x0a`
CMD-UART Get-Response	`get webcat server 0 ServerAdr 80 auto username username\x0d\x0a`
CMD-UART Set-Request	`set webcat server ServerAdr 80 auto username username password\x0d\x0a`
CMD-UART Set-Response	`set webcat server 0 ServerAdr 80 auto username username password\x0d\x0a`

4.15.2.3 Server Resource (variable: resource)

Definition		
Command option:	Get / Set	
Parameter 1:	**<resource>**	Server resource.
	Default value	-
	Range	String with maximum of 64 characters.
Description:	Server resource.	
GET-Rights:	0x01	
SET-Rights:	0x04	

Examples	
HTTP/JSON Get-Request	`["get","webcat","resource"]`
HTTP/JSON Get-Response	`["get","webcat","resource","0","serverresource"]`
HTTP/JSON Set-Request	`["set","webcat","resource","serverresource"]`
HTTP/JSON Set-Response	`["set","webcat","resource","0","serverresource"]`
CMD-UART Get-Request	`get webcat resource\x0d\x0a`
CMD-UART Get-Response	`get webcat resource 0 serverresource\x0d\x0a`
CMD-UART Set-Request	`set webcat resource serverresource\x0d\x0a`
CMD-UART Set-Response	`set webcat resource 0 serverresource\x0d\x0a`

4.15.2.4 Server Extension (serverext)

Definition		
Command option:	Get / Set	
Parameter 1:	**<origin>**	An optional origin header field [RFC6454] is used to protect against.
	Default value	-
	Range	String with maximum of 64 characters.
Parameter 2:	**<protocol>**	An optional protocol header field [RFC6454] which indicates the subprotocol that the server has selected.
	Default value	-
	Range	String with maximum of 64 characters.
Parameter 3:	**<extension>**	An optional extension header field [RFC6454] with a list of values indicating which extensions the client would like to speak.
	Default value	-
	Range	String with maximum of 64 characters.
Description:	This configuration is optional and should only be set, if the server requires the information.	
GET-Rights:	0x01	
SET-Rights:	0x04	

Examples	
HTTP/JSON Get-Request	`["get","webcat","serverext"]`
HTTP/JSON Get-Response	`["get","webcat","serverext","0","http://example.com","chat","deflate-stream"]`
HTTP/JSON Set-Request	`["set","webcat","serverext","http://example.com","chat","deflate-stream"]`
HTTP/JSON Set-Response	`["set","webcat","serverext","0","http://example.com","chat","deflate-stream"]`
CMD-UART Get-Request	`get webcat serverext\x0d\x0a`
CMD-UART Get-Response	`get webcat serverext 0 http://example.com chat deflate-stream username\x0d\x0a`
CMD-UART Set-Request	`set webcat serverext http://example.com chat deflate-stream\x0d\x0a`
CMD-UART Set-Response	`set webcat serverext 0 http://example.com chat deflate-stream\x0d\x0a`

4.15.2.5 Status (variable: status)

Definition		
Command option:	Get	
Parameter 1:	**<status>**	Webcat status.
	Default value	0
	Range	see list ⇨ 4.15.1 Webcat Status Values.
Description:	Returns the status of the Webcat service.	
GET-Rights:	0x01	

Examples	
HTTP/JSON Get-Request	`["get","webcat","status"]`
HTTP/JSON Get-Response	`["get","webcat","status","0","1"]`
CMD-UART Get-Request	`get webcat status\x0d\x0a`
CMD-UART Get-Response	`get webcat status 0 1\x0d\x0a`

4.16 MQTT-Client

MQTT-Client is a binary UART to MQTT Tunnel-Application. Messages can be transferred to and received from a MQTT-Server (Broker) via the BIN UART. Setting the server parameters and switching the client on or off are done by defined commands. All commands can be send over the CMD UART. Notifications will be received from the CMD UART too. All commands answer with a return code. A return code of 0 means successful operation. Other values describe possible errors.

Basic supported commands for the MQTT-Client service

- server — basic server parameters (server ip, port, tls, login and password)
- resource — server resource
- cfg — session configuration
- lastwill — last will message options
- state — on/off functionality
- status — status information of MQTT-Client
- publish — publish a message on a topic
- subscribe — subscribe to a topic
- unsubscribe — unsubscribe from a topic
- list — list all subscriptions

Basic steps for using Webcat

1. Configure the Server and session parameters (server, resource, cfg, lastwill)
2. Start the MQTT-Client module
3. Subscribe to max. 5 topics

4. Receive and publish messages
5. Stop the MQTT-Client module

Configure an MQTT Server to connect to

The first step is to define a server that should be used as MQTT destination:

```
set mqttc server <server ip or domain name> <port> <security> [<login>
<password>]
```

Server IP and port are mandatory parameters. The other 2 parameters are optional. *Login* and *password* are empty strings "" per default.

Afterwards, define the resource if needed.

```
set mqttc resource <resource>
```

If this parameter is not set, then the default resource is "/".

Next, setup the session configuration.

```
set mqttc cfg <clientID> <cleanSession> <keepAlive> <retryTimeout>
```

Finally, configure the last will message.

```
set mqttc lastwill <QoS> <topic> <msg>
```

Start the MQTT – UART tunnel connection

The connection to the server will be started as the module is turned on. This is done with the following command:

```
set mqttc state on
```

Shortly after executing this command the MQTT-Client connection is established. The user will get a notification like

```
info mqttc connect OK

info mqttc connection ERROR
```

Additionally, the host controller can check the connection state with:

```
get mqttc status
```

The status codes are explained in the status value table below.

Data transfer

If the MQTT-Client connection has been established successfully, then the data exchange is done using the following commands.

Send a message of *size* bytes on a specific *topic* with a given *QoS* level to the broker.

```
send mqttc publish <QoS> <topic> <retain> <size>
```

The message payload of *size* must be written to the binary UART.

The user will get a notification like:

```
info mqttc publish OK

info mqttc publish ERROR
```

Subscribe to a specific *topic* with a given *QoS* level.

```
set mqttc subscribe <QoS> <topic>
```

The user will get a notification like:

```
info mqttc subscribe OK

info mqttc subscribe ERROR
```

Unsubscribe from a specific *topic*.

```
set mqttc unsubscribe <topic>
```

The user will get a notification like:

```
info mqttc unsubscribe OK

info mqttc unsubscribe ERROR
```

List all subscribed topics with their individual QoS levels (max. 5 entries).

```
get mqttc list
```

Incoming messages will produce notifications like:

One chunk, message to be continued:

```
info mqttc recvmsg <QoS> <topic> <retain> <duplicate> <totalSize> <size>
```

Last chunk, message to be complete:

```
info mqttc recvmsg <QoS> <topic> <retain> <duplicate> <totalSize> <size> OK
```

Message incomplete:

```
info mqttc recvmsg ERROR
```

The message payload of *size* bytes can be read from the BIN UART.

Internal MQTT errors will produce notifications like:

```
info mqttc internal ERROR
```

Quit the MQTT-Client Connection

Quitting the MQTT-Client connection is done by turning the module off.

```
set mqttc state off
```

The user will get a notification like:

```
info mqttc disconnect OK
```

4.16.1 MQTT-Client Status Values

Status Code	Description
0	Not configured
1	Ready to send data
2	Data is sending
3	Error
4	Connecting

4.16.2 API-Commands (Module: mqttc)

4.16.2.1 State (state)

Definition		
Command option:	Set / Get	
Parameter 1:	**<state>**	The state of the MQTT-Client module
	Default value	off
	Range	[on, off]
Description:	The MQTT-Client module returns the current state and can be controlled with the parameter.	
GET-Rights:	0x01	
SET-Rights:	0x04	

Examples	
HTTP/JSON Get-Request	`["get","mqttc","state"]`
HTTP/JSON Get-Response	`["get","mqttc","state","0","off"]`
HTTP/JSON Set-Request	`["set","mqttc","state","on"]`
HTTP/JSON Set-Response	`["set","mqttc","state","0","on"]`
CMD-UART Get-Request	`get mqttc state\x0d\x0a`
CMD-UART Get-Response	`get mqttc state 0 off\x0d\x0a`
CMD-UART Set-Request	`set mqttc state on\x0d\x0a`
CMD-UART Set-Response	`set mqttc state 0 on\x0d\x0a`

4.16.2.2 Server (variable: server)

Definition		
Command option:	Get / Set	
Parameter 1:	**<server-addr>**	Server address or host name.
	Default value	-
	Range	String with maximum of 64 characters (underscores are not allowed).
Parameter 2:	**<port>**	Server port number.
	Default value	-
	Range	[0…65535]
Parameter 3:	**<security>**	HTTP-Server security.
	Default value	-
	Range	[off, on, auto]
Parameter 4:	**<username>**	Server username.
	Default value	-
	Range	String with maximum of 128 characters.
Parameter 5:	**<password>**	Server password.
	Default value	-
	Range	String with maximum of 160 characters.
Description:	Configuration of the MQTT server	
GET-Rights:	0x01	
SET-Rights:	0x04	

Examples	
HTTP/JSON Get-Request	`["get","mqttc","server"]`
HTTP/JSON Get-Response	`["get","mqttc","server","0","ServerAdr","80","auto","username"]`
HTTP/JSON Set-Request	`["set","mqttc","server","ServerAdr","80","auto","username","username", "password"]`
HTTP/JSON Set-Response	`["set","mqttc","server","0","ServerAdr","80","auto","username", "password"]`
CMD-UART Get-Request	`get mqttc server\x0d\x0a`
CMD-UART Get-Response	`get mqttc server 0 ServerAdr 80 auto username\x0d\x0a`
CMD-UART Set-Request	`set mqttc server ServerAdr 80 auto username password\x0d\x0a`
CMD-UART Set-Response	`set mqttc server 0 ServerAdr 80 auto username password\x0d\x0a`

4.16.2.3 Server Resource (variable: resource)

Definition		
Command option:	Get / Set	
Parameter 1:	**<resource>**	Server resource.
	Default value	"/"
	Range	String with maximum of 64 characters.
Description:	Server resource configuration.	
GET-Rights:	0x01	
SET-Rights:	0x04	

Examples	
HTTP/JSON Get-Request	`["get","mqttc","resource"]`
HTTP/JSON Get-Response	`["get","mqttc","resource","0","serverresource"]`
HTTP/JSON Set-Request	`["set","mqttc","resource","serverresource"]`
HTTP/JSON Set-Response	`["set","mqttc","resource","0","serverresource"]`
CMD-UART Get-Request	`get mqttc resource\x0d\x0a`
CMD-UART Get-Response	`get mqttc resource 0 serverresource\x0d\x0a`
CMD-UART Set-Request	`set mqttc resource serverresource\x0d\x0a`
CMD-UART Set-Response	`set mqttc resource 0 serverresource\x0d\x0a`

4.16.2.4 Session Config (variable: cfg)

Definition		
Command option:	Get / Set	
Parameter 1:	**<clientID >**	The client ID.
	Default value	-
	Range	String with maximum of 64 characters.
Parameter 2:	**<cleanSession>**	The flag for a clean session.
	Default value	-
	Range	[0 = false, 1 = true]
Parameter 3:	**<keepAlive>**	The session keep alive value in seconds.
	Default value	-
	Range	[0…65535]
Parameter 4:	**<retryTimeout>**	The session establishment retry timeout in seconds.
	Default value	-
	Range	[0…65535]
Description:	The MQTT-Client session configuration.	
GET-Rights:	0x01	
SET-Rights:	0x04	

Examples	
HTTP/JSON Get-Request	`["get","mqttc","cfg"]`
HTTP/JSON Get-Response	`["get","mqttc","cfg","0","id123","0","60","30"]`
HTTP/JSON Set-Request	`["set","mqttc","cfg","id123","0","60","30"]`
HTTP/JSON Set-Response	`["set","mqttc","cfg","0","id123","0","60","30"]`
CMD-UART Get-Request	`get mqttc cfg\x0d\x0a`
CMD-UART Get-Response	`get mqttc cfg 0 id123 0 60 30\x0d\x0a`
CMD-UART Set-Request	`set mqttc cfg id123 0 60 30\x0d\x0a`
CMD-UART Set-Response	`set mqttc cfg 0 id123 0 60 30\x0d\x0a`

4.16.2.5 Last will Config (variable: lastwill)

Definition		
Command option:	Get / Set	
Parameter 1:	**<QoS>**	The Quality of Service (QoS) level.
	Default value	-
	Range	[0…2]
Parameter 2:	**<topic>**	The last will topic.
	Default value	-
	Range	String with maximum of 128 characters.
Parameter 3:	**<msg>**	The last will message.
	Default value	-
	Range	String with maximum of 128 characters.
Description:	The MQTT-Client last will configuration.	
GET-Rights:	0x01	
SET-Rights:	0x04	

Examples	
HTTP/JSON Get-Request	`["get","mqttc","lastwill"]`
HTTP/JSON Get-Response	`["get","mqttc","lastwill","0","0","top1","bye"]`
HTTP/JSON Set-Request	`["set","mqttc","lastwill","0","top1","bye"]`
HTTP/JSON Set-Response	`["set","mqttc","lastwill","0","0","top1","bye"]`
CMD-UART Get-Request	`get mqttc lastwill\x0d\x0a`
CMD-UART Get-Response	`get mqttc lastwill 0 0 top1 bye\x0d\x0a`
CMD-UART Set-Request	`set mqttc lastwill 0 top1 bye\x0d\x0a`
CMD-UART Set-Response	`set mqttc lastwill 0 0 top1 bye\x0d\x0a`

4.16.2.6 Publish Message (variable: publish)

Definition		
Command option:	Send	
Parameter 1:	**<QoS >**	The Quality of Service (QoS) level.
	Default value	-
	Range	[0…2]
Parameter 2:	**<topic>**	The message topic.
	Default value	-
	Range	String with a maximum of 128 characters.
Parameter 3:	**<retain>**	The retain flag.
	Default value	-
	Range	[0 = false, 1 = true]
Parameter 4:	**<size>**	The size of the message.
	Default value	-
	Range	[0…1024]
Description:	The publish message on a given topic and with QoS level and retain flag. The payload with the given size must be written to the binary UART.	
SEND-Rights:	0x04	

Example message payload: „hello“.

Examples	
HTTP/JSON Send-Request	`["send","mqttc","publish","0","top1","0","5"]`
HTTP/JSON Send-Response	`["send","mqttc","publish","0","0","top1","0","5"]`
CMD-UART Send-Request	`send mqttc publish 0 top1 0 5\x0d\x0a`
CMD-UART Send-Response	`send mqttc publish 0 0 top1 0 5\x0d\x0a`
CMD-UART Notification	`info mqttc publish top1 OK\x0d\x0a`
CMD-UART Notification	`info mqttc publish top1 ERROR\x0d\x0a`
CMD-UART Notification	`info mqttc recvmsg 0 top1 0 0 15 5\x0d\x0a`
CMD-UART Notification	`info mqttc recvmsg 0 top1 0 0 10 5 OK\x0d\x0a`
CMD-UART Notification	`info mqttc recvmsg ERROR\x0d\x0a`

4.16.2.7 Subscribe on Topic (variable: subscribe)

Definition		
Command option:	Set	
Parameter 1:	**<QoS>**	The Quality of Service (QoS) level.
	Default value	-
	Range	[0…2]
Parameter 2:	**<topic>**	The topic of the message.
	Default value	-
	Range	String with maximum of 128 characters.
Description:	Subscribe to a topic with a QoS level.	
SET-Rights:	0x04	

Examples	
HTTP/JSON Set-Request	`["set","mqttc","subscribe","0","top1"]`
HTTP/JSON Set-Response	`["set","mqttc","subscribe","0","0","top1"]`
CMD-UART Set-Request	`set mqttc subscribe 0 top1\x0d\x0a`
CMD-UART Set-Response	`set mqttc subscribe 0 0 top1\x0d\x0a`
CMD-UART Notification	`info mqttc subscribe top1 OK\x0d\x0a`
CMD-UART Notification	`info mqttc subscribe top1 ERROR\x0d\x0a`

4.16.2.8 Unsubscribe from Topic (variable: unsubscribe)

Definition		
Command option:	Set	
Parameter 1:	**<topic>**	The topic of the message.
	Default value	-
	Range	String with maximum of 128 characters.
Description:	Unsubscribe from a topic.	
SET-Rights:	0x04	

Examples	
HTTP/JSON Set-Request	`["set","mqttc","unsubscribe","top1"]`
HTTP/JSON Set-Response	`["set","mqttc","unsubscribe","0","top1"]`
CMD-UART Set-Request	`set mqttc unsubscribe top1\x0d\x0a`
CMD-UART Set-Response	`set mqttc unsubscribe 0 top1\x0d\x0a`
CMD-UART Notification	`info mqttc unsubscribe top1 OK\x0d\x0a`
CMD-UART Notification	`info mqttc unsubscribe top1 ERROR\x0d\x0a`

4.16.2.9 List all Subscriptions (variable: list)

Definition		
Command option:	Get	
Description:	Show all subscriptions with their individual QoS levels.	
GET-Rights:	0x01	

Examples	
HTTP/JSON Get-Request	`["get","mqttc","list"]`
HTTP/JSON Get-Response	`["get","mqttc","list","0","0","top1","0","top2",…]`
CMD-UART Get-Request	`get mqttc list\x0d\x0a`
CMD-UART Get-Response	`get mqttc list 0 0 top1 0 top2 …\x0d\x0a`

4.16.2.10 Status (variable: status)

Definition		
Command option:	Get	
Parameter 1:	**<status>**	The MQTT-Client status.
	Default value	-
	Range	See 4.16.1 MQTT-Client Status Values.
Description:	Returns the status of the MQTT-Client service.	
GET-Rights:	0x01	

Examples	
HTTP/JSON Get-Request	`["get","mqttc","status"]`
HTTP/JSON Get-Response	`["get","mqttc","status","0","1"]`
CMD-UART Get-Request	`get mqttc status\x0d\x0a`
CMD-UART Get-Response	`get mqttc status 0 1\x0d\x0a`

5 Status Information

5.1 Telegram Return Code

The telegram returns a return code with every GET or SET request. The return code can be delivered in two different ways.

- The return code is delivered with an ERROR telegram ⇨ Return Code Error Telegram.
- The return code is delivered in the normal telegram after the variable ⇨ Return Code Info and Return Code for Normal Telegram.

Return Code Info

Return Code	Description
0	No error detected
10	Info that the binary UART still has data left
11	Warning: Data overflow on the binary UART

Return Code Error Telegram

Return Code	Description
100	Empty telegram
101	Command not found. Only get or set command is valid.
102	Module not found
103	Variable not found
104	Data error
105	Internal buffer error
106	Command returned busy

Return Code for Normal Telegram

Return Code	Description
201	No permission to get or set this parameter
202	Get request is not possible for this variable
203	Set request is not possible for this variable
204	Parameter error
205	Format error
206	Variable not found
207	Communication error with the UART
208	Internal Storage error
209	Invalid interface
301	The parameter length is too long for the variable
302	The application is not yet ready
303	Netcat wrong exclusive lock ID
304	Binuart is already locked
305	The module/service is not configured correctly
306	The module/service is busy
307	The binuart is busy
308	Command is rejected
501	It is not possible to open a websocket

Examples for Error Telegram	
HTTP/JSON Get-Request	`["test","system","firmware"]`
HTTP/JSON Get-Response	`["ERROR","101"]`
CMD-UART Get-Request	`test system firmware\x0d\x0a`
CMD-UART Get-Response	`ERROR 101 "Command not found"\x0d\x0a`

Examples for a Normal Telegram Return Code	
HTTP/JSON Set-Request	`["set","wlan","cfg","sta","ssid","pwd","7"]`
HTTP/JSON Set-Response	`["set","wlan","cfg","204","sta","ssid","pwd","7"]`
CMD-UART Set-Request	`set wlan cfg sta ssid pwd 7\x0d\x0a`
CMD-UART Set-Response	`set wlan cfg 204 sta ssid pwd 7\x0d\x0a`

5.2 Wi-Fi Parameter

Wi-Fi Status Information

Status Code	Description
0	Not initialized
1	Not connected to a network
2	Establishing a connection
3	IP address is obtained
4	Connected to the network
5	Searching for networks
6	Entered network not found
7	Authentication failed
10	Access Point not active
11	Access Point created

5.3 Mail Service

Mail Module Status

Status Code	Description
0	Not configured
1	Configuration ok - ready to send
2	Mail is sending
3	Failed to connect to server
4	Authentication failed

5.4 User Management

User Rights

Rights	Description
0x80	Rights to change the user configuration update
0x40	Rights to do a firmware update
0x20	Rights to SET/GET data via BINUART
0x10	Rights to SET/GET data via CMDUART HIGH
0x08	Rights to SET/GET data via CMDUART LOW
0x04	Rights to SET parameters with a high priority
0x02	Rights to SET parameters with a low priority
0x01	Rights to GET parameters

5.5 Firmware Update

Firmware Update Status List

Status Code	Description
0	Ready for update
1	Update active
2	Checking for new firmware
3	Update done
4	Update failed

5.6 HTTP Client

HTTP Client Status List

Status Code	Description
0	Not configured
1	Ready to send data
2	Data is sending
3	Error

5.7 Netcat

Netcat Status List

Status Code	Description
0	Not configured
1	Ready to send data
2	Data is sending
3	Error
4	Connecting

5.8 UDP

UDP Status List

Status Code	Description
0	Not configured
1	Ready to send data
2	Data is sending
3	Error
4	Connecting

5.9 Webcat

Webcat Status List

Status Code	Description
0	Not configured
1	Ready to send data
2	Data is sending
3	Error
4	Connecting

5.10 MQTT Client

MQTT Status List

Status Code	Description
0	Not configured
1	Ready to send data
2	Data is sending
3	Error
4	Connecting

6 Contact Details

6.1 Contact Us

Please contact your local Panasonic Sales office for details on additional product options and services:

For Panasonic Sales assistance in the **EU**, visit

https://eu.industrial.panasonic.com/about-us/contact-us

Email: wireless@eu.panasonic.com

For Panasonic Sales assistance in **North America**, visit the Panasonic Sales & Support Tool to find assistance near you at

https://na.industrial.panasonic.com/distributors

Please visit the **Panasonic Wireless Technical Forum** to submit a question at

https://forum.na.industrial.panasonic.com

6.2 Product Information

Please refer to the Panasonic Wireless Connectivity website for further information on our products and related documents:

For complete Panasonic product details in the **EU**, visit

http://pideu.panasonic.de/products/wireless-modules.html

For complete Panasonic product details in **North America**, visit

http://www.panasonic.com/rfmodules