| Plugin-Daten | |
|---|---|
| **Autor** | Michael Schlenstedt |
| **Logo** |  |
| **Status** | STABLE |
| **Version** | 2.3.0 |
| **Min. LB Version** | 2.2.1 |
| **Release Download** | https://github.com/mschlenstedt/LoxBerry-Plugin-1-Wire-NG/archive/refs/tags/1-Wire-NG-2.3.0.zip |
| **Pre-Release Download** | https://github.com/mschlenstedt/LoxBerry-Plugin-1-Wire-NG/archive/refs/tags/1-Wire-NG-2.5.0.zip |
| **Beschreibung** | Das Plugin kann einen an den LoxBerry angeschlossenen 1-Wire-Busmaster auslesen und die Sensorwerte an den Miniserver senden. |
| **Sprachen** | EN, DE |
| **Diskussion** | https://www.loxforum.com/forum/projektforen/loxberry/plugins/227489-plugin-1-wire-ng |

# 1-Wire-NG

Version History...

https://github.com/mschlenstedt/LoxBerry-Plugin-1-Wire-NG/releases

## Download

Direkter Download-Link: Siehe *Tabelle oben*

Letzter Entwicklungsstand im Repo: https://github.com/mschlenstedt/LoxBerry-Plugin-1-Wire-NG

## Funktion des Plugins

Das Plugin liest mit Hilfe der Software OWFS (https://github.com/owfs/owfs oder https://www.owfs.org/) den 1-Wire-Bus aus und gibt die Werte per MQTT-Protokoll an einen MQTT-Broker weiter. Dabei können unterschiedliche 1-Wire-Busmaster an den LoxBerry angeschlossen werden. Es werden USB-, serielle (LinkUSB) und I2C-Busmaster unterstützt. Auch der in den LoxBerry per GPIO integrierte Busmaster wird unterstützt. Die Anzahl an Busmastern ist nur durch die verwendete Hardware begrenzt.

Als MQTT-Broker empfiehlt sich das MQTT Gateway Plugin, welches die Werte direkt an den Miniserver senden kann.

Standardmäßig unterstützt das Plugin folgende Sensoren: DS2405, DS18S20, DS1920, DS2406, DS2407, DS2423, DS2450, DS1921, DS1822, DS2438, DS18B20, DS2408, DS2413, DS18B25. Neue (unbekannte) Sensoren können per individueller Konfiguration ganz einfach selbst hinzugefügt werden.

Über einen Watchdog wird die Funktion des Plugins permanent überwacht. Bei Problemen wird versucht das Plugin neu zu starten.

# Konfiguration

## 1-Wire / OWFS

Hier konfiguriert man die Software OWFS, die für die 1-Wire-Kommunikation zuständig ist. Über den Button "OWFS WebUI" unten auf der Seite gelangt man direkt zum WebUI von OWFS.

Wähle zunächst aus, welche Busmaster Du mit dem Plugin verwenden möchtest. Du kannst so viele Busmaster an den LoxBerry anschließen, wie es die Leistung Deiner Hardware erlaubt. Auch eine gemischte Installation z. B. aus seriellen und USB-Busmastern ist möglich. **Speichere die Einstellungen JETZT ab und starte danach den LoxBerry neu!**


Manche Busmaster können mehrere Busse bedienen. Wähle im nächsten Schritt aus, welche Busse Du verwendest. Gibt es nur einen Bus, dann aktiviere diesen! Nur an aktiven Bussen wird nach Sensoren gesucht! Deaktiviere aber Busse, die Du nicht verwendest, um Ressourcen zu sparen. Über den Button "Show" kannst Du Dir alle gefunden Busse und deren Bezeichnung anschauen.


Im nächsten Schritt werden die verwendeten Ports der OWFS-Software (owserver und owhttpd) festgelegt. Ändere hier nur etwas, wenn Du unbedingt musst!


Im nächsten Schritt werden einige Default-Werte festgelegt, die für alle angeschlossenen Sensoren gelten. Du kannst dann später für einzelne Sensoren spezielle Werte (z. B. kürzere Abfrageintervalle) festlegen (siehe unten).

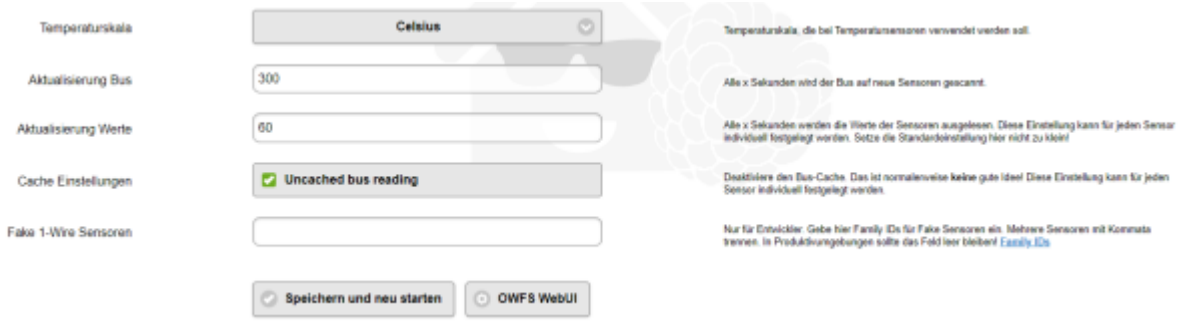
Zunächst kannst Du auswählen, in welcher Einheit bei Temperatursensoren die gemessenen Werte zurückgemeldet werden (Celsius oder Fahrenheit).

*Refresh Interval Devices* legt fest, in welchem Abstand nach neuen Devices auf dem Bus gesucht werden soll und wie häufig die Konfiguration (z. B. Namen von Sensoren, siehe unten) eingelesen werden. Normalerweise brauchst Du hier keine kürzeren Intervalle als 5 Minuten (300 Sekunden). *Default Refresh Interval Values* legt fest, in welchen Abstand die Sensoren auf dem Bus abgefragt werden sollen. Achte darauf, dass Du bei sehr kurzen Intervallen den Bus stark belastest! Auch macht

es wenig Sinn, z. B. Temperatursensoren sekündlich abzufragen. So schnell ändert sich die Temperatur in aller Regel nicht! Auch diese Angabe ist in Sekunden. Standard ist ein Abfrageintervall von 60 Sekunden. Laß es am Besten so und konfiguriere später nur einzelne Sensoren, bei denen Du einen kürzeren Intervall benötigst, individuell (siehe unten).

*Default Cache Setting* legt fest, ob die Software OWFS seinen internen Cache verwenden soll. Auch damit soll der 1-Wire-Bus entlastet werden. Du solltest standardmäßig den Cache **deaktivierten** (also diese Option anhaken). Insbesondere wenn Du iButton schnell auslesen möchtest, muss diese Option aktiviert werden! Nähere Informationen findest Du hier: What is Uncached?

Bei *Fake 1-Wire-Devices* können Fake-Sensoren aktiviert werden, die von der Software nur simuliert werden. Das ist in aller Regel nur für Entwickler interessant! Gebe einfach die FamilyID der Sensoren ein, die Du simulieren möchtest. Mehrere Sensoren werden durch Kommata getrennt. Die Sensoren erscheinen in der hier angegebenen Anzahl auf dem Bus 0 und ändern ständig ihre Werte, damit man sie auslesen kann.




Nach Abschluss der Konfiguration wird mit "Speichern und neu starten" die Konfiguration gespeichert und OWFS neu gestartet. Wenn Du einen seriellen Busmaster oder den GPIO Busmaster verwendest und erstmals aktiviert hast, ist ein Neustart des LoxBerry erforderlich.

## Sensoren

Im Plugin muss einmalig dieser Tab aufgerufen werden, damit nach neuen Sensoren gescannt wird! **Vorher startet der Dienst "OWFS2MQTT nicht!**

Hier kannst Du für jeden angeschlossenen Sensor auf dem Bus eine individuelle Konfiguration (z. B. das Abfrageintervall) festlegen und zudem einen eindeutigen Namen für jeden Sensor vergeben. Der Name wird auch im MQTT-Topic verwendet. So kannst Du später einen Sensor austauschen (der dann eine andere Seriennummer hat), ohne dass Du Deine Konfiguration in LoxoneConfig anpassen musst.

Für den Start verbinde bitte alle Sensoren mit dem 1-Wire-Bus (auch alle iButton oder DS2401 von z. B. Fenstersensoren) und klicke auf *Scan for Devices*. iButton kannst Du auch nacheinander an den Bus hängen und dann jeweils wieder "*Scan for Devices*" aktivieren. Einmal erkannte Sensoren werden nicht wieder aus der Konfiguration gelöscht.

Über die Button rechts kannst Du einzelne Sensoren löschen und die individuelle Konfiguration aufrufen.

| Name | Address | Type | Custom Settings | Uncached Reading | Refresh | Check Presents | Values | Actions |
|---|---|---|---|---|---|---|---|---|
| CustomDevice | 10.5BB221C2C59A | DS18S20 | Yes | No | 1 | No | temperature9 | |
| 10.8E53FD908C0D | 10.8E53FD908C0D | DS18S20 | No | No | 60 | No | | |
| 26.896D741BB3EF | 26.896D741BB3EF | DS2438 | No | No | 60 | No | | |

Zunächst solltest Du für jeden Sensor einen eindeutigen Namen eingeben. Damit wird der Name anstelle der Seriennummer im MQTT-Topic verwendet und Du kannst so später einen defekten Sensor ganz einfach austauschen, ohne dass Du die LoxConfig dazu anpassen musst.

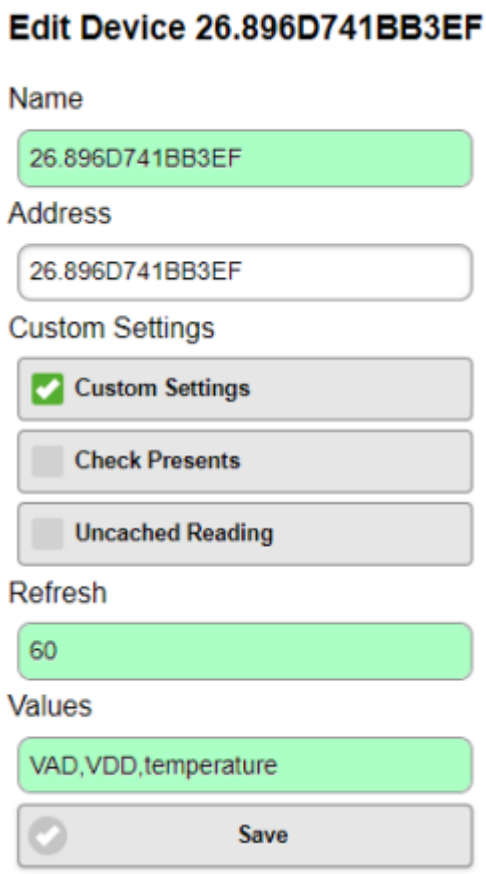
Du kannst auch (beim Austausch eines Sensors) einfach eine neue Seriennummer (Address) bei einem bestehenden Sensor eingeben. Damit wird die Konfiguration direkt für den neuen Sensor übernommen. Die Eingabe erfolgt dabei mit der bei OWFS üblichen Punkt-Schreibweise FAMILYID.ADDRESS, also z. B. 10.5BB221C2C59A.

Für eine individuelle Konfiguration aktiviere *Custom Settings*.

Mit *Refresh* kannst Du nun ein individuelles Abfrageintervall für diesen Sensor festlegen und z. B. auch *Uncached Reading* nur für diesen Sensor aktivieren. Für iButtons empfehle ich Uncached Reading und ein Abfrageintervall von 0.1 Sekunden.

Möchtest Du bei jedem Abfrageintervall überprüfen, ob der Sensor gerade am Bus hängt, aktiviere *Check Presents*. Das ist z. B. bei iButtons sinnvoll (und auch standardmäßig für iButtons schon aktiviert).

Unter *Values* gibst Du die Werte des Sensors ein, die Du abfragen möchtest. Mehrere Werte werden durch Kommata getrennt. Welche Werte für den Sensor verfügbar sind, kannst Du entweder direkt über die OWFS WebUI herausfinden (siehe 1-Wire / OWFS) oder auf der folgenden Webseite nachlesen: https://github.com/owfs/owfs-doc/wiki/1Wire-Device-List




### MQTT

Hier werden die MQTT-Einstellungen konfiguriert. Das Plugin setzt zwingend ein installiertes MQTT Gateway Plugin voraus (nur LoxBerry 2.0) - bei Loxberry 3.0 ist das MQTT Gateway bereits im Core enthalten. Sämtliche MQTT Einstellungen werden dort konfiguriert. Auf der Plugin-Einstellungsseite kann man daher lediglich das zu verwendende MQTT Topic konfigurieren. Der gewählte Topic wird automatisch im MQTT Gateway Plugin bzw. ab LoxBerry 3.0 im Widget MQTT registriert.

### Weitere Artikel

Hier findest Du weitere detailliertere Beschreibungen zu einigen Themen:

- Einbau i2c
- Einbau Onboard
- Einbau USB

## FAQ

**F:** Kann ich die Sensoren überwachen und bei Bedarf vom Miniserver aus die Services neu starten?

**A:** Ja, das geht. Überwache dazu das Topic `owfs/keepaliveepoch` und/oder `owfs/plugin` . Wie das geht findest Du im LoxWiki. Die Pluginservices kannst Du mit folgendem Befehl im Browser oder einem Virtuellen Ausgangsbefehl neu starten:

```
http://loxberry:password@loxberry/admin/plugins/1-wire-ng/index.cgi?ajax=restartservices
```

# Roadmap

- Deutsche Übersetzung
- Werte per MQTT auf den Bus schreiben für Aktoren (aktuell nur Lesen möglich)
- Template Builder

# Links

- Unterstütze Sensoren: https://github.com/owfs/owfs-doc/wiki/1Wire-Device-List
- OWFS Wiki: https://github.com/owfs/owfs-doc/wiki
- OWFS Homepage: https://owfs.org
- Erklärung Cache-Funktion von OWFS: https://owfs.org/index_php_page_what-is-uncached.html
- 1-Wire im LoxWiki: 1-Wire Melder und Sensoren
- Sehr hilfreiche Übersicht zum Aufsetzen und Analysieren von größeren und verlässlichen 1 Wire Netzwerken (pdf): Guidelines for Reliable 1-Wire Networks

# Fragen stellen und Fehler melden

Im Loxforum in diesem Thread:
https://www.loxforum.com/forum/projektforen/loxberry/plugins/227489-plugin-1-wire-ng

From:
https://wiki.loxberry.de/ - **LoxBerry Wiki - BEYOND THE LIMITS**

Permanent link:
**https://wiki.loxberry.de/plugins/1_wire_ng/start?rev=1736356194**

Last update: **2025/01/08 18:09**