

Tasmota Device Manager – Tasmota-Geräte bequem per grafischer Oberfläche administrieren.

Im [Tasmota](#)-Ökosystem gibt es ein hervorragendes Management-Tool für [Tasmota](#)-Geräte. Die Software ist mit Python programmiert (ab Version 3.5) und setzt eine Kommunikation zu den Geräten über MQTT voraus. Die Installation ist kinderleicht und die Bandbreite der einstellbaren Parameter riesig. Die Software läuft, weil Python, auf Linux, Mac und Windows.

Installation:

Überprüfen ob Python 3.6 oder höher installiert ist

Mit dem Python-Paketmanager pip die Pakete *PyQT5* und *paho-mqtt* installieren

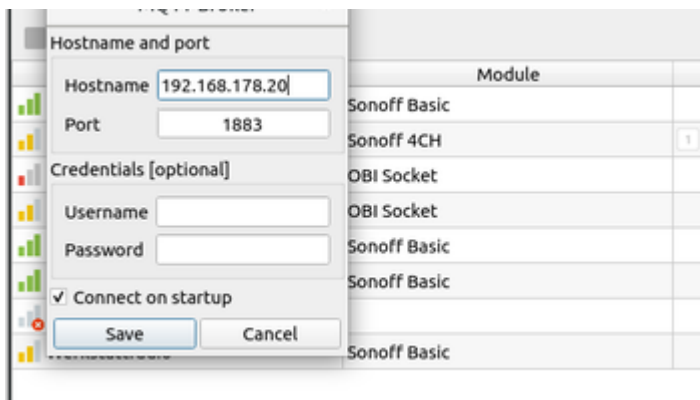
Von Github herunterladen: <https://github.com/jziolkowski/tadm/archive/master.zip>

Bzw. über die Webseite <https://github.com/jziolkowski/tadm>

Archiv in das Verzeichnis tadm-master auspacken

In das Verzeichnis wechseln und mit `python3 tadm.py` starten

Als erstes unter dem Menüpunkt MQTT den Eintrag Broker wählen und die IP sowie weitere Verbindungsinformationen angeben:

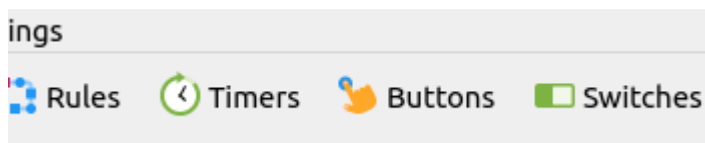


Nachdem die Verbindung zum MQTT-Broker hergestellt ist, findet TDM automatisch alle aktiven [Tasmota](#)-Geräte:

FriendlyName	Module	Power	Color	LoadAvg	LinkCount	Uptime
Drucker	Sonoff Basic	<input type="checkbox"/>		19	1	3d 22:27:11
Multischalter	Sonoff 4CH	<input type="checkbox"/>		19		0d:15:15
OBI-1	OBI Socket	<input type="checkbox"/>		19	1	1d 02:42:56
OBI-2	OBI Socket	<input type="checkbox"/>		27	1	1d 02:41:06
Pumpe	Sonoff Basic	<input type="checkbox"/>		19		3d 22:26:51
Schalter	Sonoff Basic	<input type="checkbox"/>		19		0d:15:45
Th10						
Werkstatradio	Sonoff Basic	<input type="checkbox"/>		1077	1	3d 22:27:02

View mode: Home Health Firmware WiFi MQTT

Nun können die Geräte in der Liste durch Klicken markiert werden und über eine Auswahl der gewünschten Funktionsbereiche im oberen Bereich des Hauptbildschirms einzelne Parameter geändert werden:



Durch Klicken auf die oben dargestellten Funktionsbereiche öffnen sich weitere Fenster mit sehr umfangreichen Einstellmöglichkeiten. Unten folgt eine Fotogalerie mit Bildern zu Console, [Rules](#), Timers und Power.

```

[14:04:15] Schalter/tele/SENSOR {"Time":"2019-10-04T13:04:15","Switch1":"ON","Switch2":"ON"}
[14:04:20] Schalter/cmd/status
[14:04:20] Schalter/stat/STATUS {"Status":{"Module":1,"FriendlyName":"Schalter","Topic":"Schalter","ButtonTopic":"Ventil-Pferdebiese","Power":0,"PowerOnState":0,"LedState":1,"SaveData":1,"SaveState":1,"SwitchTopic":"0","SwitchMode":{"3,3,0,0,0,0,0,0},"ButtonRetain":0,"SwitchRetain":0,"SensorRetain":0,"PowerRetain":0}}
[14:04:45] Schalter/tele/STATE {"Time":"2019-10-04T13:04:45","Uptime":"0T06:17:15","Vcc":3.523,"SleepMode":"Dynamic","Sleep":50,"LoadAvg":19,"POWER":"OFF","Wifi":{"AP":1,"SSID":"JCQ-98FEE4","RSSI":"00:1F:17:17:70","Channel":9,"RSSI":86}}
[14:04:45] Schalter/tele/SENSOR {"Time":"2019-10-04T13:04:45","Switch1":"ON","Switch2":"ON"}
  
```

Rule1 - Enabled [Once] [Stop on error] [Upload] Remaining: 386

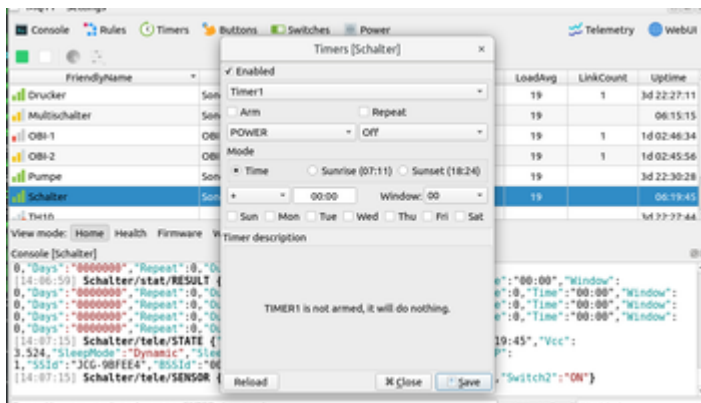
```

on Power!state=0 do
  publish Multischalter/cmd/POWER1 1
endon
on Power!state=0 do
  publish Multischalter/cmd/POWER1 0
endon
  
```

Automatic polling: VARs MEMs RuleTimers

```

[14:05:42] Schalter/cmd/mem4
[14:05:42] Schalter/cmd/mem5
[14:05:42] Schalter/stat/RESULT {"Mem2":""}
[14:05:42] Schalter/stat/RESULT {"Mem3":""}
[14:05:42] Schalter/stat/RESULT {"Mem4":""}
[14:05:42] Schalter/stat/RESULT {"Mem5":""}
[14:05:45] Schalter/tele/STATE {"Time":"2019-10-04T13:05:45","Uptime":"0T06:18:15","Vcc":3.514,"SleepMode":"Dynamic","Sleep":50,"LoadAvg":19,"POWER":"OFF","Wifi":{"AP":1,"SSID":"JCQ-98FEE4","RSSI":"00:1F:17:17:70","Channel":9,"RSSI":86}}
[14:05:45] Schalter/tele/SENSOR {"Time":"2019-10-04T13:05:45","Switch1":"ON","Switch2":"ON"}
  
```



TDM ist ein mächtiges Werkzeug zur Konfiguration und scheint mir zur Zeit das umfangreichste Management-Tool zu sein.