Plugin-Daten	
Autor	Oliver Engel
Logo	
Status	STABLE
Version	0.09
Min. LB Version	1.0
Release Download	https://foshkplugin.phantasoft.de/files/loxberry-FOSHKplugin.zip
Pre-Release Download	https://foshkplugin.phantasoft.de/files/loxberry-FOSHKplugin-0.0.10Beta.zip
Beschreibung	Dieses Plugin bindet verschiedene Wetterstationen des Hersteller Fine Offset Electronics (FOSHK) an einen Loxone-Miniserver (oder beliebige andere Zielsysteme) über UDP an. Entwickelt wurde das Plugin für und mit einem Froggit DP1500 das baugleich auch unter dem Namen Ecowitt GW1000 verkauft wird.
Sprachen	EN, NL, SK, DE
Diskussion	https://www.loxforum.com/forum/projektforen/loxberry/plugins/222662

FOSHKplugin

Version History...

Version 0.10 - 04.07.2023 - öffentlicher Beta-Test

- changed: maxdailygust is now maxdailygustkmh in metric data (kmh) - the unit of maxdailygust is mph - **ATTENTION! may cause compatibility issues!**
- fixed: typo in generic-FOSHKplugin-install.sh fixed sytemd –> systemd
- fixed: UDP command Plugin.intvlwarning=enable/disable triggered co2warning instead (copy/paste-Error)
- fixed: min/max values for leaf moisture sensors 1-8 were not written to the daily CSV
- fixed: WS90 rain-handling - replacing the new names with the old ones now works correctly (e.g. „rrain_piezo“ –> „rainratein“)
- fixed: scanWS MAC address output
- fixed: Sending of additional temp/hum sensors now according to WeatherCloud API documentation: temp02/hum02
- fixed: sunhours calculation - formula corrected like WeeWX extension (Jterrettaz)
- fixed: modifications for altered AWEKAS-API (response OK)
- fixed: possible error in the creation of the daily CSV fixed
- fixed: LoxBerry - piezo variable names in Loxone template fixed (added mm to the name)
- improved: improved 10 minutes average calculation „winddir_avg10m“ (avgWind)
- improved: ignore „@“ in the field identifier of OUT_TEMP and OUT_HUM to standardise the assignments as with FWD_REMAP
- improved: Pushover notification for missing weather station changed to red text color
- improved: existing values can now be assigned via FWD_REMAP to the WeatherCloud keys co, no, no2, so2, o3, tempagro, et, pwrsply, battery and noise
- improved: sending pm25 & aqi of WH41/43 channel #1 to WeatherCloud if no WH45 present

(WH45 still preferred)
- improved: conversion to WU: tempNf, humidityN, qcStatus, softwareType
- improved: better error handling for saving as file via ftp(s)
- improved: InfluxDB: missed forwards are queued and transmitted when the destination is available again
- improved: error handling for Weathercloud forwards
- improved: error handling for Pushover push notifications (automatic retry)
- improved: parameter -scanWS gathers all weather stations now - including GW2000
- improved: WH45 for WU - if no WH41 present, use data from WH45 instead
- improved: internal functions stringToDict & getfromDict
- improved: if the structure of the daily CSV is changed, a new file is created
- new: introducing REBOOT_WARNING - check wether current runtime < last runtime - if so: warn (log, push, UDP: rebootwarning=1 dailyboot=#count)
  can be deactivated with http: /FOSHKplugin/rebootwarning=disable and UDP: Plugin.rebootwarning=disable
  LoxBerry: commands are included as virtual outputs in the template
- new: „dailyboot“ (number of restarts) and „rebootwarning“ as keys, reset at midnight and is also part of the daily CSV (if EVAL_VALUES active)
  LoxBerry: Keys „rebootwarning“ and „dailyboot“ are included as virtual inputs in the template
- new: „sunshine“ represents the presence of sunshine; can be artificially prolonged with SUNSHINE_HOLD to prevent constant changes
  LoxBerry: key „sunshine“ also included in template as virtual input
- new: introducing SUNSHINE_HOLD = seconds - Hold time in seconds for value sunshine, this time continues to be output sunshine = True, even if there is no sunshine (default: 0)
- new: FWD_TYPE = MIYO; sends temperature, wind and rain state (rain = rainratemm > 0 or hourlyrainmm > 1 or dailyrainmm > 1) to a MIYO cube (irrigation system)
  rain forecast is still missing; could be implemented too via API AerisWeather or AccuWeather (both in test)
- new FWD_TYPE = INFLUX2MET and INFLUX2IMP - support of InfluxDB2 - Python 3.6 or later and Python lib influxdb-client are required!
  bucket = dbname; org = fwd_sid; token = fwd_pwd
  missed forwards are queued and transmitted when the destination is available again
- new: (LoxBerry) virtual input FOSHK-getMinMax for command Plugin.getminmax integrated in Loxone template
- new: send the fwd_type, prgname & prgver as an argument with postFile
- new: FWD_WARNING, FWD_WARNINT, http://ipaddress:portnumber/FOSHKplugin/fwdstat, enable/disable via http & UDP
- new: support for leafwetness sensors for Ambient Weather stations
- new: change the date/time output format via Config\DT_FORMAT (default = %d.%m.%Y %H:%M:%S - dd.mm.yyyy hh:mm:ss) for all (!) date/time outputs: log, csv, push
- new: forward warning via Pushover
  after a configurable number of successive unsuccessful attempts, a push message is sent through Pushover
- new: statistics page for forwards: http://ipaddress:portnumber/FOSHKplugin/fwdstat
- new: scan weather station page: http://ipaddress:portnumber/FOSHKplugin/scanWS
- new: missed forwards for forward types EW and RAWEW may be queued and sent when destination is available again (with FWD-xx\FWD_QUEUE = True)
- new: exclude list for battery warning: BATTERY_WARNEXCLUDE - a comma separated list of keys to exclude from battery warning e.g. wh90batt
- new: Awekas-API: deliver weather report conditions & tendency
  missed forwards are automatically queued as FOSHKplugin-queue/FWD-nr/FOSHKplugin-

queued-data-nr.csv in config dir (or dir configured as FWD_QDIR)
- new: Adaptations for installation under (Open)Suse, Synology, Fedora (zypper, ipkg, dnf) in generic version
- new: custom push notifications via Pushover (user-defined push messages when values exceed or fall below a definable value)
- new: Conversion to UTF-8 of all programme parts (may cause problems)
- new: Weather Report for Awekas - automatically report rain, storm and thunderstorm
- new: FWD_TYPE = BANNER - automatic generation of banners and stickers with current weather data
- new: further queryable key names for getvalue, JSON, ... and forward types MQTT, InfluxDB, BANNER and TAGFILE:
  prgname (FOSHKplugin)
  prgver (current version number)
  winddirtext (if available, 10min winddir-mean otherwise winddir as short text (N, NE, NNO, ...)
  aqtime (Time of processing by FOSHKplugin)
  pchange1in (1 hour pressure change in inHg)
  pchange3in (3 hour pressure change in inHg)
  lightningmi (lightning distance in miles)
- new: FWD_TYPE = TAGFILE - user-defined output format based on tags and templates
- new: config option Export\LIMIT_WINDGUST = n to prevent processing of unrealistic values for windgustmph and maxdailygust (e.g. for WS80/WS90)
  if value >= n the windgustmph will be renamed to _windgustmph (thus not processed); last "good" maxdailygust will be used as maxdailygust
- new: calculation of windrun (in miles) and windrunkm (in km) and daily solar radiation sum (srsum) - also included in Loxone template

Version 0.09 - 02.04.2022

- kleinere Fehlerbehebungen und Optimierungen
- intensives Code-Cleaning - Umbenennung und Vereinheitlichung der Konvertierfunktionen
- Fehler bei eingehenden Daten im WU-Protokoll und aktiviertem EVAL_VALUES behoben
- http-Anfragen mit &refresh=n aktualisiert die angezeigte Seite alle n Sekunden
  Beispiel: http://ipadresse:portnummer/APRS&refresh=30 aktualisiert die Anzeigeseite des APRS-Ausgabestrings alle 30 Sekunden
  oder: http://ipadresse:portnummer/status&minmax&refresh=60 zeigt die aktuellen Werte inkl. Status und Min/Max an und aktualisiert alle 60 Sekunden
- bei eingehenden Daten im WU-Format barominrelin mit baromrelhpa gleichsetzen, Konvertierung von WU nach EW für WH6006 modifiziert
- Kompatibilität zu GW1100 sichergestellt
- interner WU-Server: WN34- und WN35-Kompatibilität hergestellt
- bei Konvertierung nach Ambient Weather wird wh80batt korrekt auf battout gesetzt
- bessere Protokollierung für FWD_EXEC - FWD-Nummer wird für bessere Zuordnung protokolliert; Anzeige einer Änderung erfolgt nur bei tatsächlicher Änderung
- Blattfeuchte-Level für Meteotemplate, WC, Awekas, Weather365 und WSWin - statt 0..99 wird nun als Level 0..15 (float) gesendet
- alternative Namen für RAWEW (EWRAW), RAWUDP (UDPRAW), RAWCSV (CSVRAW), AMBRAW (RAWAMB) und TXTFILE (TEXTFILE) eingeführt
- in der FWD_URL kann nun für Ausgabeformate REALTIMETXT, CLIENTRAWTXT, CSVFILE, WSWIN, TXTFILE und RAWTEXT auch ein Dateiname übergeben werden
- bei Erzeugung der WSWin-CSV erfolgt nun auch per http(s)/POST und ftp(s) ein Anhängen neuer Daten an die bereits vorhandene Datei (append)

- neue, verbesserte Sonnenstundenberechnung sunhours (nach https://github.com/Jterrettaz/sunduration) mit dynamischen, ortsabhängigen Schwellwert (vielen Dank Werner!), erfordert Coordinates\LAT und Coordinates\LON
  ohne LAT/LON oder bei Sunduration\SUN_CALC = False wird die bereits bekannte Berechnung mit fixem Schwellwert von 120W/m² genutzt
  kann mit Sunduration\SUN_MIN (minimaler Schwellwert, default=0) und Sunduration\SUN_COEF (default=0.8 - zu wenig Sonnenschein erfasst: Wert verkleinern; zuviel: vergrößern) modifiziert werden
  aus Kompatibilitätsgründen muss diese Funktion mit Sunduration\SUN_CALC = True aktiviert werden
- html-Abfrage für WSWIN implementiert - http://ipadresse:portnummer/WSWIN gibt eine WSWin-kompatible Datenzeile der letzten Werte aus
- neuer Forward-Type EWUDP (UDPEW) - konvertiert eingehende EW-, WU- und AMB-Meldungen nach Ecowitt/UDP (etwa für Personal Weather Tablet/UDP broadcast listener)
- FWD_IGNORE zum Filtern aller ausgehenden Keys jetzt gültig für alle Forwards - Keys in dieser Liste werden nicht verschickt
- Remap-Funktion FWD_REMAP implementiert - Ausgabe-Keys können jetzt mit Werten aller bekannten internen Keys definiert werden
  Einige Ziele unterstützen nur eine Auswahl an Sensoren, Ambient Weather unterstützt z.B. nur einen internen/externen PM2.5-Sensor oder Awekas oder WSWin nur 4 Bodenfeuchtesensoren. FOSHKplugin überträgt jedoch immer logisch fortlaufend - beginnt also beim jeweils ersten Sensor und sendet die jeweils gültige max. Anzahl der Kanäle.
  Mit FWD_REMAP kann eine entsprechende Zuordnung bzw. Auswahl erfolgen.
  Beispiel: *FWD_REMAP = @tf_ch1c=@tf_ch8c* - setzt den metrischen Key tf_ch1c (entspricht metr. Temperaturwert 1. Kanal) auf den Wert von tf_ch8c (8. Kanal)
  dabei können Werte anderer Keys (mit @) und statische Werte zugewiesen (etwa @tf_ch1c=12.3) sowie eigene Keys definiert werden (soiltemp2=@tf_ch7c);
- neuer Forward-Typ APRS ermöglicht das Versenden der Daten an CWOP
  call sign wird als FWD_SID übergeben, ein ggf. nötiges Passwort kann mit FWD_PWD übergeben werden; FWD_URL enthält die Adresse:Port des Ziels
  kann auch per http mit http://ipadresse:portnummer/APRS?user=CALLSIGN abgerufen werden
- Weather365: Bodentemperaturen der Sensoren 2..4 werden nun auch übertragen - beachte das ggf. nötige Remappen!
- MeteoTemplate: Unterstützung von WN35 (Blattfeuchte) und WN34 als soil temp/TSn - die Tiefe kann mit TS0n=cm als ADD_ITEM oder per FWD_REMAP hinzugefügt werden
- MeteoTemplate: Batteriewerte von PM2.5-Sensoren werden nun statt mit PMnBAT mit PPnBAT ausgegeben
- Unterstützung des WS90-Sensors (wh90batt)
- Sollen Ausgabedaten zwar per Script (FWD_EXEC) verarbeitet, nicht jedoch versendet werden, kann dies mit einer Rückmeldung von "EXECONLY" aus dem Script realisiert werden: echo EXECONLY als letzter Ausgabebefehl im Script.
- Befehlserkennung für VI FOSHK-co2 in Loxone-Vorlage angepasst
- use Config::Simple; im webfrontend eingebaut (wegen Pre-Announcement for deprecation)

Version 0.08 - 27.06.2021

- kleinere Bugfixes
- erweiterte Debug-Möglichkeiten in der Loxone-Version (service.sh support und supzip)
- besseres Logging im Sende-Log: Nummer des Forwards wird mitprotokolliert um fehlerhaften Block im Config-File leichter finden zu können
- anpassbares Logging-Level - weniger Logging mit Blick auf das Wesentliche

über Logging\LOG_LEVEL im Config-File lässt sich nun das Logging feinjustieren - bei
- ALL werden wie bisher alle Zeilen protokolliert
- INFO - alle Zeilen außer ERROR, WARNING, INFO und OK werden ausgeblendet
- WARNING - alle Zeilen außer ERROR und WARNING und OK werden ausgeblendet
- ERROR - nur Zeilen mit ERROR und OK werden ausgegeben
aus Kompatibilitätsgründen ist ALL voreingestellt - ich empfehle jedoch LOG_LEVEL = INFO - somit wird alles, was nicht problemlos erfolgreich war, protokolliert

- LOG_LEVEL lässt sich auch im Betrieb per http://ipadresse:port/FOSHKplugin/loglevel=[ALL, INFO, WARNING, ERROR] anpassen - bei Neustart gilt jedoch wieder der im Config-File eingestellte Wert
- mit Logging\LOG_ENABLE = False lässt sich das Logging global abschalten, ohne Änderungen an den Logfile-Namen vorzunehmen
- flexiblerer Firmware-Update-Check (duplicate options)
- erhöhte Sende-Sicherheit bei http(s) und ftp(s)-Forwards durch Sendungswiederholung:
jetzt erfolgen 3 Versuche, die Daten zuzustellen; der zweite Versuch findet nach 5 und der dritte nach weiteren 10 Sekunden statt
die Wiederholung erfolgt jedoch nicht, wenn der Rückgabecode auf einen lokalen Fehler hinweist (400..499)
- Unterstützung des Forward per MQTT sowohl für metrische Werte (FWD_TYPE=MQTTMET) als auch für imp. Werte (FWD_TYPE=MQTTIMP)
erfordert python3-setuptools und paho-mqtt (wird automatisch installiert)
MQTT-Broker muss in FWD_URL definiert werden: ipaddress:port@hierarchy%prefix - topic name ist der Name des Keys
- Unterstützung der wetter.com-API (noch unter weewx-Pseudonym)
- Unterstützung der weather365.net-API
- Unterstützung der wettersektor.de-API
- Exportmöglichkeit der Daten als realtime.txt und clientraw.txt per http(s), ftp(s) und file mit FWD_TYPE = REALTIMETXT bzw. CLIENTRAWTXT
dabei legt die FWD_URL fest, ob die Datei per http(s) geposted, per ftp(s) übertragen oder als Datei abgelegt werden soll
ein realtime.txt-kompatibler String kann per http://ipadresse:portnummer/realtime.txt abgefragt und gespeichert werden
ein clientraw.txt-kompatibler String kann per http://ipadresse:portnummer/clientraw.txt abgefragt und gespeichert werden
- Speicherung als WSWin-kompatible CSV-Datei wswin.csv zum automatischen Import durch WSWin per Dateiüberwachung
der Speicherort muss per Samba-Freigabe für Windows-Rechner lesbar (ggf. schreibbar) sein
der Import kann durch WSWin automatisch und auch unregelmäßig erfolgen - WSWin liest einfach alle bisher noch nicht verarbeiteten Zeilen ein
dabei ist keine X-CSV nötig
- Exportmöglichkeit als Text- bzw. CSV-File mit enthaltenen Keynamen per file, http(s) und ftp(s) mit FWD_TYPE = TXTFILE bzw. CSVFILE - vorhandene Dateien werden überschrieben; diese Formate sind auch per http:ipaddress:port/FOSHKplugin.txt bzw. .csv abrufbar
- FWD_PWD kann für Forward über das Web-Interface eingegeben werden
- separator bei Ausgabe via /STRING und /RAW kann jetzt neben einem einzelnen Zeichen auch %20 (als Leerzeichen) oder ein ganzes Wort sein
- Berechnung der Wolkenhöhe cloudf (in feet) bzw. cloudm (in Meter) implementiert, Coordinates\ALT = xx im Config-File mit Höhe über NN in Metern nötig - erfordert EVAL_VALUES = True
- Sonnenscheindauer sunhours implementiert - zeigt die Dauer der täglichen Sonnenscheindauer in Stunden an (solarradiation >= 120W/m²) - erfordert EVAL_VALUES = True

- sunhours, co2 und leafwetness (für den kommenden WN34) in Meteotemplate unterstuetzt; sunhours auch bei Awekas-API
- Koordinaten können unter Coordinates\ALT, LAT, LON konfiguriert werden
  ALT wird zur Ermittlung der Wolkenhöhe genutzt (spread * 122)
  LAT/LON werden nur zur Übertragung in die Exportformate Awekas-API, clientraw.txt und Weather365.net genutzt und können somit auch leer bleiben
- Config\UDP_MAXLEN (default=2000) legt maximale Länge eines UDP-Datagrams fest
  Ist die Länge des zu sendenden Paketes größer als festgelegt wird das Paket in mehrere Datagramme aufgeteilt, die jeweils eine ungefähre Länge UDP_MAXLEN haben und den Identifier SID=FOSHKweather am Anfang jedes Datagrams enthalten.
  Dabei wird die Ursprungszeile aber so getrennt, dass die Zuordnung key=value erhalten bleibt - die Trennung erfolgt also immer **HINTER** UDP_MAXLEN beim nächsten Auffinden eines Leerzeichens.
  Werte mit enthaltenen Leerzeichen werden durch doppelte Anführungsstriche eingefasst, damit man auf UDP-Serverseite eine Möglichkeit zum Parsen hat.

Beispiel neighborhood="Hohen Neuendorf" bei einem Wert mit Leerzeichen - jedoch neighborhood=Berlin (ohne Leerzeichen)

- mit Warning\CO2_WARNING = True kann eine Warnung (UDP, http, Log, Pushover) aktiviert werden, wenn der CO2-Messwert höher als der unter Warning\CO2_WARNLEVEL konfigurierte liegt
- mit Config\UDP_STATRESEND = n kann eine Zykluszeit (n Sekunden) definiert werden, in der die Warnmeldungen unabhängig von Statusänderungen versandt werden
- Änderung bei ptrend1 und ptrend3 - bei starkem Anstieg oder Abfall des Luftdrucks (+0.7/-0.7 bzw. +2/-2) wird als Trend eine 2 bzw- -2 ausgegeben
- ptrend1, pchange1, ptrend3 und pchange3 werden nun auch per Ecowitt (Type=EW) weitergeleitet, wenn EVAL_VALUES = True
  pchange1 und pchange3 enthalten dabei die Differenz zwischen dem aktuellen Wert und dem Wert von vor 1 oder 3 Stunden in inHg
- mit CSV\CSV_DAYFILE = /path/to/filename.csv wird die Erstellung eines Tages-CSV /path/to/filename.csv mit den min/max-Werten des Tages aktiviert
- Parameter bool, status, units und separator sind nun bei allen Request möglich (soweit sinnvoll - bei RAW also nicht)
- Abfrage /JSON erweitert - erzeugt bei Abruf mit Parameter bool ein JSON mit numerischen/boolschen Werten - kann mit minmax, status und units erweitert werden
  Beispiel: http://ipadresse:portnummer/JSON?minmax&status&bool gibt den letzten Datensatz mit numerischen/boolschen Werten statt mit Strings aus
- MQTTMET/MQTTIMP-Forward gibt nun numerische/boolsche Werte aus
- InfluxDB-Unterstützung: sowohl per pull (als JSON via telegraf) als auch nativ (FWD_TYPE = INFLUXMET/INFLUXIMP) integriert
  die Angabe der Datenbank erfolgt in der FWD_URL-Zeile: FWD_URL = http://192.168.15.237:8086@Database erzeugt eine Datenbank Database auf 192.168.15.237:8086
  und überträgt die Daten unverschlüsselt im konfigurierten Intervall; Nutzername und Password können per FWD_SID und FWD_PWD übergeben werden
- neue Config-Optionen REBOOT_ENABLE und RESTART_ENABLE - ermöglichen den Neustart der Wetterstation und von FOSHKplugin per http/UDP
  die Aufrufe lauten http://ipaddress:port/FOSHKplugin/rebootWS zum Neustart der Wetterstation und http://ipaddress:port/FOSHKplugin/restartPlugin zum Neustart des FOSHKplugin-Service
- neuer Forward-Typ RAWTEXT ermöglicht die Ablage der eingehenden Werte der Wetterstation

als Textdatei lokal im Dateisystem und remote per http(s)/POST und ftp(s)
- Firmware-Update-Status (updatewarning) wird nun sofort nach Eingang neuer Daten der aktualisierten Wetterstation aktualisiert
- Anpassung für WH6006 (dateutc mit "%3A" statt ":", indoorhumidity)

Version 0.07 - 19.02.2021

- Fehler bei IGNORE_EMPTY behoben: UDP-Versand an Loxone funktionierte nicht, wenn IGNORE_EMPTY deaktiviert war
- Log-Ausgaben: "custom mode" umbenannt nach "custom server"
- sendet nun bei http-Empfang der Daten neben dem response-code 200 auch den Text OK zum Sender
- Fehlerbehebung: known issue bzgl. socket-Problemen und Chrome hoffentlich behoben - nach einem 5 Sekunden TimeOut sollte der socket wieder freigegeben werden
- Fehlerbehebung Gewitterwarnung (nicht jedes Gewitter wurde gemeldet)
- Fehlerbehebung: Programmfehler bei PM2.5-Werten oberhalb von 500 behoben
- Fehlerbehebung: Behandlung von %20 im Feld dateutc (etwa von der WH2600 LAN) eingeführt
- Config-parsing hinsichtlich Boolean-Werten robuster gestaltet (mkBoolean)
- Forwards können nun im Config-File aktiviert/deaktiviert (FWD_ENABLE=True/False) und kommentiert werden (FWD_CMT)
- Multi-Instanz: mehrere Instanzen von FOSHKplugin können nun parallel - in unterschiedlichen Verzeichnissen - betrieben werden
- Unterstützung des Ambient Weather-Formats sowohl für eingehende Nachrichten als auch als Forward (AMB/RAWAMB)
  bei Fehlen von yearlyrainin wird totalrainin und bei Fehlen von rainratein wird hourlyrainin genutzt
- Forward der Eingangsdaten im Weathercloud-Format per GET als Typ WC möglich
- Forward der Eingangsdaten im Meteotemplate-Format per GET als Typ MT möglich
- Forward der Eingangsdaten im Awekas-Format per GET als Typ AWEKAS möglich
- Unterstützung des WH45 (PM25, PM10, CO2-Sensor) mit zusätzlicher AQI- und CO2-Level-Berechnung
- Vorbereitung für den neuen Blattfeuchtesensor WN35
- Gewitterentwarnung: Anzahl der Blitze (lcount) sowie min. und max. Entfernung (ldmin und ldmax) werden übermittelt
- Verbesserung hinsichtlich TimeOut-Verhalten; http hat nun einen TimeOut von 8 und UDP von 3 Sekunden
- Ecowitt-Forward: ist totalrain vorhanden - yearlyrain aber nicht, wird yearlyrain automatisch mit Wert von totalrain gesetzt
- neue Konfigurationsoption Export\OUT_TIME = True setzt Zeitstempel eingehender Nachrichten von der Wetterstation auf Empfangszeit
- fake-Modus nun auch für eingehende Nachrichten im WU- und Ambient-Format aktiviert
- ein automatischer Restart des FOSHKplugin-Dienstes bei ausbleibenden Daten der Wetterstation über Warning\WSDOG_RESTART konfigurierbar
- wichtige Status-Mitteilungen können nun zusätzlich per Pushover übermittelt werden (Update-, Sensor-, Watchdog-, Batterie-, Sturm- und Gewitterwarnung)
- generic: Anzeige aller erkannten Wetterstationen während Installation via generic-FOSHKplugin-install.sh sowie bei Start von ./foshkplugin.py -scanWS
- Mittelwertberechnung (bei EVAL_DATA=True) für Wind und Windrichtung implementiert, gibt windspdmph_avg10m, winddir_avg10m aus
- Böen-Maximum der letzten 10 Minuten implementiert: windgustmph_max10m/windgustkmh_max10m gibt die max. Böe in den letzten 10 Minuten aus

- windspdkmh_avg10m und winddir_avg10m in Loxone-Vorlage integriert
- Ausgabe des Lux-Wertes (solarradiation*126.7) als Feld brightness (bei EVAL_DATA=True)
- mit Logging\IGNORE_LOG lassen sich Zeilen vom Logging im Standard-Log ausnehmen (Komma-getrennte Liste von Suchworten) - etwa crondaemon
- mit FWD_EXEC lässt sich bei jedem Forward ein Script angeben, dass mit dem Ausgabestring als Parameter gestartet wird und dessen letzte Ausgabezeile als neuer Ausgabestring für den Versand übernommen wird
- Wert-Abfrage: ein http-Request http://ipadresse:portnummer/getvalue?key=[key] gibt den Wert für den Schlüssel [key] aus, wobei der Schlüssel der RAW-Schlüssel wie auch der umgewandelte Schlüsselname sein darf
  Beispiel: curl http://192.168.15.236:8080/getvalue?key=windspeedmph gibt den Wert "1.34" des Schlüssels windspeedmph aus
  in Verbindung mit FWD_EXEC können somit Daten anderer Instanzen abgefragt und eingebunden werden
- zusätzlich zur http-GET-Abfrage /CSV und /CSVHDR mit dynamischer Feldeinteilung wird nun auch /SSV und /SSVHDR mit statischen Feldern unterstützt, Grundlage ist die Feldbeschreibung in CSV\CSV_FIELDS
  da die Blitz-Daten im GW1000/DP1500 nicht im NVRAM gespeichert werden, gehen diese Werte bei einem Neustart des Geräts verloren
  FOSHKplugin speichert Zeit und Entfernung des letzten übertragenen Blitzes im Config-File ab und nutzt diese, wenn keine Werte vom Blitzsensor übermittelt werden
  diese Werte werden allen Ausgangsformaten zur Verfügung gestellt
- wird Warning\LEAKAGE_WARNING aktiviert, erfolgen Warnungen bei per Log, UDP, http oder Pushover (leakwarning) wenn eine Leckage erkannt wurde
- bei jedem erfolgreichen Start wird ein Backup der foshkplugin.conf angelegt (foshkplugin.conf.foshkbackup)

Version 0.06 - 02.08.2020

- Zeitpunkt des letzten Blitzereignisses lightning_loxtime mit in Loxone-Vorlage integriert
- Gewittererkennung/-warnung implementiert - sendet bei vorhandenen Blitzsensor WH57/DP60 Gewitterwarnung
- Überarbeitung der Sturmwarn-Funktion, Ausgabe der Luftdrucktendenz 1h/3h sowie Änderung des Luftdrucks 1h/3h
- WU-Forward/JSON von AqPM2.5 wenn Feinstaubsensor vorhanden (nur pm25_ch1 wird weitergeleitet!)
- AQI-Berechnung bei EVAL_DATA=True und vorhandenem DP200/WH41/WH43 aktiviert
- Batterie-Warnung per Log und UDP implementiert; fällt der mitgelieferte batt-Wert unter einen intern definierten Schwellwert, erfolgt eine Warnung
- Fehler bei Berechnung der korrekten Uhrzeit (Sommerzeit) bei W4L-Export behoben
- spanische Himmelsrichtungen für W4L-Export hinzugefügt
- Patch-Funktion für W4L: Wetterstationswerte "-9999" werden ebenfalls nicht importiert
- Fehler bei Ausgabe des Namens des wieder Daten liefernden Sensors (SENSOR_MANDATORY) behoben
- Status der Warnungen für Sturm, Gewitter, Sensor und Batterie werden zwischengespeichert und sind somit remanent
- WU-Forward/JSON Umbenennung von solarRadiation zu solarradiation
- WU-Forward/JSON Unterstützung von Bodenfeuchtesensoren
- WU-Forward: keys mit leerem value werden nicht übermittelt
- WU-Forward: Upload von dewptf (war dewpt) und rainin (war rainratein) repariert
- neue Formel für Taupunkt-Berechnung (dewpoint) aktiv (erfordert math)

- sendet nun bei http-Empfang der Daten einen response-code 200 zum Sender
- UDP-Nachricht für time bei wswarning von "time: " auf "time=" geändert
- bei allen get/post-Aktionen: Prüfung des Rückgabewertes 200..202 –> ok (war 200)
- Text-Fehler in Hilfe behoben
- experimentell: Forward des PM2.5-Wertes zu luftdaten.info als Typ LD, Angabe der ID unter FWD_SID im Config-File nötig
- Forward der Eingangsdaten ohne Konvertierung per UDP als Typ RAWUDP möglich
- Forward der Eingangsdaten ohne Konvertierung per EW/POST als Typ RAWEW möglich
- Forward der Eingangsdaten ohne Konvertierung per POST als Typ RAWCSV möglich
- Forward der Ausgabedaten als CSV als Typ CSV möglich
- Forward der Ausgabedaten per UDP an weitere Ziele mit FWD_TYPE=UDP möglich, dabei wird auch der Status übertragen
- Timeout-Handling bei Forward angepasst (nun 3 Sekunden)
- Sprachfassung NL und SK integriert
- Ausgabesprache kann per LANGUAGE=DE/EN etc. im Config-File eingestellt werden (generic)
- virtuelle Ausgänge für Shutdown, getStatus, debugOn und debugOff implementiert
- Separator bei http-GET /RAW auswählbar
- neue http-GET-Ausgabe /STRING zur Ausgabe der Ausgabezeile mit wählbaren Separator
- im /JSON und bei der Ausgabe per /STRING und /UDP können nun auch die Statusmeldungen abgefragt werden
- einfache Authentifizierung per AUTH_PWD implementiert; Daten und Anfragen werden per http nur angenommen, wenn das hier angegebene Password in der URL enthalten ist (im Ecowitt-Modus empfiehlt sich dafür der Wert von PASSKEY)

Version 0.05 - 26.04.2020

- Sturmwarnung bleibt für 60 Minuten nach letzter Grenzwertunter-/überschreitung aktiv; Zeitraum kann via STORM_EXPIRE im Config-File angepasst werden
- Übermittlung des UV-Wertes im WU-Format angepasst, nun in Großbuchstaben UV= statt uv=
- Patch-Funktion für Weather4Loxone ist nun unabhängig von der genutzten Weather4Loxone-Version (vorhandene fetch.pl wird nicht überschrieben sondern angepasst)

Version 0.04 - 20.02.2020

- default-config angepasst - Kommentare hinter Block nicht zulässig!
- verbesserte Buttons (CSS) - Schiebeschalter nun grau bei "off" und grün bei "on"
- erweiterte CGI-Debug-Funktion; default: off; enable mit $myDebug = 1 in der index.cgi
- myDebug für zusätzliche Debug-Informationen auch im Python-Programm implementiert (default: False)
- Beschreiben der Wetterstation via WS-Set sollte nun (endlich) vollumfänglich funktionieren
- Id & Key in den Einstellungen der Wetterstation werden ignoriert und nicht vom Plugin überschrieben

Version 0.03 - 18.01.2020

- USE_METRIC wieder funktional (jetzt also auch imperiale Werte per UDP und CSV möglich)
- weitere mögliche Probleme beim Setzen der Wetterstationsparameter via WS-Set behoben (Path wird nun immer auf defaults gesetzt)
- Prüfung der nutzbaren LoxBerry-Ports (http/udp) optimiert
- Kommunikation mit der Wetterstation überarbeitet - nun jeweils 5 Versuche bei Lesen und Schreiben
- besseres Logging/Debugging bei Fehlern bei Set-WS; "buntere" und besser parse-bare Log-Files;

entfernt
- generic: conf-File - Vorlage und Hilfstexte überarbeitet
- Ignorierliste Forward\FWD_IGNORE für Forwards eingebaut: definiert - kommasepariert - Felder, die NICHT verschickt werden sollen
- Forward\FWD_TYPE=UDP/EW/RAW für http-Forward der Werte (UDP-Ausgabezeile) an andere Ziele als WU eingeführt
- nun bis zu 10 Forwards mit unterschiedlichen Einstellungen möglich (aktuell nur im Config-File zu pflegen: Forward-1..9 analog zu Forward)
- Watchdog: kommen seit 3*eingestelltem Intervall keine Werte von der Wetterstation, Fehler melden!
  es erfolgt EINE Warnung und bei erneuter Übermittlung der Wetterstation eine Entwarnung im Log sowie per UDP:
  SID=FOSHKweather wswarning=1 last=346611722
  SID=FOSHKweather wswarning=0 last=346616459
  standardmäßig aktiv; kann im Config-File deaktiviert werden: Warning\WSDOG_WARNING=False
  Intervall kann im Config-File eingestellt werden: Warning\WSDOG_INTERVAL=3

Warnung auch in Loxone-Vorlage enthalten

- Alarm senden (Log, UDP) wenn Sensor (auch mehrere) keine Daten liefert (etwa weil Akku/Batterie leer)
  SID=FOSHKweather sensorwarning=1 missed=wh65batt time=347196201
  SID=FOSHKweather sensorwarning=1 back=wh65batt time=347196201
  aktuell nur im Config-File zu pflegen:
  Warning\SENSOR_WARNING=True sowie Warning\SENSOR_MANDATORY="wh65batt"
- Sturmwarnung: fällt oder steigt der Luftdruck um mehr als 1.75 Hektopascal in einer Stunde, erfolgt eine Warnung vor Starkwind/Sturm
  vgl. http://www.bohlken.net/luftdruck2.htm
  es erfolgt EINE Warnung und bei Entspannung des Luftdrucks eine Entwarnung im Log und per UDP:
  SID=FOSHKweather stormwarning=1 time=346611722
  SID=FOSHKweather stormwarning=0 time=346616459
  standardmäßig aktiv; kann im Config-File deaktiviert werden: Warning\STORM_WARNING=False
  WarnDiff kann im Config-File eingestellt werden: Warning\STORM_WARNDIFF=1.75
  Warnung auch in Loxone-Vorlage enthalten
- Vorbereitung Wassersensor WH55 und Blitzsensor WH57 (noch unklar ob lightning_time = timestring oder unixtime!)
- preupgrade-Script: Upgrade-Verzeichnisse werden nun auch ohne Elternverzeichnis angelegt (mkdir -p)
- preuninstall-script entfernt; Deinstallation erfolgt bei LoxBerry ab v2.0.1.1 im uninstall-Script
- Web-Oberfläche: Anzeige der Versionsnummer eingebaut (um Nachfragen zur verwendeten Version im Fehlerfall zu minimieren)
- UDP-Versand an das Zielsystem lässt sich mit UDP_ENABLE=False abschalten
- Ignorierliste für den UDP-Versand eingeführt: Config\UDP_IGNORE (nur im Config-File zu pflegen)

Version 0.02 - 28.12.2019

- ### aus FWD-Log-Nachricht entfernt
- Umrechnung temp1f in temp1c für Innensensor auf Kanal 1 implementiert
- Timeout bei sendReboot, setWSconfig und getWSINTERVAL von 1 auf 2 Sekunden erhöht (somit sollte WS-Set sicherer funktionieren)

- Probleme beim Setzen der Wetterstationsparameter via WS-Set behoben (Id & Key werden - wenn nicht schon vorhanden - gesetzt)
- Umstellung der LoxBerry-Versionsnummerierung damit zukünftig die Auto-Update-Funktion greifen kann

Version 0.01 - 15.12.2019

- erste öffentliche Version

Download

https://foshkplugin.phantasoft.de/files/loxberry-FOSHKplugin.zip (aktuelle stable-Version)
https://foshkplugin.phantasoft.de/files/loxberry-FOSHKplugin-0.0.10Beta.zip (aktuelle öffentliche Beta)

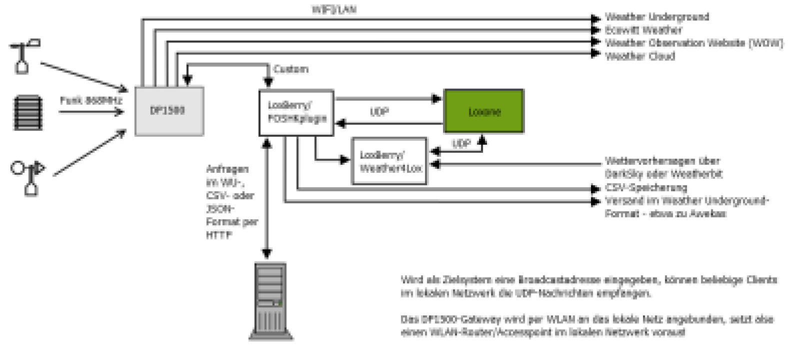
Funktion des Plugins

Dieses Plugin bindet verschiedene Wetterstationen und -sensoren des Hersteller Fine Offset Electronics (FOSHK) an einen Loxone-Miniserver über UDP an. Unterstützt werden alle Geräte, bei denen sich ein eigener Server als Ziel zur Übermittlung der Daten im WU- oder Ecowitt-Format einrichten lässt.

Funktionen:

- nimmt http-Nachrichten einer Wetterstation (DP1500, GW1000, HP1000SE, Sainlogic 7 in 1, ELV WS980WiFi, Eurochron EFWS 2900, ???) im WU- oder Ecowitt-Protokoll **lokal** über WLAN entgegen
- erfordert keine Cloud-Dienste oder Internetverbindung
- sendet per UDP die umgerechneten Werte an einen beliebigen Host oder per Broadcast ins Netz weiter
- kann empfangene Werte per MQTT weiter senden
- speichert auf Wunsch die umgerechneten Daten sortiert und/oder extrahiert als CSV
- ermöglicht das Weiterversenden an bis zu 50 Server, die von der Station selbst nicht unterstützt werden (etwa Awekas, PWSWeather, Windy, wetter.com, weather365.net, Ambient Weather oder Luftdaten.info)
- kann als Ecowitt-Relay dienen (Forward im Ecowitt-Protokoll - etwa für weewx oder PWS Dashboard oder Personal Weather Tablet oder andere Software, die Daten im Ecowitt-Format erfordert)
- kann eingehende WU- , Ecowitt- und Ambient Weather-Nachrichten per UDP weiterleiten
- kann Anfragen im WU-Protokoll beantworten
- integrierter Webserver liefert per http den jeweils letzten Datensatz im UDP-, CSV-, RAW- und JSON-Format sowie als Webseite
- erzeugt auf Wunsch eine Loxone-Vorlagendatei mit allen virtuellen In- und Outputs
- stellt dem Plugin Weather4Loxone die Messwerte der lokalen Wetterstation direkt bereit
- Anbindung an beliebige Datenbanksysteme über telegraf möglich
- direkte Unterstützung von InfluxDB
- erzeugt auf Wunsch eine Import-Datei für den automatischen Import der Daten in WSWin
- für den Loxone-Betrieb ist keine weitere Software nötig (WS View nur zum Anlernen neuer

Sensoren oder zur Konfiguration der Standard-Weiterleitungsdienste)
- funktioniert auch ohne Loxone/LoxBerry als systemd-Dienst zur Anbindung anderer Systeme (generic FOSHKplugin)



Der Miniserver hat bei dieser Lösung relativ wenig zu tun; er muss keine Daten abholen oder Werte konvertieren - das Plugin sendet von sich aus die bereits umgewandelten Daten an den Miniserver, wann immer neue Messwerte von der Wetterstation eintreffen. Außerdem stehen die Messwerte auch beliebig anderen Diensten über diverse Schnittstellen zur Verfügung.

Der integrierte Webserver verarbeitet neben "updateweatherstation" zur Annahme eines Datensatzes im WU-Format (Weather Underground-Protokoll) auch andere http-Aufrufparameter im GET:

/CSVHDR	die Feldbezeichnungen (der Header) des letzten Datensatzes werden als CSV semikolonsepariert ausgegeben. Wird zusätzlich units=e angegeben, erfolgt die Ausgabe der Felder für die imperialen Werte.																								
/CSV	alle gemeldeten metrischen Werte des letzten Datensatzes werden als CSV semikolonsepariert ausgegeben (units=e liefert die imperialen Werte)																								
/UDP	der letzte UDP-String wird per http ausgegeben; durch Zusatz von status innerhalb der URL erfolgt zusätzlich die Ausgabe der aktuellen Stati.																								
/RAW	der von der Wetterstation gelieferte Datensatz wird unverändert per http ausgegeben; Separator kann mit separator=Z geändert werden, wobei Z ein einzelnes Zeichen ist																								
/STRING	der umgewandelte Datensatz sowie der aktuelle Status wird mit Separator ";" per http ausgegeben; per Zusatz von units=e in der URL erfolgt die Ausgabe mit den imp. Werten; Separator kann jeweils mit separator=Z geändert werden, wobei Z ein einzelnes Zeichen ist. Durch Zusatz von status innerhalb der URL erfolgt zusätzlich die Ausgabe der aktuellen Stati. Beispiel: http://ipadresse:port/STRING?units=e?separator=, gibt die imp. Werte mit Komma als Separator aus																								
/JSON	Ausgabe per http als JSON (standardmäßig metrisch; per Zusatz von units=e in der URL erfolgt die Ausgabe mit den imp. Werten). Durch Zusatz von status innerhalb der URL erfolgt zusätzlich die Ausgabe der aktuellen Stati. Beispiel: http://ipadresse:port/JSON?units=m?status gibt alle metrischen Werte inkl. der aktuellen Stati als JSON aus																								
/realtime.txt	Ausgabe per http als realtime.txt (Cumulus Export-File)																								
/clientraw.txt	Ausgabe per http als clientraw.txt (Weather Display Export-File)																								
/getvalue?key=keyname	gibt den Wert für den Schlüssel keyname aus - keyname darf ein RAW- oder umgewandelter Schlüssel sein - sinnvoll etwa für Abfragen etwa via curl oder wget Beispiel: http://ipadresse:port/getvalue?key=windspeedmph gibt den Wert für den Schlüssel windspeedmph aus																								
/ (ohne)	simple Webseite mit den aktuellen metrischen Daten in Tabellenform; durch Angabe von status auch mit nzeige der Statusmeldungen; gibt bei zusätzlicher Angabe von units=e Werte auch im US-System (inch, mph etc.) aus																								
/FOSHKplugin/state	Status des Dienstes; wenn aktiv: "**running**"																								
/FOSHKplugin/status	Anzeige der Statusmeldungen für Gewitter, Sturm, Batterie, Fehlen eines Sensors, Stations-Watchdog, … als simple Webseite																								
/FOSHKplugin/debug=enable	aktiviert den Debug-Modus für erweiterte Meldungen im Logfile																								
/FOSHKplugin/debug=disable	deaktiviert den Debug-Modus für erweiterte Meldungen im Logfile																								
/FOSHKplugin/pushover=enable	temporäres Aktivieren der Push-Mitteilungen via Pushover - Pushover muss jedoch bereits korrekt konfiguriert sein																								
/FOSHKplugin/pushover=disable	temporäres Deaktivieren der Push-Mitteilungen via Pushover																								
/FOSHKplugin/patchW4L	"Patchen" einer Weather4Loxone-Installation (lokale Grabber-Scripte kopieren und lokalen Abruf durch W4L aktivieren)																								
/FOSHKplugin/recoverW4L	Wiederherstellen der originalen Weather4Loxone-Konfiguration vor dem "Patchen"																								
/observations/current/json/units=m	Rückmeldung eines WU-kompatiblen Datensatzes mit metrischen Werten (°C, kmh, mm, hPa)																								
/observations/current/json/units=e	Rückmeldung des WU-kompatiblen Datensatzes mit imperialen Werten (°F, mph, in, inHg)																								
/w4l/current.dat	Rückmeldung einer W4L-kompatiblen current.dat: 1617139459	Tue, 30 Mar 2021 23:24:19 +0200	CET	Europe/Berlin	+0200	Hohen Neuendorf	DE	52.669481	13.266531	53	8.2	8.2	92	Westsüdwest	256	0.0	0.0	8.2	1027.09	7.0	0.00	7.5	0	0.0	0.0

Bei Nutzung der Authentifizierung über AUTH_PWD im Config-File müssen sämtliche Requests mit einem angehängtem **?auth=[PASSKEY]** ausgeführt werden. Einzig FOSHKplugin/state funktioniert auch ohne diese Authentifizierung. Sinnvoll ist dies, wenn FOSHKplugin nicht im sicheren lokalen Netzwerk sondern direkt im Internet - etwa auf einem Root-Server - arbeiten soll. Ohne diesen Sicherheitsmechanismus könnte sonst jedermann Daten einliefern oder Zustände abfragen oder ändern. Grundsätzlich rate ich jedoch vom Betrieb auf frei im Internet stehenden "unsicheren" Hosts ab.

Im POST-Modus werden Daten der Wetterstation im Ecowitt-Format angenommen, wenn in der URL das Schlüsselwort "report" enthalten ist. Da im Ecowitt-Format deutlich mehr Werte von der Wetterstation übertragen werden können (etwa die Batteriewerte der Sensoren), empfehle ich diese Betriebsart (die vom Plugin auch so bei WS-Set gesetzt wird).

Vergleich der von der Wetterstation übermittelten Werte zur ungefähr gleichen Zeit mit identischer Sensorausstattung zwischen WU- und Ecowitt-Format:

WU-Format:

ID=id&PASSWORD=key&tempf=41.0&humidity=97&dewptf=40.3&windchillf=41.0&winddir=172&windspeedmph=0.00&windgustmph=0.00&rainin=0.000&dailyrainin=0.150&weeklyrainin=0.197&monthlyrainin=1.209&yearlyrainin=1.228&solarradiation=0.00&UV=0&indoortempf=74.8&indoorhumidity=38&baromin=29.695&soilmoisture=51&soilmoisture2=49&lowbatt=0&dateutc=now&softwaretype=GW1000A_V1.5.4&action=updateraw&realtime=1&rtfreq=5

Ecowitt-Format:

PASSKEY=3DD1D3E8695D6E7B8B9E0D1121E1434&stationtype=GW1000A_V1.5.4&dateutc=2019-12-24+22:29:23&tempinf=74.7&humidityin=38&baromrelin=29.695&baromabsin=29.563&tempf=41.0&humidity=97&winddir=172&windspeedmph=0.00&windgustmph=0.00&maxdailygust=0.47&solarradiation=0.00&uv=0&rainratein=0.000&eventrain=0.150&hourlyrain=0.000&dailyrain=0.150&weeklyrain=0.197&monthlyrain=1.209&yearlyrain=1.228&totalrain=1.228&tempinf=74.7&humidityin=38&temp1f=73.9&humidity1=61&soilmoisture1=51&soilmoisture2=49&wh65batt=0&ch1=0&ch2=0&ch3=0&ch4=0&ch5=0&ch6=0&ch7=0&ch8=0&wh25batt=0&wh26batt=0&wh31ch1=0&wh31ch2=0&wh31ch3=0&wh31ch4=0&wh31ch5=0&wh31ch6=0&ch1_batt=1&ch2_batt=1&ch3_batt=1&ch4_batt=1&ch5_batt=1&ch6_batt=1&ch7_batt=1&ch8_batt=1&wh57batt=5&wh57freq=868M&wh57mode=GW1000_Pro

Im WU-Format fehlen nicht nur die Batteriewerte der Sensoren sondern auch die Temperatur- und Feuchtigkeitswerte der Innensensoren. Einige zusätzliche Sensoren (etwa Blitzsensor und Wassersensor) werden von WU überhaupt nicht unterstützt - diese fehlen in den WU-Daten also komplett. Dafür liefert das WU-Format aber den Taupunkt und Windchill von sich aus mit; bei Ecowitt müssen diese Werte via Schalter "optionale Berechnungen" durch das Plugin errechnet werden.

Installation

Grundsätzlich sollten initial die Sensoren über die App WS View angelernt und eingerichtet werden.

WS View-App aus dem jeweiligen Shop holen (siehe Links)
Wetterstation lt. Herstelleranleitung einrichten
Wenn soweit über die App alles funktioniert - Messdaten also innerhalb von WS View angezeigt werden - kann die Anbindung an Loxone erfolgen.

Im Hauptbildschirm des LoxBerry ist auf **"*Plugin-Verwaltung*"** zu klicken und der Link des loxberry-FOSHKplugin.zip unter **"*Installiere neues Plugin:*"** sowie die SecurePIN einzugeben und auf **"*Installation*"** zu klicken. Nach erfolgreicher Installation steht das FOSHKplugin unter Plugins in der Hauptübersicht zur weiteren Konfiguration und Aktivierung bereit.

Konfigurationsoptionen

Alle erforderlichen Einstellungen werden bei der Installation bereits auf sinnvolle Werte gesetzt. Auf der Einstellungs-Seite sind die oberen Eingabe-Felder für die Loxone/LoxBerry-Konfiguration auszufüllen bzw. die automatisch vorgegebenen Werte ggf. anzupassen:

Die Konfiguration der Wetterstation erfolgt im unteren Bereich. Sind die dort einzugebenden Daten unbekannt, können diese über die jeweiligen " ***Erkenne*** "-Buttons abgefragt werden.
Der zulässige Wertebereich für den Datenversand von der Wetterstation zum lokalen Server (Sende-Intervall) beträgt lt. WS View 16 bis 600 Sekunden. Über das Plugin lässt sich der Intervall aber auch auf eine Sekunde setzen (getestet am DP1500). Und tatsächlich kommen dann Messwerte im Sekundentakt an! Ich gehe aber davon aus, dass sich der Hersteller der App etwas mit diesen Limits gedacht hat und empfehle, innerhalb dieses Bereichs zu bleiben.

Nach einem weiteren **"*Speichern*"** sind die Konfigurationsdaten abgespeichert und die eigentliche Konfiguration des Plugins beendet. Nur die Wetterstation selbst muss noch von etwaigen Änderungen informiert werden. Dies erfolgt über den Button **"*WS-Set*"** . Dabei wird in der Wetterstation der Wetter-Service Customized im Ecowitt-Protokoll mit dem hier konfigurierten Sende-Intervall zur IP-Adresse des LoxBerry auf den konfigurierten HTTP-Port aktiviert. Ein ggf. mit WS View modifizierter Path wird dabei überschrieben. FOSHKplugin setzt die ursprünglichen defaults - also /data/report/ - bei Ecowitt bzw. /weatherstation/updateweatherstation.php? bei WU.

Mit dem Button **"*Restart*"** kann der systemd-Dienst des Plugins neugestartet werden. Änderungen an der Konfiguration werden erst nach einem Neustart des Dienstes aktiv.

Über **"*Vorlage*"** lässt sich die Loxone-Vorlagedatei downloaden. Darin enthalten sind sämtliche virtuellen In- und Outputs zur leichteren Integration im Miniserver.

Wichtig: Die Änderung des HTTP-Ports oder der IP-Adresse des LoxBerry sowie des Sende-Intervalls erfordert das Speichern der Settings in der Wetterstation via Button WS-Set!

Unter **"*Optionale Einstellungen*"** sind noch diverse Zusatzfunktionen konfigurierbar:

metrische Einheiten:
wenn aktiviert erfolgt die Umrechnung der in US-Einheiten von der Wetterstation gelieferten Werte für UDP-Versand und CSV-Export direkt durch das Plugin

leere Werte überspringen:
bei Aktivierung werden ggf. von der Wetterstation kommende Werte -9999 nicht per UDP an den Miniserver verschickt

nutze Loxone-Zeit:

Über diesen Schalter wird festgelegt, ob eine Umrechnung der UTC-Zeit auf Loxone-Zeit erfolgen soll. Bei Aktivierung wird ein zusätzliches Feld loxtime im Loxone-kompatiblen Zeitformat (Sekunden seit 01.01.2009) angefügt.

optionale Berechnungen:
Bei Aktivierung werden die Werte für Taupunkt, Windchill-Temperatur, Hitzeindex und gefühlte Temperatur und - sofern ein Feinstaubsensor DP200/WH41/WH43 vorhanden ist - der AQI-Wert aktuell und dessen 24h-Mittel aus den vorliegenden Messwerten errechnet und den von der Wetterstation kommenden Daten für die Export-Verarbeitung (UDP, WU, CSV, W4L, ...) hinzugefügt. Dabei werden ggf. bereits von der Wetterstation kommende Werte NICHT überschrieben. Ist die Sturmwarnung aktiviert, erfolgt zusätzlich die Berechnung des Luftdrucktrends und der Luftdruckänderung (für die letzte Stunde sowie für die letzten 3 Stunden).

optionale Elemente:
hängt einen String mit statischen Werten an die von der Wetterstation kommende Raw-Datenzeile an, ggf. vorhandene Variablennamen mit gleichen Namen werden dabei überschrieben. Sinnvoll, um ein paar Felder (wie Geolokalisierung: lat/lon/elev oder Ort: neighborhood) per UDP/WU/CSV/W4L etc. weiterzugegeben. Diese Felder durchlaufen die komplette Exportverarbeitung, tauchen somit in allen Ausgabeformaten auf. Diese Funktion kann auch dazu genutzt werden, um von der Wetterstation kommende Werte von der Weiterverarbeitung auszuschließen. Dazu muss hier einer Variablen ein leerer Wert zugewiesen werden (also: &variable3=&variable4=wert4) und "leere Werte überspringen" aktiviert sein.
Format: &variable1=wert1&variable2=wert2

Log-Dateien:
Sind für die Inbetriebnahme sowie bei Problemen sehr nützlich. Man sollte jedoch abwägen, ob das dauerhafte Mitschreiben der Logs wirklich sinnvoll ist. Bei Einsatz einer SD-Karte als Speichermedium schreibt man sich sonst irgendwann die SD-Karte kaputt.
Vorallem das Export-Log kann - wenn ein sehr kurzer Intervall eingestellt ist, sehr schnell sehr groß werden, da für jede von der Wetterstation kommende Nachricht - je nach Konfiguration - eben auch ein Eintrag für UDP, Weiterleitung (FWD) und CSV erzeugt wird.

Zum Deaktivieren eines bestimmten Log-Files ist schlicht der Name der jeweiligen Datei zu entfernen. Seit v0.08 ist es möglich, das Logging global über den Switch im Config-File Logging\LOG_ENABLE = True/False an- und abzuschalten.

Weiterleiten an:
Es ist nur ein externes Ziel für den Versand per "Customized Upload" in der Konfiguration einer Wetterstation vorhanden. Da wir dieses bereits für den LoxBerry nutzen, kann man hier eine Weiterleitung an einen zusätzlichen Dienst (etwa Awekas) einstellen. Aktuell unterstützt das Plugin 50 Weiterleitungsziele, wobei nur eines über die Weboberfläche zu konfigurieren ist. Die restlichen Ziele sind ggf. direkt über die Config-Datei einzurichten.

Bei der Angabe der URL ist zu beachten, dass vom Plugin nur die Messwerte hinzugefügt werden. Etwaige Authentifizierungen oder Update-Befehle müssen also bereits an dieser Stelle eingegeben werden.
Für einen Upload zu Weather Underground (der natürlich auch direkt über die Wetterstation möglich ist) sähe eine solche Zeile also wie folgt aus:

https://rtupdate.wunderground.com/weatherstation/updateweatherstation.php?ID=[meine ID]&PASSWORD=[mein Password]&action=updateraw&

Erfolgreich getestet habe ich hier den Versand an die Dienste Awekas, Windy und PWSWeather:

URL für Awekas:
http://ws.awekas.at/weatherstation/updateweatherstation.php?ID=[awekasid]&PASSWORD=[awekaspassword]&

URL für Windy:
https://stations.windy.com/pws/update/[windyAPIkey]?

URL für PWSWeather:
http://www.pwsweather.com/pwsupdate/pwsupdate.php?ID=[PWS-ID]&PASSWORD=[PWS-Password]&
(in Kürze offenbar
https://pwsupdate.pwsweather.com/api/v1/submitwx?ID=[PWS-ID]&PASSWORD=[PWS-Password]&)

Andere WU-kompatible Dienste sollten ebenfalls funktionieren.
Bleibt das Feld frei, erfolgt keine Weiterleitung.

Mit "***Weiterleiten Format:***" wird festgelegt, in welchem Format die weitergeleiteten Nachrichten der Wetterstation versandt werden sollen.
Für WU-kompatible Server sollte das WU-Format ausgewählt werden. Für andere Szenarien gibt es auch das UDPGET-Format, bei dem die ggf. umgewandelten metrischen Werte wie bei UDP (jedoch nicht durch Leerzeichen sondern durch html-konforme "&" separiert) verschickt werden. Darüber sollten sich virtuelle http-Eingänge realisieren lassen.
Weiter verbessert und ausgiebig getestet wurde das EW-Format. Dabei werden eingehende Nachrichten der Wetterstation in das Ecowitt-Format umgewandelt und im Ecowitt-Protokoll per HTTP-Post weiterversandt. Somit lassen sich darüber auch weitere Hosts per Ecowitt-Protokoll bedienen (Relay).
Mit Typ RAW werden die eingehenden Daten ohne Konvertierung per http-get weitergeleitet. Um den originalen RAW-String ohne jegliche Erweiterung im EW-Format per POST zu versenden, bietet sich der Typ RAWEW an. Über RAWUDP können die RAW-Daten auch per UDP verschickt werden, dabei ist als FWD_URL dann destination-ip:destination-port anzugeben. Sollen weitere Ziele die verarbeiteten (und ggf. umgerechneten) Daten per UDP erhalten, ist der Forward-Typ UDP nützlich. Auch hier erfolgt die Angabe des Ziels über die FWD_URL mit destination-ip:destination-port.
Ebenfalls können die für den Dienst luftdaten.info erforderlichen Werte eines vorhandenen Feinstaubsensors DP200/WH41/WH43 über den Typ LD gesendet werden:

URL für Luftdaten:
https://api.sensor.community/v1/push-sensor-data/

Die zur Anmeldung erforderliche Sensor-ID ist dazu im Config-File unter FWD_SID einzutragen. Als Intervall für das Senden der Feinstaubsensor-Werte sollte 150 Sekunden konfiguriert werden (FWD_INTERVAL = 150 im Config-File). Der Dienst erwartet neben dem PM2.5-Wert auch den PM10-Wert (den der Feinstaubsensor DP200/WH41/WH43 aber nicht liefern kann). Daher sendet das Plugin jeweils einen Dummy-Wert von 1 für PM10 mit.

Übersicht über die verschiedenen Forward-Möglichkeiten:

FWD_TYPE	input-Format	out-Transport	out-Format
WU	WU, EW, AMB	GET	Weather Underground (WU–>WU bzw. EW–>WU)
RAW	WU, EW, AMB	GET	wie input (WU–>WU bzw. EW–>EW)

UDPGET	WU, EW, AMB	GET	wie Ausgabe zu Loxone mit Header und ggf. Umrechnung jedoch URL-kompatibel mit "&" statt Leerzeichen
WC	WU, EW, AMB	GET	Weathercloud
MT	WU, EW, AMB	GET	Meteotemplate (API)
AMB	WU, EW, AMB	GET	Ambient Weather
AWEKAS	WU, EW, AMB	GET	Awekas (API)
WETTERCOM	WU, EW, AMB	GET	wetter.com/Wetterarchiv (API)
EW	WU, EW, AMB	POST	erweitertes Ecowitt (WU–>EW bzw. EW–>EW)
RAWEW	WU, EW, AMB	POST	unverändertes Ecowitt (EW–>EW und WU–>EW)
LD	WU, EW, AMB	POST	Luftdaten.info-Format (nur PM2.5, PM10, Temp, Humidity, rel. Pressure, abs. Presssure)
CSV	WU, EW, AMB	POST	wie Ausgabe zu Loxone mit ggf. Umrechnung jedoch separiert mit Semikolon statt mit Leerzeichen und ohne Header
RAWCSV	WU, EW, AMB	POST	wie input (WU–>WU bzw. EW–>EW) jedoch separiert mit Semikolon statt mit Leerzeichen
WEATHER365	WU, EW, AMB	POST	weather365.net API
UDP	WU, EW, AMB	UDP	wie Ausgabe zu Loxone mit Header und ggf. Umrechnung per UDP (Ziel-IP:Port muss als FWD_URL deklariert werden)
RAWUDP	WU, EW, AMB	UDP	wie input-Format jedoch Versand per UDP (EW→EW und WU–>WU)
REALTIMETXT	WU, EW, AMB	diverse	sendet eine realtime.txt (Cumulus Export-Datei) via http(s)/POST oder per ftp(s) zu einem entfernten Ziel oder speichert die Datei im Dateisystem
CLIENTRAWTXT	WU, EW, AMB	diverse	sendet eine clientraw.txt (Weather Display Export-Datei) via http(s)/POST oder per ftp(s) zu einem entfernten Ziel oder speichert die Datei im Dateisystem
CSVFILE	WU, EW, AMB	diverse	sendet eine Datei FOSHKplugin.csv mit dem aktuellen Datensatz via http(s)/POST oder per ftp(s) zu einem entfernten Ziel oder speichert die Datei im Dateisystem
TXTFILE	WU, EW, AMB	diverse	sendet eine Datei FOSHKplugin.txt mit dem aktuellen Datensatz via http(s)/POST oder per ftp(s) zu einem entfernten Ziel oder speichert die Datei im Dateisystem
RAWTEXT	WU, EW, AMB	diverse	sendet eine Datei rawtext.txt mit dem aktuellen RAW-Datensatz via http(s)/POST oder per ftp(s) zu einem entfernten Ziel oder speichert die Datei im Dateisystem
MQTTMET	WU, EW, AMB	MQTT	sendet eingehende Daten der Wetterstation per MQTT an einen MQTT-Broker im metrischen Format weiter
MQTTIMP	WU, EW, AMB	MQTT	sendet eingehende Daten der Wetterstation per MQTT an einen MQTT-Broker im imperialen Maßsystem weiter
WSWIN	WU, EW, AMB	File	speichert eine WSWin-kompatible wswin.csv im Dateisystem, die per Dateiüberwachung automatisiert von WSWin eingelesen werden kann
INFLUXMET	WU, EW, AMB	InfluxDB	speichert den aktuellen Datensatz mit metrischen Werten in einer InfluxDB-Datenbank
INFLUXIMP	WU, EW, AMB	InfluxDB	speichert den aktuellen Datensatz mit imperialen Werten in einer InfluxDB-Datenbank

Daten der unter **"Felder ignorieren:"** gepflegten Ignorierliste werden beim betreffenden Forward

nicht versandt.

Mit "***Weiterleiten Intervall***" kann ein von der Wetterstation unabhängiger Intervall (in Sekunden) konfiguriert werden. Bleibt dieses Feld frei, erfolgt der Versand im Sende-Intervall der Wetterstation.

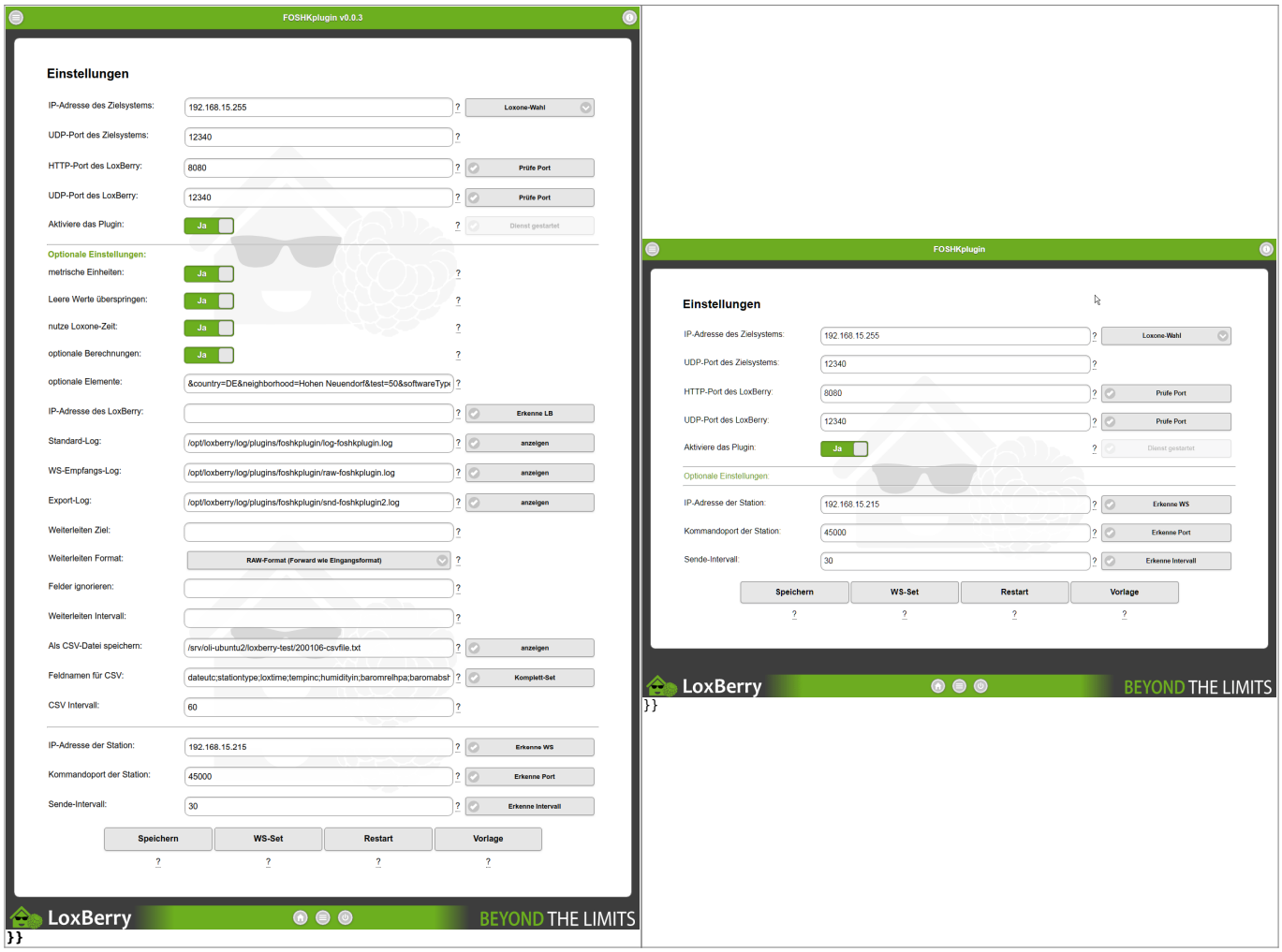
Als CSV speichern:
Die Messergebnisse können zusätzlich als Kommaseparierte Datei (CSV) abgespeichert werden. Der Ablageort sowie der Dateiname wird hier angegeben. Auch hier gilt das bereits für Log-Dateien erwähnte Problem mit dem Schreiben auf SD-Karten. Hier sollte also ggf. ein besser geeignetes Medium (etwa NFS) gewählt werden.

Feldnamen für CSV:
Hier werden alle im CSV gewünschten Felder - mit einem Separator (Semikolon, Komma oder Leerzeichen) getrennt - aufgeführt.
Nicht alle Felder eines Datensatzes lohnen für eine Speicherung im CSV. So ändern sich die Inhalte der Felder SID, PASSKEY, freq oder model nur sehr selten.
Durch Weglassen dieser Feldnamen werden diese Felder somit von der Speicherung ausgeschlossen.
Die Reihenfolge der Spalten im CSV-File ergibt sich aus der Reihenfolge der hier angegebenen Felder.

CSV Intervall:
Hier kann ein eigener Zeitabstand für das Abspeichern eines Datensatzes im CSV definiert werden. Bleibt das Feld frei wird der Sende-Intervall der Wetterstation genutzt.

Der Intervall für CSV- und Weiterleitungs-Funktion kann nicht kleiner als der eingestellte Sende-Intervall der Wetterstation sein, da nur bei Eingang eines Datensatzes von der Wetterstation Daten zum Weiterverarbeiten vorliegen.

Übersicht des GUI	animierte Übersicht des GUI (mit Rollover-Hilfe)



Für einige (seltene) Einstellungen gibt es derzeit kein Web-Interface. Daher müssen diese Einstellungen direkt in der Config-Datei vollzogen werden. Diese Konfigurationsoptionen sind für den normalen Betrieb nicht nötig, können aber erweiterte Einsatzszenarien unterstützen.

Bei einer Standard-Installation befindet sich die Config-Datei foshkplugin.conf in /opt/loxberry/config/plugins/foshkplugin/. Diese Datei kann mit einem beliebigen Editor geändert werden. Sollten im Config-File diese Einträge nicht enthalten sein (was bei schon länger aktiven FOSHKplugin-Installationen möglich ist), können diese händisch nachgetragen werden. Oder man kopiert die entsprechenden Blöcke aus der default-Config im ZIP-File, über das das Plugin-Update installiert wurde. Wichtig: Die geänderten Einstellungen werden erst nach einem Neustart des Plugins aktiv!

Über das LoxBerry-Web-Interface lässt sich ein Forward für den Versand an einen weiteren externen Server konfigurieren. Das ist hilfreich, wenn man z.B. seine Wetterdaten auch einem anderen Wetterdienst (etwa Awekas oder Windy) zur Verfügung stellen möchte.
Man kann mit diesem Plugin jedoch insgesamt **50 Weiterleitungsziele** definieren. Dabei wird neben dem WU-Format (also eine Auswahl der Werte und ausschließlich im US-Maßsystem) für WU-kompatible Systeme auch ein Pseudo-UDP-Format mit metrischen Messwerten sowie der Versand an Systeme im Ecowitt-Protokoll unterstützt.
Im Config-File können dazu folgende Einträge getätigt werden:

[Forward-**n**] FWD_URL = FWD_TYPE = "" FWD_IGNORE = "" FWD_INTERVAL = FWD_SID =	# neben Forward (per Webinterface konfigurierbar) werden auch Forward-1 bis Forward-9 unterstützt # die URL des Zielsystems, etwaige Authentifizierungen sind bereits hier mit anzugeben # hier wird zwischen WU, UDPGET, EW, LD, RAW, UDP und RAWUDP unterschieden; WU-format (default) # eine Komma-separierte Liste der Felder, die nicht verschickt werden sollen # der Intervall in Sekunden, in dem die Nachrichten verschickt werden sollen # Sensor-ID für luftdaten.info

Daten der Ignorierliste FWD_IGNORE werden beim betreffenden Forward nicht versandt.

Die Ausgabezeile sähe dann z.B. bei FWD_TYPE=UDPGET so aus:

Forward im WU-Format zu Awekas:

Die standardmäßig aktivierte Prüfung des Eingangs von neuen Nachrichten von der Wetterstation sowie deren Intervall können im Config-File deaktiviert werden:

[Warning] WSDOG_WARNING = True WSDOG_INTERVAL = 3	# aktiviert (Standard) oder deaktiviert die Prüfung eingehender Daten von der Wetterstation # Anzahl der Sende-Intervalle ohne Daten von der Wetterstation, bevor eine Warnung per Log/UDP erfolgt

Eine aktivierte Prüfung kann hilfreich sein um den Ausfall der Wetterstation frühzeitig zu bemerken.

Die Wetterstation sammelt die Daten aller Sensoren zusammen und verschickt diese als ein Datenpaket an FOSHKplugin. Wenn ein Sensor ausfällt, sendet die Wetterstation ohne weitere Hinweise eben alle zur Verfügung stehenden Daten.
Um nun den Ausfall eines spezifischen Sensors feststellen zu können gibt es die Liste SENSOR_MANDATORY, in die normalerweise von der Wetterstation kommende Felder eingetragen werden können. FOSHKplugin prüft das eingehende Datenpaket auf Vorhandensein der in dieser Liste enthaltenen Sensordaten und meldet bei aktivierter Prüfung SENSOR_WARNING=True unter [Warning] das Fehlen per Log-Eintrag und per UDP.

SENSOR_WARNING = False SENSOR_MANDATORY = "wh65batt"	# aktiviert/deaktivert die Sensor-Warn-Funktion (Standard: deaktiviert) # eine kommaseparierte Liste von Feldern, die erwartet werden; bei Fehlen eines dieser Felder erfolgt eine Warnung

Somit lässt sich zeitnah der Ausfall eines Sensors feststellen. Zu beachten ist jedoch, dass fehlende Werte bei verschiedenen Sensoren von der Wetterstation zwischengespeichert und - trotz Ausfall des Sensors - noch weitere 10 Minuten weiter geschickt werden. Zumindest beim Wassersensor WH55, dem Bodenfeuchtesensor DP100/WH51 sowie dem Feinstaubsensor WH41/DP200 konnte ich diese Arbeitsweise feststellen. Weitere Sensoren habe ich noch nicht auf dieses Verhalten hin getestet. Man erhält also u.U. erst mit 10 Minuten Verspätung eine Info über den Ausfall eines Sensors.

Die Prüfung der verschiedenen Sensoren auf Batteriekapazität kann innerhalb von Loxone recht schnell kompliziert werden, da die übergebenen Werte von Fine Offset uneinheitlich geregelt sind. So gibt es Sensoren, die bei ausreichender Batteriekapazität im jeweiligen batt-Feld eine 0 mitsenden

und andere, die den Batterie-Level (5..1) übertragen. Auch die verfügbare Spannung wird von manchen Sensoren übertragen.
Ab v0.06 gibt es die per default aktivierte Batteriewarnung, die bei Unterschreitung eines intern festgelegten Levels eine Warnung per UDP und im Log-File sendet. Diese Prüfung kann mit BATTERY_WARNING=False unter [Warning] im Config-File abgeschaltet werden.

BATTERY_WARNING = True	# aktiviert/deaktivert die Batterie-Warn-Funktion (Standard: aktiviert)

Innerhalb von Loxone kann dann bei Erkennung von FOSHK-batterywarning=1 festgestellt werden, dass ein Sensor demnächst keine ausreichende Batteriekapazität hat. Für die Feststellung um welchen Sensor es sich handelt ist entweder massiv mit der Befehlserkennung von Loxone zu basteln (diese zusätzliche Info über den betreffenden Sensor wird als Text übertragen) oder man schaut in der WS View-App oder im Log-File von FOSHKplugin nach.

Auch die per default aktive Sturmwarnung kann unter [Warning] mit False abgeschaltet oder ein anderer Schwellwert (default = 1.75 oder 3.75 in 3 Stunden) für den Anstieg/Abfall des Luftdrucks innerhalb einer Stunde und Zeit, nach der eine Entwarnung erfolgt, eingestellt werden:

STORM_WARNING = True STORM_WARNDIFF = 1.75 STORM_WARNDIFF3H = 3.75 STORM_EXPIRE = 60	# aktiviert/deaktiviert die integrierte Sturmwarnung von FOSHKplugin (Standard: aktiv) # Wertänderung in hPa pro Stunde ab dem das Plugin eine Warnung per Log/UDP ausgibt # Wertänderung in hPa für 3 Stunden, die eine Warnung auslöst # Sturmwarnung bleibt für 60 Minuten nach letzter Grenzwertunter-/überschreitung aktiv

Ist ein Blitzsensor WH57/DP60 vorhanden, erfolgt bei Erfüllung verschiedener Kriterien eine Gewitterwarnung. Diese Warnung kann im Config-File deaktiviert werden. Die Kriterien zur Auslösung dieser Warnung sind im Config-File konfigurierbar:

TSTORM_WARNING = True TSTORM_WARNCOUNT = 1 TSTORM_WARNDIST = 20 TSTORM_EXPIRE = 30	# aktiviert/deaktiviert die integrierte Gewitterwarnung von FOSHKplugin (Standard: aktiv) # Anzahl der detektierten Blitze ab der das Plugin eine Warnung per Log/UDP ausgibt # Warnung erfolgt nur bei Detektion der Blitze innerhalb des hier eingestellten Bereiches (km) # Zeit in Minuten nach letztem Blitz, bevor Warnung aufgehoben wird

Wenn ein Versand von UDP-Nachrichten an das Zielsystem grundsätzlich unterbunden werden soll (etwa, weil nur die Forward-/Relay-Funktion genutzt wird), lässt sich dies mit dem Schalter UDP_DISABLE realisieren. Eine Liste von Feldern, die nicht per UDP verschickt werden sollen, lässt sich unter UDP_IGNORE pflegen.

[Config] UDP_ENABLE=False UDP_IGNORE= AUTH_PWD=	# deaktivert den Versand von UDP-Nachrichten (nicht aber RAWUDP als Forward) # kommaseparierte Auswahl von Feldern, die nicht per UDP verschickt werden sollen # optionales Password für ein- und ausgehende http-Anfragen. Wenn gesetzt werden ausschließlich Anfragen akzeptiert, die das Password in der URL enthalten. Im Ecowitt-Mode ist dafür der PASSKEY geeignet.

Die standardmäßig aktivierte Prüfung auf verfügbare Firmware-Updates für die Wetterstation kann im Bereich [Update] des Conf-Files angepasst werden:

```
[Update]
UPD_CHECK=True          # Update-Prüfung aktivieren/deaktiveren (default: aktiv)
UPD_INTERVAL=86400      # Intervall der Updateprüfung in Sekunden (default 86400 = 24h)
UPD_URL=                # Link zum Firmware-Update-Info-File (default: intern vorgegeben)
```

Fake-Mode

Die Werte für Außentemperatur und Luftfeuchte kommen normalerweise entweder von einem Kombisensor (WH65, WS80) oder dem dezidierten Außensensor WH32.
Steht aber weder ein Kombisensor noch ein WH32 zur Verfügung, können Werte eines beliebigen Innensensors DP50/WH31 (der sinnvollerweise dann natürlich außen mit entsprechenden Wetterschutz installiert werden sollte) als Werte des Außensensors ausgegeben werden.
Im Config-file ist dazu festzulegen, welcher Key für den jeweiligen Wert genutzt werden soll:

```
[Export]
OUT_TEMP=temp1f     # ersetzt den keyname temp1f mit tempf
OUT_HUM=humidity1   # ersetzt den keyname humidity1 mit humidity
```

Innerhalb von FOSHKplugin wird bei Eingang einer Ecowitt-Zeile von der Wetterstation dann einfach der Teilstring "&temp1f=" durch "&tempf=" sowie "&humidity1=" durch "&humidity=" ersetzt. Die Werte selbst bleiben bestehen.
Diese Einstellung ist global und wirkt sich somit auf alle Exports/Forwards/Ausgaben (außer bei RAW und RAWEW) von FOSHKplugin aus.
Die Wetterstation selbst weiss davon natürlich nichts - somit kennen die dort konfigurierten Dienste (Ecowitt, WU, WOW, etc.) weiterhin keine Außenwerte.
Möchte man die Werte des Innensensors als Außensensor-Werte auch für diese Dienste ausgeben, müssen die Dienste innerhalb der Wetterstation deaktiviert und stattdessen von FOSHKplugin durchgeführt werden. Dazu sind im Config-File entsprechende Forwards zu definieren (die eckigen Klammern sind dabei jeweils **NICHT** mit anzugeben):

```
WU:
[Forward-11]
FWD_INTERVAL = 300
FWD_URL =
https://rtupdate.wunderground.com/weatherstation/updateweatherstation.php?ID=[WU-ID]&PASSWORD=[WU-Password]&action=updateraw&
FWD_TYPE = WU
```

```
EW:
[Forward-12]
FWD_INTERVAL = 300
FWD_URL = http://cdnrtpdate.ecowitt.net/data/report/
FWD_TYPE = EW
```

```
WOW:
[Forward-13]
FWD_INTERVAL = 300
FWD_URL = http://wow.metoffice.gov.uk/automaticreading?siteid=[siteid]&siteAuthenticationKey=[siteAuthenticationKey]&
FWD_TYPE = WU
```

```
Weathercloud:
[Forward-14]
FWD_INTERVAL = 300
FWD_URL = http://api.weathercloud.net/v01/set?wid=[weathercloudid]&key=[key]&
FWD_TYPE = WC
```

```
Meteotemplate:
[Forward-15]
FWD_INTERVAL = 60          # should be shorter than 5 minutes
FWD_URL = http://192.168.15.100/template/api.php?PASS=[meteotemplatepwd]&
FWD_TYPE = MT
```

```
#Awekas-API:
[Forward-16]
FWD_CMT = forward im Awekas-API-Format
FWD_TYPE = AWEKAS
FWD_URL = http://data.awekas.at/eingabe_pruefung.php?
FWD_SID = Awekas-ID
FWD_PWD = Awekas-password
FWD_INTERVAL = 60
FWD_ENABLE = True
```

```
#wetter.com-API:
[Forward-17]
FWD_CMT = wetter.com via API
FWD_TYPE = WETTERCOM
FWD_URL = http://interface.wetterarchiv.de/weather
FWD_SID = station-ID
FWD_PWD = station-password
FWD_INTERVAL = 300
FWD_ENABLE = True
```

```
#weather365-API:
[Forward-18]
FWD_CMT = weather365 via API
FWD_TYPE = WEATHER365
FWD_URL = https://channel1.weather365.net/stations/index.php
FWD_SID = station-ID
FWD_INTERVAL = 300
FWD_ENABLE = True
```

```
#realtime.txt:
[Forward-19]
FWD_CMT = export realtime.txt via ftp
FWD_TYPE = REALTIMETXT
FWD_URL = ftp://anydomain.de/httpdocs/mydomain.phantasoft.de/upload
FWD_SID = ftp-username
FWD_PWD = ftp-password
FWD_INTERVAL = 30
FWD_ENABLE = True
```

```
#clientraw.txt:
[Forward-20]
FWD_CMT = export clientraw.txt to filesystem
FWD_TYPE = CLIENTRAWTXT
FWD_URL = /some/path/anyhwere/in/filesystem/
FWD_INTERVAL = 30
FWD_ENABLE = True
```

```
#MQTT:
[Forward-21]
FWD_CMT = export via MQTT (metric)
FWD_TYPE = MQTTMET
FWD_URL = 192.168.15.236:1883@hierarchy%prefix
FWD_INTERVAL = 30
FWD_ENABLE = True
FWD_SID = username
FWD_PWD = password
```

Änderungen am Config-File wie auch Änderungen an der Konfiguration über das Web-Interface erfordern den Neustart des Plugins über den *Restart*-Button!

Einrichtung in der Loxone Config Software

Das Plugin generiert auf Wunsch durch Klick auf den Button ***Vorlage*** eine Vorlage-Datei mit einer Vielzahl von virtuellen In- und Outputs, die in Loxone importiert werden kann. Zu beachten ist, dass der IE11 leider Probleme mit dem Download der Vorlagen-Datei macht. Hier ist also ggf. ein anderer Browser zu nutzen. Erfolgreich getestet habe ich das hier mit Edge, FireFox und Chrome.

Auf Grundlage der UDP-Befehlserkennung können aber auch weitere Meldungen von der Wetterstation oder Befehle für die Wetterstation implementiert werden.

Aktuell werden innerhalb der Vorlagendatei folgende Werte als virtuelle (analoge) UDP-Inputs angeboten:

für das Gateway DP1500/GW1000:
FOSHK-humidityin
FOSHK-tempinc

Bodenfeuchtesensoren DP100/WH51:
FOSHK-soilbatt1..8 (Batteriewert in Volt)
FOSHK-soilmoisture1..8

Für die Innen-Temperatur/Feuchtigkeitsmesser DP50/WH31:
FOSHK-batt1..8 (Batterie-Status; 1 = Alarm, 0 = ok)
FOSHK-humidity1..8
FOSHK-temp1..8c

Für die Feinstaubsensoren DP200/WH41/WH43:
FOSHK-pm25_avg_24h_ch1..4
FOSHK-pm25_ch1..4
FOSHK-pm25batt1..4 (Batterie-Status; 5 = max)
FOSHK-pm25_AQI_ch1..4
FOSHK-pm25_AQI_avg_24h_ch1..4
FOSHK-pm25_AQIlvl_ch1..4 (Level 1..6)
FOSHK-pm25_AQIlvl_avg_24h_ch1..4 (Level 1..6)

Status/Betriebsanzeige/Tracker:
FOSHK-running (1 = gestartet, 0 = gestoppt)
FOSHK-loxtime (Loxone-Zeit)
FOSHK-wswarning (1=Wetterstation meldet sich nicht)
FOSHK-sensorwarning (1=Sensor fehlt)
FOSHK-batterywarning (1=Batteriewarnung)
FOSHK-stormwarning (1=Sturmwarnung)
FOSHK-tswarning (1=Gewitterwarnung)
FOSHK-updatewarning (1=Update für Wetterstation verfügbar)
FOSHK-co2warning (1=CO2-Wert ist über Limit)

Für den Blitzsensor WH57/DP60:
FOSHK-lightning (Entfernung letzter Blitz in km)
FOSHK-lightning_time (Zeit letzter Blitz Unixtime)
FOSHK-lightning_loxtime (Zeit letzter Blitz)
FOSHK-lightning_num (Anzahl Blitze)
FOSHK-wh57batt (Batterie-Status; 5 = max)

Für den Boden/Wasser-Temperatursensor WN34:
FOSHK-tf_chNc (Temperatur in °C wobei N=1..8)
FOSHK-tf_battN (Batteriewert; N=1..8)

Für den Blattfeuchtesensor WN35:
FOSHK-leafwetness_chN (Blattfeuchte-Level wobei N=1..8)
FOSHK-leafbattN (Batteriewert; N=1..8)

Für den Wassersensor WH55:
FOSHK-leak_ch1..4 (1=Alarm, 0=ok)
FOSHK-leakbatt1..4 (Batterie-Status; 5 = max)

Für den Luftqualitätssensor WH45:
FOSHK-co2lvl
FOSHK-pm25_AQI_co2
FOSHK-pm25_AQIlvl_co2
FOSHK-pm25_AQI_24h_co2
FOSHK-pm25_AQIlvl_24h_co2
FOSHK-pm10_AQI_co2
FOSHK-pm10_AQIlvl_co2
FOSHK-pm10_AQI_24h_co2
FOSHK-pm10_AQIlvl_24h_co2
FOSHK-co2_batt

Für die Außenmessstationen WH3000SE All-In-One oder
HP1000SE All-In-One (WH65):
FOSHK-baromabshpa
FOSHK-baromhpa
FOSHK-baromrelhpa
FOSHK-dailyrainmm
FOSHK-dewptc
FOSHK-eventrainmm
FOSHK-feelslikec
FOSHK-heatindexc
FOSHK-hourlyrainmm
FOSHK-monthlyrainmm
FOSHK-humidity
FOSHK-loxtime
FOSHK-maxdailygust
FOSHK-rainratemm
FOSHK-solarradiation
FOSHK-tempc
FOSHK-totalrainmm
FOSHK-uv
FOSHK-weeklyrainmm
FOSHK-wh65batt (ist 1 bei Batteriewarnung, sonst 0)
FOSHK-windchillc
FOSHK-winddir
FOSHK-windgustkmh
FOSHK-windspeedkmh
FOSHK-yearlyrainmm
FOSHK-ptrend1 (Luftdruck-Trend 1h: 2=stark steigend, 1=steigend, -1 fallend, -2 stark fallend)
FOSHK-pchange1 (Luftdruckänderung 1h in hPa)
FOSHK-ptrend3 (Luftdruck-Trend 3h: 2=stark steigend, 1=steigend, -1 fallend, -2 stark fallend)
FOSHK-pchange3 (Luftdruckänderung 3h in hPa)
FOSHK-wproglvl (Wetterprognose Level)[2]
FOSHK-wnowlvl (Level Wetter aktuell)[2]

Zusätzliche Eingänge für die W4L-Integration siehe weiter unten!

Die Namen der Keys im UDP-Datagram entsprechen obigen Namen ohne "FOSHK-".

Level Wetterprognose und entsprechende Strings:

	wprogtxt					
wproglvl	DE	EN	NL	FR	ES	SK
0	Sturm mit Hagel	storm with hail	Storm met hagel	Tempête de grêle	Tormenta con granizo	Búrka s krupobitím
1	Regen/Unwetter	rain/storm	Regen/storm	Pluie / tempête	Tormenta de lluvia	Dáž?/búrka
2	regnerisch	rainy	regenachtig	pluvieux	lluvioso	daždivý
3	baldiger Regen	soon rain	binnenkort regen	bientôt la pluie	pronto lloverá	skoro dáž?
4	gleichbleibend	constant	constante	constant	constante	konštantný
5	lange schön	nice for a long time	lang mooi	longtemps belle	continuo hermosa	dlho krásne
6	schön & labil	nice & unstable	mooi en onstabiel	beau et instable	hermosa e inestable	krásne a nestabilné
7	Sturmwarnung	storm warning	Storm waarschuwing	Avertissement de tempête	Aviso de tormenta	Varovanie pred búrkou

Level Wetter aktuell und entsprechende Strings:

	wnowtxt					
wnowlvl	DE	EN	NL	FR	ES	SK
0	stürmisch, Regen	stormy, rainy	stormachtig, regen	orageux, pluie	tormentoso, lluvia	búrky, dáž?
1	regnerisch	rainy	regenachtig	pluvieux	lluvioso	daždivý
2	wechselhaft	unstable	veranderlijk	changeable	cambiable	premenlivý
3	sonnig	sunny	zonnig	ensoleillé	soleado	slne?no
4	trocken, Gewitter	dry, thunderstorm	droog, onweer	sec, orage	seco, tormenta	suchá, búrka

Die Datenpunkte für Status/Betriebsanzeige/Tracker sind eventbasiert, sie kommen also nur bei einer Zustandsänderung. Diese Meldungen sind per Default aktiviert. Einzig SENSOR_WARNING muss manuell im Config-File aktiviert (True) und um die Liste der zu überwachenden Werte (SENSOR_MANDATORY) erweitert werden. Es empfiehlt sich, auch diese vrtuellen Eingänge innerhalb Loxone als Analogeingänge zu definieren. Dann bleibt der jeweils durch das Plugin gesetzte Status für Loxone aktiv.

Aktuell gibt es folgende digitale UDP-Ausgangsverbinder:

FOSHK-Reboot	ermöglicht den Neustart des DP1500-Gateways von Loxone aus
FOSHK-Shutdown	beendet das Plugin; wenn als systemd-Dienst gestartet, wird es erneut gestartet
FOSHK-getStatus	fordert die aktuellen Statuswerte vom Plugin an (running, wswarning, sensorwarning, ...)
FOSHK-debugOn	aktiviert den Debug-Modus für erweitertes Logging
FOSHK-debugOff	deaktiviert den Debug-Modus für erweitertes Logging
FOSHK-leakWarnOn	aktiviert die Meldungen des Leckage-Sensors
FOSHK-leakWarnOff	deaktiviert die Meldungen des Leckage-Sensors
FOSHK-co2WarnOn	aktiviert die Meldungen des CO2-Sensors
FOSHK-co2WarnOff	deaktiviert die Meldungen des CO2-Sensors

Die Bedeutung der Ein- und Ausgänge sollte sich größtenteils aus den Namen ergeben. Diese basieren auf den ursprünglichen Feldnamen im Ecowitt-Format, bei dem ggf. die jeweils enthaltene Einheit geändert wurde. Zur Einbindung der Vorlagendatei in die Loxone-Konfiguration finden sich im LoxWiki sicher ein paar hilfreiche Hinweise.

Hinweis:
Es empfiehlt sich, ungenutzte Ein- und Ausgänge aus der Loxone-Konfiguration zu löschen, da der Speicherplatz des Loxone Miniservers (zumindest v1) recht knapp bemessen ist.

Interaktion mit Weather4Loxone

Da es aktuell noch keine offizielle API-Schnittstelle zum W4L-Plugin gibt, muss vorerst auf den Loxone-Grabber im W4L-Plugin zurückgegriffen werden. Dabei werden die Daten der Wetterstation zu FOSHKplugin und von dort zum Loxone-Miniserver übertragen. Vom Miniserver kann sich das W4L-Plugin die Werte abholen, um sie per Cloud Weather Emulator / Webseite innerhalb Loxone ansprechend zu visualisieren. Entsprechende Namen für die erforderlichen virtuellen Eingänge sind in der Loxone-Vorlage bereits enthalten:

w4l_cur_hu w4l_cur_pr w4l_cur_prec_1hr w4l_cur_prec_today w4l_cur_sr	w4l_cur_tt w4l_cur_uvi w4l_cur_w_dir w4l_cur_w_gu w4l_cur_w_sp	w4l_cur_w_ch w4l_cur_tt_fl w4l_cur_dp w4l_cur_hi

Durch "Patchen" der W4L-Installation kann eine nahtlose Integration von FOSHKplugin in W4L ermöglicht werden.
Dabei entfällt das Hin- und Hergeschiebe der Messwerte; W4L fragt FOSHKplugin bzgl. der lokalen Wetterdaten ohne Umwege direkt ab.
Bei jedem Abruf der Wetterdaten via W4L werden die gerade aktuellen Daten der lokalen Wetterstation von W4L lokal zu Loxone übertragen. Dies erfolgt automatisch und ohne Konfigurationsänderung, wenn bisher W4L bereits im Einsatz war, da die bereits vorhandenen Variablen nur mit den lokalen Werten der Wetterstation überschrieben werden.
Der Loxone-Grabber innerhalb von W4L (und das Anlegen etwaiger Transfervariablen w4l_irgendwas) ist dann nur noch für spezielle Fälle nötig.

Bei aktiviertem FOSHKplugin reicht ein einmaliger Aufruf der URL

http://loxberry-ip:port/FOSHKplugin/patchW4L

um folgende Schritte auszuführen:

- trägt Grabber-Daten in weather4lox.cfg ein (SERVER\LOCALGRABBER=1, SERVER\LOCALWUGRABBER=0 sowie LOCAL\URL=url und WULOCAL\URL=url)
- erzeugt ein Backup von vorhandener w4l/bin/fetch.pl –> w4l/bin/fetch.pl.foshkbackup
- erzeugt in vorhandener fetch.pl von W4L zwei neue Blöcke zum Aufruf der Scripte grabber_local.pl und grabber_wu-local.pl
- kopiert aus bin\ die Scripte grabber_local.pl und grabber_wu-local.pl nach w4l/bin/

Auf Konsolenebene kann dies aber auch durch sudo -u loxberry foshkplugin.py -patchW4L bewerkstelligt werden.

Fortan erscheinen die lokalen Wetterdaten sowohl im Loxone-Emulator als auch auf der durch W4L erzeugten Wetterseite. Ebenso werden alle W4L-Variablen mit Werten der lokalen Wetterstation ersetzt.
Zu beachten ist, dass Loxone nur einmal pro Stunde seine Wetterdaten abruft, was sich von außen leider nicht ändern lässt.
Die von W4L generierte Wetterseite ist jedoch - je nach konfigurierten Intervall - aktuell(er).

Weiterhin gilt, dass W4L die lokalen Daten nur in dem Intervall abruft und somit aktualisiert, in dem es die Wetterdaten von Darksky oder Weatherbit abholt. Dieser Intervall darf seitens der Betreiber dieser Wetterdienste jedoch nicht beliebig kurz sein. Wenn aktuellere Daten in Loxone notwendig sind, sollten daher die übertragenen UDP-Werte von FOSHKplugin genutzt werden.

Ich habe hier einen W4L-Intervall des Abrufs der Wetterdaten von DarkSky durch W4L von 5 Minuten konfiguriert - somit werden alle 5 Minuten die lokalen Wetterdaten dem Loxone-Miniserver zur Verfügung gestellt und ich bleibe im Limit für die erlaubten Abfragen bei DarkSky.

Nach einem Update von Weather4Loxone muss dieses Patchen erneut durchgeführt werden da die "gepatchte" Version von fetch.pl mit einer neuen Version des Scriptes vom W4L-Plugin überschrieben wird!

Deinstallation des "Eingriffs" mit:
sudo -u loxberry foshkplugin.py -recoverW4L oder http://loxberry-ip:port/FOSHKplugin/recoverW4L

- trägt Grabber-Daten in weather4lox.cfg aus (SERVER\LOCALGRABBER=1, SERVER\LOCALWUGRABBER=0 sowie LOCAL\URL=url und WULOCAL\URL=url)
- löscht grabber_local.pl und grabber_wu-local.pl in w4l/bin/
- benennt die Backup-Datei w4l/bin/fetch.pl.foshkbackup in w4l/bin/fetch.pl um

Achtung!
Der Plugin-Deinstallationsprozess der aktuellen LoxBerry-Version löscht zwar die Dateien w4l/bin/grabber_local.pl und w4l/bin/grabber_wu-local.pl und bennennt die w4l/bin/fetch.pl.foshkbackup wieder nach w4l/bin/fetch.pl um, trägt jedoch im weather4lox.cfg die erzeugten Konfigurationdsdaten nicht wieder aus! Das ist zwar unkritisch - widerspricht aber jeglicher Perfektion.
Daher bitte vor der eigentlichen Deinstallation des Plugins VORHER ein recoverW4l über bash oder den Aufruf des dafür vorbereiteten Links starten.
Mit einer zukünftigen Version von LoxBerry sollte es möglich werden, dass die interne Deinstallationsroutine des Plugins vor der eigentlichen Deinstallation des Plugins ablaufen kann. Das Plugin ist bereits darauf vorbereitet.

Betrieb mehrerer Wetterstationen - Betrieb mehrerer paralleler FOSHKplugin-Installationen

Grundsätzlich ist FOSHKplugin dafür gedacht, **EINE** Wetterstation zu unterstützen.
Es gab aber inzwischen schon 2 unabhängige Anfragen, ob man nicht mehrere parallele Installationen von FOSHKplugin auf einem LoxBerry installieren könnte, um damit die Daten mehrerer Wetterstationen verarbeiten zu können. Interessant ist dies vorallem, wenn die maximale Anzahl der durch die Station unterstützten Sensoren eines Typs (etwa Bodenfeuchtesensoren WH51) bereits erreicht ist, man aber weitere Sensoren dieses Typs benötigt und daher eine zweite Station kauft.

Mit FOSHKplugin ab v0.09 ist diese Möglichkeit gegeben.

Anleitung:
Um FOSHKplugin unter eigenem Namen und mit separater Konfiguration parallel in einer weiteren Instanz auf dem LoxBerry zu installieren, sind folgende 4 Schritte nötig:

1. ZIP-Datei des Plugins downloaden
2. plugin.cfg aus der ZIP-Datei extrahieren
3. plugin.cfg anpassen
    1. Hierzu mit einem Editor folgende Zeilen ändern - TITLE ist dabei der Name, unter dem das "neue" Plugin dann im LoxBerry erscheint und NAME und FOLDER bezeichnen den internen Namen und das Installationsverzeichnis.

Ich empfehle die Verwendung des Namens der sendenden Station als Suffix zu den bereits eingetragenen Werten - jedoch müssen diese Bezeichnungen LoxBerry-weit einmalig sein:
NAME=foshkplugin –> NAME=foshkplugin-gw1100
FOLDER=foshkplugin –> FOLDER=foshkplugin-gw1100
TITLE=FOSHKplugin –> TITLE=FOSHKplugin-GW1100

1. geänderte plugin.cfg wieder in die ZIP-Datei einpacken, dabei die bereits vorhandene Datei überschreiben

Anschließend lässt sich dieses neue Plugin auf dem üblichen Weg in der Plugin-Verwaltung des Loxberry als Plugin installieren.

Zu beachten ist, das der parallele Betrieb von mehreren FOSHKplugin-Installationen tatsächlich getrennt erfolgt - jedes FOSHKplugin verweist also auf eine andere Wetterstation, hat einen eigenen Port zur Datenentgegennahme (HTTP-Port des LoxBerry) und erfordert auch einen separaten UDP-Sendeport (UDP-Port des Zielsystems) mit eigenen virtuellen Eingängen im Loxone-Miniserver.
Bei der Plugin-Installation wird die jeweils zuerst gefundene Wetterstation in die Plugin-Konfiguration geschrieben - hier sollte also geprüft werden, ob die jeweiligen Plugins tatsächlich auf unterschiedliche Stationen verweisen (IP-Adresse der Station) und ggf. mit dem Button WS-Set die korrigierte Konfiguration in die Wetterstation geschrieben werden.
Zur Steuerung des Plugins von Loxone aus ist der UDP-Port des Plugins (UDP-Port des LoxBerry) ebenfalls eindeutig (einmalig) zu konfigurieren.

Zusammengefasst:

UDP-Port des Zielsystems (der Port, auf dem der Loxone MS die eingehenden Daten erwartet)
HTTP-Port des LoxBerry (der Port, auf dem FOSHKplugin eingehende Daten der Wetterstation erwartet)
UDP-Port des LoxBerry (der Port, auf dem FOSHKplugin auf etwaige Steuerbefehle vom Loxone-MS lauscht)
sowie
IP-Adresse der Station (die IP-Adresse zum Schreiben der Konfiguration in die Wetterstation)

sollten bei den einzelnen Plugin-Installationen unbedingt **UNTERSCHIEDLICH** sein!

Ich habe auch ein kleines Script cloneFOSHKplugin.bat gebastelt, das diese Änderungen mehr oder weniger automatisch vornehmen kann. Allerdings ist die Änderung der drei Einträge in der Datei plugin.cfg auch manuell sehr schnell gemacht. Das Script erzeugt aber ein Paket mit neuem Namen und sollte (!) weniger anfällig für etwaige Fehler sein.

Man kann sich den Link (oder das Script selbst) auf den Windows-Desktop packen und kann das ZIP-File des LoxBerry-Plugins FOSHKplugin per drag&drop einfach darauf fallenlassen.
Es sollte sich dann ein Dos-Fenster öffnen und nach NAME und TITLE des neu zu erstellenden - geclonten - Plugins fragen.
Das fertige "geclonte" Paket findet sich anschließend dann an der Stelle, wo das Ursprungspaket war (vermutlich der Download-Ordner).

Im ZIP-File enthalten ist auch eine Textdatei cloneFOSHKplugin.txt mit weiteren Hinweisen zum Clonen des FOSHKplugin-LoxBerry-Plugins.

Roadmap

... ein paar Ideen habe ich noch; auf Eure Wünsche bin ich gespannt ...

Fragen stellen und Fehler melden

Im Loxforum gibt es für dieses Plugin einen eigenen
Thread: https://www.loxforum.com/forum/projektforen/loxberry/plugins/222662 - ich und auch andere Plugin-Nutzer lesen dort mit und helfen gern bei Fragen und Problemen. Bitte aber immer mit möglichst genauer Fehlermeldung oder -beschreibung und Angabe der genutzten Version und Typ und Hersteller der Wetterstation und mit Screenshots und/oder Log-File-Ausschnitten zur Verdeutlichung des Problems.

Hilfe zur Selbsthilfe

Wenn die Kommunikation zwischen Wetterstation und FOSHKplugin oder FOSHKplugin und Loxone-Server nicht klappt, bitte zuerst nochmal an die Arbeitsweise dieses Systems denken.
Wir haben 3 Geräte im Einsatz mit folgendem Datenfluss:

Wetterstation	–>	FOSHKplugin auf LoxBerry	–>	Loxone-Miniserver

Diese 3 Geräte haben unterschiedliche IP-Adressen und erfordern verschiedene Ports:

Wetterstation mit IP-Adresse 1		FOSHKplugin auf LoxBerry mit IP-Adresse 2		Loxone-Miniserver mit IP-Adresse 3
http-Sendeport 1	–>	http-Empfangsport 1		
		UDP-Sendeport 1	–>	UDP-Empfangsport 2

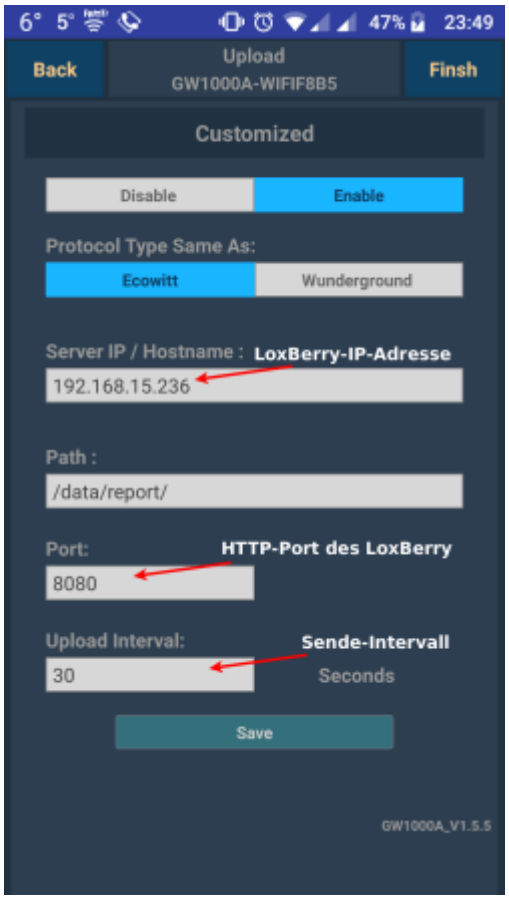
Die Wetterstation sendet also die Sensordaten per http an den LoxBerry, auf dem das Plugin FOSHKplugin läuft.
Dieses Plugin nimmt die Daten per http auf dem unter " ***HTTP-Port des LoxBerry:*** " konfigurierten Port entgegen, wandelt diese um und sendet den Datensatz dann per UDP an den unter " ***UDP-Port des Zielsystems:*** " konfigurierten Port an das unter " ***IP-Adresse des Zielsystems:*** " eingetragene Ziel (hier der Loxone-MS).
Das Ziel der Wetterstation ist also die IP-Adresse und der Port des LoxBerry.
Und das Ziel des FOSHKplugin stellt der Loxone-Server dar.

Diese IP-Adressen und Ports dürfen nicht verwechselt werden!

Ansonsten empfiehlt sich, die Problemforschung von der Quelle zum Ziel durchzuführen:

1. Prüfen, ob die Sensordaten in der WS View-App angezeigt werden
2. Prüfen der Einstellungen für den Customized Service in WS View.

Geh dazu bitte in die WS View App, wähle Deine Station und geh zu More und dann weiter zu Weather Services.
Mit viermal ***Next*** solltest Du zu den Einstellungen für den ***Customized Service*** gelangen.
Dort sollte dann ***Customized*** enabled und ***Protocol*** auf Ecowitt eingestellt sein.
Die angegebene IP-Adresse bei **Server IP/Hostname** sollte der IP-Adresse Deines LoxBerrys entsprechen und als ***Path*** sollte /data/report/ eingetragen sein.
Auch die Portnummer unter **Port** sollte dem im Plugin konfigurierten " ***HTTP-Port des LoxBerry*** " entsprechen.
Solltest Du die IP-Adresse Deines LoxBerry nicht kennen, klicke auf den Button " ***Erkenne LB*** " unter **"*Optionale Einstellungen*"** des Plugins.



Sind die Daten im WS View soweit korrekt eingetragen, sollten - von etwaigen Netzwerkproblemen abgesehen - die Daten von der Wetterstation korrekt verschickt werden.

Dann ist FOSHKplugin/LoxBerry zu prüfen:
Erster Anlaufpunkt bei Problemen sollte das interne Logging des FOSHKplugin sein. Unter **"*Optionale Einstellungen*"** gibt es dazu 3 unterschiedliche Log-Files:

Im " ***Standard-Log*** " werden Start und Stopp des Plugins protokolliert. Auch etwaige Fehlermeldungen, Warnungen und eingehende Meldungen erscheinen dort.
Im " ***WS-Empfangs-Log*** " werden alle von der Wetterstation entgegengenommenen Daten

(Rohdaten) mitgeschrieben.
Und im " ***Export-Log*** " erscheinen alle vom Plugin nach außen geschickte Daten - nebst etwaigen Export-spezifischen Fehlermeldungen.

Gibt es hier keinerlei Hinweise auf irgendwelche Probleme wird es knifflig, ich benötige dann die Log-Files und ggf. Screenshots sowie Hintergrundinformationen (LoxBerry-Version, Image oder selbstaufgezogen, echte Hardware oder virtuelle Maschine, andere Plugins, sonstige Seltsamkeiten) um helfen zu können.
Hilfreich kann hier auch das LoxBerry-Apache-Log sein. Zu finden ist es im Log-Manager von LoxBerry unter " ***Apache Log*** ".
Kommen jedoch Daten von der Wetterstation im " ***WS-Empfangs-Log*** " an, sind die Daten zumindest schonmal im Plugin.

Die Kommunikation zwischen FOSHKplugin und dem Zielsystem gilt es nun zu prüfen:
In den Einstellungen des FOSHKplugin ist also sicherzustellen, dass die unter " ***IP-Adresse des Zielsystems*** " angegebene Adresse tatsächlich die des Ziels (also der Loxone-Server) ist und dieses auch erreichbar ist.
Desweiteren sollte der unter " ***UDP-Port des Zielsystems*** " angegebene Port wirklich der Port sein, auf dem das Zielsystem (Loxone-Server) die eingehenden Nachrichten an den virtuellen Eingängen erwartet.

kompatible Wetterstationen

Es sollten alle Wetterstationen unterstützt sein, deren Konfiguration über die WS View-App erfolgt und bei der man ein benutzerdefiniertes Ziel eintragen kann (Weather Service: Customized). Eventuell funktioniert es sogar bei Stationen, die den Customized-Modus einfach nur ausblenden. Da ich diese Stationen jedoch nicht mit FOSHKplugin getestet habe, kann ich keine Gewähr dafür geben. Das müsste man ggf. auf eigene Gefahr ausprobieren.

Laut den verfügbaren Anleitungen sollten die Wetterstationen von Froggit WH3000 SE, WH4000 SE und HP1000SE PRO - auch bei Übermittlung im Ecowitt-Format - mit FOSHKplugin kompatibel sein.

Die Froggit WH2600 SE LAN überträgt wohl ausschließlich im WU-Format - ein customized Server ist aber einstellbar. Somit sollte auch diese Station mit FOSHKplugin funktionieren. Vermutlich muss man aber hier die Einstellung von Server, Port und Intervall in der App tätigen.

In den Anleitungen zur neuen WH5500 und WH6000 finde ich keine Informationen zur Einstellung eines Customized Servers. Somit werden diese Stationen wohl nicht mit FOSHKplugin funktionieren. Offenbar stammen diese Stationen auch nicht von FOSHK sondern vom chinesischen Hersteller CCL ...

Sicher funktionieren sollten jedoch:

- Froggit DP1500 (GW1000) uneingeschränkt - hier in Betrieb mit WH3000SE (WH65), DP50 (WH31), DP100 (WH51), WH41 (DP200), WH55 (DP70) und WH57 (DP60)
- Ecowitt GW1000 da baugleich
- Froggit HP1000SE PRO WiFI Wetterstation (aktuelle Version)
- Sainlogic 7 in 1
- WS980WiFi von ELV
- Eurochron EFWS 2900

Noch ein Hinweis bzgl. Hersteller FOSHK und "kompatible" Wetterstationen:
Der chinesische Hersteller FOSHK verkauft seine Produkte an eine Vielzahl von Weiterverkäufern, die diese Geräte umlabeln (oder selbst das sogar lassen) und unter eigenem Namen anbieten. Inwieweit die Verkäufer dann tatsächlich noch irgendwelche Anpassungen vornehmen oder die Software beeinflussen, kann ich nicht sagen. Rein äußerlich sehen jedenfalls die Wetterstationen ELV Ventus W830, Sainlogic, ChiliTec, Conrad Eurochron EFWS 2900 oder Waldbeck Huygens der WH3000SE von Froggit sehr ähnlich. Die ELV WS980WiFi sieht einer Froggit WH4000 verdammt ähnlich und eine dnt WiFi-Wetterstation WeatherScreen PRO sieht doch beinahe wie eine HP1000SE von Froggit aus. In den zum Teil vorab verfügbaren Bedienungsanleitungen sollte man erkennen können, ob die Wetterstation einen Customized Weather Service (idealerweise im Ecowitt-Format) anbietet. Ist das gegeben, sollte die Anbindung via FOSHKplugin kein Problem darstellen.

Rechtliche Hinweise

Ich übernehme keine Garantien hinsichtlich des Einsatzes dieser Software - die Nutzung geschieht auf eigene Gefahr. Treffen Sie Entscheidungen die zu Personen- oder Sachschäden führen können niemals auf Grundlage dieser Software.
Durch das Programm generierte Warnungen (z.B. Sturm oder Gewitter) können eintreffen. Das Fehlen dieser Warnungen impliziert jedoch nicht, dass diese Dinge nicht möglich sind.

Links

Link zur Android-App WS View im Google PlayStore (externer Link)
Die WS View-App für Geräte mit dem Apfel (externer Link)
Die Icon-Sammlung, die Grundlage für das hier genutzte Icon ist (externer Link)
Das "generic"-Plugin falls eine LoxBerry-Installation nicht zur Verfügung steht (externer Link)

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

Permanent link:
https://wiki.loxberry.de/plugins/foshkplugin/start

Last update: **2024/03/26 19:28**