Charger local API

Version: Prerelease 02.2023

1 Commands format

All commands follow the *Request/Response* JSON-RPC 2.0 format specification: https://www.jsonrpc.org/specification.

Charger can be controlled over WiFi via RESTful commands in the following format:
GET / POST http://{IP}/rpc/{command}

Example:
GET http://192.168.0.163/rpc/charge.start

POST http://192.168.0.163/rpc/app_config.set
Body (application/json):

```
{
  "config_key": "led_max_brightness",
  "config_value": 30
}
```

2 Error codes.

Error code	Error description
0	No error. Everything was successful.
-1	General fail. No details.
-2	JSON request format error. Usually means that a not all keys were found.
-3	Key value length error. Applicable to keys of string type which should be of a fixed length.
-4	Key value range error. The key value is out of range. Applicable to keys of integer type.
-5	Error key format. Example: setting integer value to a boolean key.
-6	Unknown key error.

3 Application configuration keys.

Key name	Key type	Default value	Accepted values	Description.
headless	boolean	true	true/false	Defines if the charger is headless (no OCPP control, just plug in the connector and it will charge) or not.
install_current	integer	6	6-32	Current value [A] to be limited by software.
led_max_brightness	integer	100	10 - 100	LED maximum brightness in %.

led_dimming_delay	integer	0	0 - INT_MAX	LED delay value in seconds before applying maximum brightness value. 0 means LEDs are always at maximum brightness.
charger_locked	boolean	false	true/false	Specify if the charger is locked or not. If key is true, no charging session will be started.
led_template_enabled	boolean	false	true/false	Specify if the custom LED template feature is enabled or not.
led_template	string	NULL	According to description	Custom LED template Segments as seen from the front. Format: "red1,green1,blue1, red2, green2,blue2,red3,green3,blue3,red4,green4,blue4" red1 – Top segment Red color. Values: 0..255 green1 – Top segment Red color. Values: 0..255 blue1 – Top segment Red color. Values: 0..255 red2 – Right segment Red color. Values: 0..255 green2 – Right segment Red color. Values: 0..255 blue2 – Right segment Red color. Values: 0..255 red3 – Bottom segment Red color. Values: 0..255 green3 – Bottom segment Red color. Values: 0..255 blue3 – Bottom segment Red color. Values: 0..255 red4 – Left segment Red color. Values: 0..255 green4 – Left segment Red color. Values: 0..255 blue4 – Left segment Red color. Values: 0..255
charger_alias	string	NULL	Any string	Charger alias. Will be used by the app.
latitude	double	0.0	Double value	Charger latitude coordinate.
longitude	double	0.0	Double value	Charger longitude coordinate.
ocpp_server_link	string	NULL		OCPP server link.
connected_phase	integer	0	1-3	Charger connected phase.
owner_id	string	NULL	Any string	Owner ID used for remote access. May only be set once after factory reset.
user_current	integer	32	6-32	User current.
gmt_offset	integer	0	-12 - 12	GMT offset. Used for charging schedule time calculations.
precise_pwm_mode	boolean	false	true/false	Enable/disable precise PWM mode. If enabled, the PWM is more stable, but the LEDs in state C are in a fixed state.
energy_management_enabled	boolean	false	true/false	Energy management control enabled flag.
charging_schedule_enabled	boolean	false	true/false	Charging schedule enabled flag. If disabled, the charger will ignore the charging schedule.
em_id	string	NULL	Any string	Current energy management device ID.
timezone	string	NULL	Any string	Timezone name

4 Counters configuration keys.

Key name	Key type	Accepted values	Description.
total_charged_time	integer	0 - INT_MAX	Total charged time [min].
total_charged_sessions	integer	0 - INT_MAX	Total charged sessions
total_charged_energy	integer	0 - INT_MAX	Total charged energy [kWh].
reset_counter	integer	0 - INT_MAX	Device Reset counter.
last_session_energy	integers	0 - INT_MAX	Last session energy value [Wh].

5 Commands

5.1 Remote charging start.

Method	**Charge.Start**
Description	Remote charging start.

Request params format	{
	"tag": string
	}
	
	"tag " – Tag name. Optional parameter.

Response result format	On success: true On error: Error message according to RPC-JSON specification and error code number described in *Error codes* chapter.

5.2 Remote charging stop.

Method	**Charge.Stop**
Description	Remote charging stop.
Request params format	No parameters.
Response result format	On success: true On error: Error message according to RPC-JSON specification and error code number described in *Error codes* chapter.

5.3 Get application configurations.

Method	**App_config.Get**
Description	Get application configurations.
Request params format	No parameters.

Response result format	{
	"key1_name": key1_type,
	"key2_name": key2_type,
	…
	"keyn_name": keyn_type
	}
	
	"key1_name" , "key2_name" … "keyN_name - See "*Application configuration keys"* chapter for description.

5.4 Set application configurations.

Method	**App_config.Set**
Description	Set application configuration OCPP style.

Request params format	{
	"config_key": string,
	"config_value": string
	}
	
	"config_key" - Config key name. See "*Application configuration keys"* chapter for description.
	"config_value" - Config key value.

Response result format	On success: true On error: Error message according to RPC-JSON specification and error code number described in *Error codes* chapter.

5.5 Get charger information.

Method	**Charger_info.Get**
Description	Get charger information.
Request params format	No parameters.
Response result format	{ "charger_state": string, "session_energy": double, "charging_time": integer, "session_id": integer, "instant_power": double, "current": double, "currents": [double, double, double], "current_limit_reason": integer, "voltage": double, "voltages": [double, double, double], "temperature": double,

	"energy_index": double, "total_charged_energy": integer "headless": bool, "led_max_brightness": integer, "install_current": integer, "dynamic_current": integer, "fw_version": string, "extended_charger_state": string, "has_active_errors": bool, "charger_is_paused": bool, "heap_free": integer, "pwm_current": integer, } "charger_state" - A string containing one letter (A, B, C, D, E, F) which conforms with IEC 61851 standard. "session_energy" - Current session charged energy in *Wh*. "charging_time" - Current session charging time in *s*. "session_id" - Current session ID. "instant_power" - Current instant power in *W*. "current" - Current value. "currents" - Array with current values. Order is L1, L2, L3. "current_limit_reason" - Current limit reason. Values: ● 0 - Current limited by HW capabilities. ● 1 - Current limited by install current. ● 2 - Current limited by user current. ● 3 - Current limited by dynamic current. ● 4 - Current limited by schedule. ● 5 - Current limited because Energy Management device is offline. ● 6 - Current limited by Energy Management device. ● 7 - Current limited by OCPP current. "voltage" - Current voltage value. "voltages" - Array with voltage values. Order is L1, L2, L3. "temperature" - Board temperature. "energy_index" - Current energy index value. "headless" – Headless flag value. "total_charged_energy" – Total charged energy. "led_max_brightness" – LED brightness value. "install_current" – Installation current value. "dynamic_current" – Dynamic current value. "fw_version" – Firmware version. "extended_charger_state" – Charger state with additional internal states. - “A”: Car disconnected. - “B”: Car connected. - “B_AUTH”: Car connected. Waiting for authentication. - “C”: Charging. - “D”: Charging. Ventilation needed. - “E”: Error state. - “OTA”: OTA state. - “LOCKED”: Charger is locked. Charging is impossible.

	- “B_PAUSE”: Charging is paused. - “B_SCHEDULER”: Charging is paused because of scheduler. "has_active_error" – Flag to specify if charger has active errors. "charger_is_paused" – Flag stating if the chargers I paused. "heap_free" - Heap free space. "pwm_current" - Current value at which the PWM was calculated.

5.6 Get software version.

Method	**Sw_version.Get**
Description	Get software version.
Request params format	No parameters.

Response result format	{
	"fw_version": string,
	"fw_id": string
	}
	
	"fw_version" - Firmware version number.
	"fw_id" - Firmware id.

5.7 Restart device.

Method	**Device.Reset**
Description	Reset device. If a charging session is active, the device will wait for its end first.
Request params format	No parameters.
Response result format	On success: true On error: Error message according to RPC-JSON specification and error code number described in *Error codes* chapter.

5.8 Get logged errors.

Method	**Logged_errors.Get**
Description	Get error flags. Logged errors are kept until reset by **Logged_errors.Reset**
Request params format	No parameters.
Response result format	{ "state_e_activated": bool, "overtemp": bool, "critical_temp": bool, "overcurrent": bool,

	"meter_fault": bool, "voltage_error": bool, "rcd_error": bool, "rcd_test_error": bool, "rcd_err_pin_error": bool, "rcd_err6_pin_error": bool, "rcd_init_start_error": bool, "rcd_init_end_error": bool, "state_f_activated": bool, "contactor_failure": bool, "cp_diode_failure": bool, "undervoltage_error": bool, "overvoltage_error": bool, } "state_e_activated" – State machine E activated flag. If *true*, flag is set. "overtemp" – Overtemperature flag. If *true*, flag is set. "critical_temp" – Critical temperature flag. If *true*, flag is set. "overcurrent" – Overcurrent flag. If *true*, flag is set. "meter_fault" – Meter fault flag. If *true*, flag is set. "voltage_error" – Voltage error flag. If *true*, flag is set. "rcd_error" – RCD error flag. If *true*, flag is set. "rcd_test_error" – RCD test error flag. If *true*, flag is set. "rcd_err_pin_error" – RCD err pin error flag. If *true*, flag is set. "rcd_err6_pin_error" – RCD 6mA pin error flag. If *true*, flag is set. "rcd_init_start_error" – RCD init start error flag. If *true*, flag is set. "rcd_init_end_error" – RCD init end error flag. If *true*, flag is set. "state_f_activated" – State machine F activated error flag. If *true*, flag is set. "contactor_failure" – Contactor failure error flag. If *true*, flag is set. "cp_diode_failure" – CP diode failure error flag. If *true*, flag is set. "undervoltage_error" – Undervoltage error flag. If *true*, flag is set. "overvoltage_error" – Overvoltage error flag. If *true*, flag is set.

5.9 Reset logged errors.

Method	**Logged_errors.Reset**
Description	Reset logged errors.
Request params format	No parameters.
Response result format	On success: true On error: Error message according to RPC-JSON specification and error code number described in *Error codes* chapter.

5.10 Get active errors.

Method	**Active_errors.Get**

Description	Get active errors.
Request params format	No parameters.

Response result format	{
	"state_e_activated": bool,
	"overtemp": bool,
	"critical_temp": bool,
	"overcurrent": bool,
	"meter_fault": bool,
	"voltage_error": bool,
	"rcd_error": bool,
	"rcd_test_error": bool,
	"rcd_err_pin_error": bool,
	"rcd_err6_pin_error": bool,
	"rcd_init_start_error": bool,
	"rcd_init_end_error": bool,
	"state_f_activated": bool,
	"contactor_failure": bool,
	"cp_diode_failure": bool,
	"undervoltage_error": bool,
	"overvoltage_error": bool,
	}
	
	"state_e_activated" – State machine E activated flag. If *true*, flag is set.
	"overtemp" – Overtemperature flag. If *true*, flag is set.
	"critical_temp" – Critical temperature flag. If *true*, flag is set.
	"overcurrent" – Overcurrent flag. If *true*, flag is set.
	"meter_fault" – Meter fault flag. If *true*, flag is set.
	"voltage_error" – Voltage error flag. If *true*, flag is set.
	"rcd_error" – RCD error flag. If *true*, flag is set.
	"rcd_test_error" – RCD test error flag. If *true*, flag is set.
	"rcd_err_pin_error" – RCD err pin error flag. If *true*, flag is set.
	"rcd_err6_pin_error" – RCD 6mA pin error flag. If *true*, flag is set.
	"rcd_init_start_error" – RCD init start error flag. If *true*, flag is set.
	"rcd_init_end_error" – RCD init end error flag. If *true*, flag is set.
	"state_f_activated" – State machine F activated error flag. If *true*, flag is set.
	"contactor_failure" – Contactor failure error flag. If *true*, flag is set.
	"cp_diode_failure" – CP diode failure error flag. If *true*, flag is set.
	"undervoltage_error" – Undervoltage error flag. If *true*, flag is set.
	"overvoltage_error" – Overvoltage error flag. If *true*, flag is set.

5.11 Get wifi status.

Method	**Wifi_status.Get**
Description	Get wifi status.
Request params format	No parameters.

Response result format	{
	"connected": bool,
	"sta_ip": string,
	"sta_name": string,
	"rssi": integer,
	"bssid": string
	}
	
	"connected" – Wifi status. *True* if connected.
	"sta_ip" – Station IP.
	"sta_name" – Station name.
	"rssi" – RSSI level.
	"bssid" – BSSID, if known.

5.12 Get system currents.

Method	**System_currents.Get**
Description	Get system currents.
Request params format	No parameters.

Response result format	{
	"cable_current ": integer,
	"install_current ": integer,
	"dynamic_current ": integer,
	"user_current": integer,
	"charging_current ": integer,
	"system_max_current ": integer,
	"schedule_current ": integer,
	"ocpp_current ": integer,
	"em_current ": integer,
	}
	
	"cable_current" – Cable current value.
	"install_current" – Install current value.
	"dynamic_current" – Dynamic current value. [depreciated]
	"user_current" – User current value.
	"charging_current" – Charging current value.
	"system_max_current" – System max current value.
	"schedule_current" – Schedule current value.
	"ocpp_current" – System max current value.
	"em_current" – Current limit used for load balancing.

5.13 Get counters configurations.

Method	**Counters_config.Get**
Description	Get counters configurations.
Request params format	No parameters.

Response result format	{
	"key1_name": key1_type,
	"key2_name": key2_type,
	…
	"keyn_name": keyn_type
	}
	
	"key1_name" , "key2_name" … "keyN_name - See "*Counters configuration keys"* chapter for description.

5.14 FW Update check/trigger

Method	**Ota.Verify**
Description	Trigger update check and update device if new version available.
Request params format	No parameters.
Response result format	On success: true On error: Error message according to RPC-JSON specification and error code number described in *Error codes* chapter.

5.15 Factory Reset

Method	**Factory.Reset**
Description	Reset to factory settings.
Request params format	No parameters.
Response result format	On success: true On error: Error message according to RPC-JSON specification and error code number described in *Error codes* chapter.

5.16 Set AP password.

Method	**Ap_password.Set**
Description	Set charger configuration.

Request params format	{
	"password": string
	}
	
	"password" – AP password. 8 to 63 characters. If length is 0, password access is removed.

Response result format	On success: true On error: Error message according to RPC-JSON specification and error code number described in *Error codes* chapter.

5.17 Pause charge.

Method	**Charge.Pause**
Description	Pause the charging process. **Alternative:** use em_current to control the charging process (using ***em_current.set***): 0 A - pause charging 6-32A - charge
Request params format	No parameters.
Response result format	On success: true On error: Error message according to RPC-JSON specification and error code number described in *Error codes* chapter.

5.18 Resume charge.

Method	**Charge.Resume**
Description	Resume charging after a **Charge.Pause** command
Request params format	No parameters.
Response result format	On success: true On error: Error message according to RPC-JSON specification and error code number described in *Error codes* chapter.

5.19 Get device info.

Method	**Device_info.Get**
Description	Get serial number.
Request params format	No parameters.
Response result format	{ "device_id": string, "connected_phase": integer, "fw_version": string,

	"charger_alias": string } "device_id" – Device ID. "connected_phase" – Connected phase. "fw_version" – Firmware version. "charger_alias" – Charger alias.

5.20 Set charging schedule.

Method	**Charging_schedule.Set**
Description	Set charging schedule.

Request params format	{
	"charging_schedule":
	[
	{
	"start_d": integer,
	"start_h": integer,
	"start_m": integer,
	"end_d": integer,
	"end_h": integer,
	"end_m": integer,
	"current": integer,
	},
	{...}
	]
	}
	
	"start_d" – Day of the week at which the interval is activated. Range: 1-7. First day of the week is Monday.
	"start_h" – Hour at which the interval is activated. Range: 0-23.
	"start_m" – Minute at which the interval is activated. Range: 0-59.
	"end_d" – Day of the week at which the interval is deactivated. Range: 1-7. First day of the week is Monday.
	"end_h" – Hour at which the interval is activated. Range: 0-23.
	"end_m" – Minute at which the interval is activated. Range: 0-59.
	"current" – Current value for the time interval. A 0 value means charging is paused, otherwise the 6-32 range is applied.

Response result format	On success: true On error: Error message according to RPC-JSON specification and error code number described in *Error codes* chapter.

5.21 Get charging schedule.

Method	**Charging_schedule.Get**
Description	Get charging schedule.

Request params format	No parameters.

Response result format	{
	"charging_schedule":
	[
	{
	"start_d": integer,
	"start_h": integer,
	"start_m": integer,
	"end_d": integer,
	"end_h": integer,
	"end_m": integer,
	"current": integer,
	},
	{...}
	]
	}
	
	"start_d" – Day of the week at which the interval is activated. Range: 1-7. First day of the week is Monday.
	"start_h" – Hour at which the interval is activated. Range: 0-23.
	"start_m" – Minute at which the interval is activated. Range: 0-59.
	"end_d" – Day of the week at which the interval is deactivated. Range: 1-7. First day of the week is Monday.
	"end_h" – Hour at which the interval is activated. Range: 0-23.
	"end_m" – Minute at which the interval is activated. Range: 0-59.
	"current" – Current value for the time interval. A 0 value means charging is paused, otherwise the 6-32 range is applied.

5.22 Override charging schedule.

Method	**Charging_schedule.Override**
Description	Ignore charging schedule for the current charging session and start charging. Charging schedule is reactivated when the car is disconnected.
Request params format	No parameters.
Response result format	On success: true On error: Error message according to RPC-JSON specification and error code number described in *Error codes* chapter.

5.23 Set Energy Management current.

Method	**EM_current.Set**
Description	Set EM current.

Request params format	{
	"EM_current": integer
	}
	
	"EM_current" – EM current value. Range: 0 or 6-32.
	0: Pause charging process.
	6 - 32: Resume charging process with current limited to value.

Response result format	On success: true On error: Error message according to RPC-JSON specification and error code number described in *Error codes* chapter.

5.24 Get Energy Management current.

Method	**EM_current.Get**
Description	Get EM current.
Request params format	No parameters.

Response result format	{
	"EM_current ": integer
	}
	
	"EM_current" – EM current value.

5.25 Get EM status.

Method	**EM_status.Get**
Description	Get EM device connection status.
Request params format	No parameters.

Response result format	{
	"em_offline ": boolean
	}
	
	"em_offline" – *True* if EM device is offline, *False* otherwise.