

Tasmota is a very well-known open-source implementation for microcontrollers based IoT devices (ESP-based, like ESP32, ESP8266) mostly used in smart home applications:

<https://tasmota.github.io/docs/>

Tasmota provides webserver based access (and many more interfaces!), where also a command API can be used via the webserver.

The library „TasmotAPI“ provides a http client implementation (based on the ashttp library) to communicate with such a Tasmota firmware based device via the web API.

The implementation just provides the webservice access itself, not the parsing of the JSON based response data -> this must be done by own code!

Commands and responses depend on the Tasmota device, please see here in the Tasmota command reference:

<https://tasmota.github.io/docs/Commands/>

How to use: please see the simple sample task provided together with the library.

For example, after requesting the command “status 10” from the Tasmota device (“status 10” means: deliver sensor data), the device response is a JSON based object containing all sensor information of the device.

“Status 10” on my test device (a smart plug with measurement functions) responses for example with the following JSON object:

The screenshot displays a PLC Ladder Logic editor with a program named `_CYCLIC`. The program contains a `PROGRAM_INIT` block where variables for `tasmotapiSendRequest_0` are initialized, including `sHost` ('192.168.168.105'), `sPort` (80), `sUser` (''), `sPassword` (''), `sCommand` ('status 10'), `pResponseBuffer` (ADR(sResponse)), and `uResponseBufferSize` (sizeof(sResponse)). The `enable` property is set to `TRUE`. Below this is a `PROGRAM_CYCLIC` block with a `IF bSendRequest = TRUE THEN` condition. Inside, it sets `tasmotapiSendRequest_0.bSendCommand := TRUE;`, calls `fb_cyclic`, and checks for `sendCommandBusy`. If not busy, it checks for `sendCommandError`. If no error, it checks for `sendCommandSuccess`. If successful, it sets `bSendRequest := FALSE;` and resets `bSendCommand`. The `Watch` window on the right shows the state of `tasmotapiSendRequest_0` and the `sResponse` array. The `enable` property is `TRUE`, `sHost` is '192.168.168.105', `sPort` is 80, `sUser` is '', `sPassword` is '', `sCommand` is 'status 10', `pResponseBuffer` is 77437898, `uResponseBufferSize` is 810, `bSendCommand` is `FALSE`, `sendCommandBusy` is `FALSE`, `sendCommandSuccess` is `FALSE`, `sendCommandError` is `FALSE`, `status` is 0, `httpStatus` is 0, and `responseDataLen` is 292. The `sResponse` array contains a JSON string: `{\"StatusSNS\":{\"Time\":\"2025-11-24T12:44:38\",\"ENERGY\":{\"TotalStartTime\":\"2025-11-23T10:33:58\",\"Total\":0.00000,\"Yesterday\":0.00000,\"Today\":0.00000,\"Power\":0.000,\"ApparentPower\":0.000,\"ReactivePower\":0.000,\"Factor\":0.00,\"Voltage\":221,\"Current\":0.0,\"ESP32\":{\"Temperature\":54.9,\"TempUnit\":\"C\"}}}}`.

Name	Value
bSendRequest	FALSE
tasmotapiSendRequest_0	
enable	TRUE
sHost	'192.168.168.105'
sPort	80
sUser	''
sPassword	''
sCommand	'status 10'
pResponseBuffer	77437898
uResponseBufferSize	810
bSendCommand	FALSE
sendCommandBusy	FALSE
sendCommandSuccess	FALSE
sendCommandError	FALSE
status	0
httpStatus	0
responseDataLen	292
sResponse	
sResponse[0]	{\"StatusSNS\":{\"Time\":\"2025-11-24T12:44:38\",\"ENERGY\":{\"TotalStartTime\":\"2025-11-23T10:33:58\",\"Total\":0.00000,\"Yesterday\":0.00000,\"Today\":0.00000,\"Power\":0.000,\"ApparentPower\":0.000,\"ReactivePower\":0.000,\"Factor\":0.00,\"Voltage\":221,\"Current\":0.0,\"ESP32\":{\"Temperature\":54.9,\"TempUnit\":\"C\"}}}}
sResponse[1]	
sResponse[2]	
sResponse[3]	
sResponse[4]	
sResponse[5]	