

**WIN-
DIGIPET**



**PREMIUM
EDITION 2025**

MQTT in Win-Digipet

Inhalt

Grundlagen	2
MQTT-Server/Broker einrichten	3
Nachträglich den MQTT-Server/Broker mit einem Passwort absichern	5
Mit dem MQTT Broker eine Testverbindung herstellen	5
Befehle für die Kommunikation von WDP über MQTT	6
Demo-Schaltung 1	9
Einbindung von Drittanbieter-Produkten via MQTT in Win-Digipet	11
Noch mehr Freiheitsgrade, komplett individuell....	15

Grundlagen

Sehr oft hat uns die Frage die letzten Jahre über ereilt: „Wie kann ich denn einfach mein Raumlicht über Win-Digipet ansteuern“ oder „Ich habe bereits ein Smart-Home-System, kann ich über Win-Digipet Einfluss darauf nehmen, um z.B. das Licht im Moba-Zimmer zu steuern“.

Neben den Lösungen zur Raumlicht-/Effektiv-Licht-Steuerung einiger Modellbahnanbieter welche z.B. die Ansteuerung von DMX-Komponenten ermöglichen (z.B. LoDi DMX-Interface, LS Digital DMX-Interface, OpenDMX/ReadyDMX für BiDiB) unterstützt Win-Digipet ab Version 2025 das offene Netzwerkprotokoll MQTT. MQTT ist ein offenes Netzwerk-Protokoll welches von vielen Smart-Home-/IOT-Systemen verstanden wird bzw. über welches mit diesen Systemen kommuniziert werden kann.

Die MQTT-Unterstützung in Win-Digipet richtet sich ausdrücklich an „Elektronik-Bastler“ welche entweder MQTT bereits nutzen oder Interesse haben sich damit zu beschäftigen. Es können auch „relativ“ einfach selbst erstellte Arduino-Geräte darüber eine Kommunikation mit WDP aufnehmen. Siehe weiter unten.

In einem MQTT-System gibt es stets einen zentralen Server („Broker“) mit dem Clients (in unserem Fall Win-Digipet und externe Geräte) kommunizieren. In Win-Digipet wird die Verbindung zu einem MQTT-Server wie ein normales Digitalsystem in den Systemeinstellungen erfasst. Hierbei ist zwingend erforderlich die IP-Adresse des MQTT-Server und die Portnummer anzugeben (Standardwert 1883). Des Weiteren muss sofern im MQTT-Server konfiguriert, noch ein Benutzername und Passwort hinterlegt werden. Außerdem muss ein sogenanntes MQTT-Thema (Topic) unter welchem Win-Digipet seine Standardnachrichten versendet eingestellt werden. In allen folgenden Beispielen wurde als (Ober-)Topic in den Systemeinstellungen Win-Digipet der Text „WDP“ hinterlegt.

Dies gilt für alle Nachrichten außer wenn man bei den Magnetartikeln das Format „MQTT Direkt“ auswählt. Wählt man dieses Format aus, dann kann man auch Nachrichten Dritt-Produkte-Hersteller senden. Dazu aber später mehr.

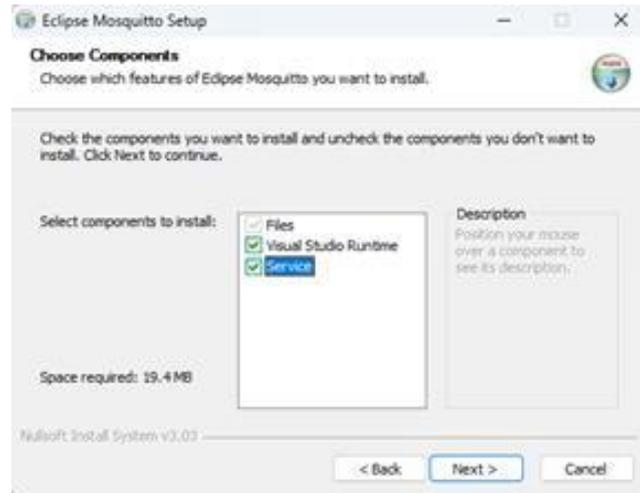
MQTT-Server/Broker einrichten

Einen MQTT-Server/Broker für ein Smart-Home betreibt man idealerweise auf einem System welches 24/7 läuft wie z.B. ein Raspberry Pi oder auch PC. Anleitungen zur Einrichtung findet man zahlreich im Netz.

Im Folgenden wird die Installation eines Servers auf einem Windows-Rechner erläutert, den man fallweise manuell starten kann. Dies ist nur relevant für diejenigen die mal ohne großen Aufwand eine MQTT-Verbindung testen wollen.

Zunächst einen Installer von dieser Seite herunterladen: <https://www.mosquitto.org/download/>

Ob nun 32 oder 64-bit Variante ist für Win-Digipet egal. Ich habe die 64 bit-Variante getestet. Den Installer-Dialog einfach durchlaufen. Auf dieser Seite kann man entscheiden ob der Broker als sogenannter Service/Dienst dauerhaft im Hintergrund laufen soll. Ich würde dies empfehlen. Denn insbesondere die Firewall ist dann manchmal gnädiger.



- Das Installationsverzeichnis merken z.B. „C:\Program Files\mosquitto“ bei 64-bit und „C:\Program Files (x86)\mosquitto“ bei 32-bit.
- Nun im Explorer das Installationsverzeichnis öffnen und dann die Datei **mosquitto.conf** mit einem Text-Editor bearbeiten (als Administrator aufrufen). Ggf. muss in der Computerverwaltung von Windows unter Dienste der Mosquitto Broker vorab gestoppt werden.
- In der **mosquitto.conf** am Ende oder mittendrin an der entsprechenden Stelle die Zeilen ergänzen.

allow_anonymous true

listener 1883 0.0.0.0

Somit verzichtet man für den ersten Versuch auf eine **Passwort-Abfrage** und ist über alle Schnittstellen des PCs erreichbar. Weiterhin muss man ggf. den Broker TCP-Port 1883 in seiner Firewall für das lokale Netzwerk freigeben.

- Nun muss man in der Computerverwaltung von Windows unter **Dienste** den Mosquitto Broker (wieder) starten.

Dieser sollte nun über die IP-Adresse des PCs von WDP aus erreichbar sein. Es empfiehlt sich grundsätzlich das Laufen des Brokers über ein MQTT-Test-Programm, zu testen. Für Windows z.B. <http://mqtt-explorer.com/>. Es gibt auch passende Apps für Android und iOS.

Interessante Videos zum Installieren eines MQTT-Clients unter Windows:

- <https://www.youtube.com/watch?v=Aquq8dwNsFw>
- <https://www.youtube.com/watch?v=odOpqM5-BDI>
- <https://www.youtube.com/watch?v=PMa65FxWbY>

Es gibt im Netz auf Youtube sehr viele Videos auch zur Einrichtung auf einem Raspberry Pi.

Nachträglich den MQTT-Server/Broker mit einem Passwort absichern

- In der Computerverwaltung von Windows unter **Dienste** den Mosquitto Broker anhalten.
- In der **mosquitto.conf** am Ende oder mittendrin an der entsprechenden Stelle die Zeilen ergänzen bzw ändern.

(bei password_file den jeweiligen Pfad eingeben)

allow_anonymous false

port 1883

password_file C:\Program Files\mosquitto\passwd

- Man öffnet in Windows die „Eingabeaufforderung“ als Administrator und wechselt in das mosquitto Verzeichnis.

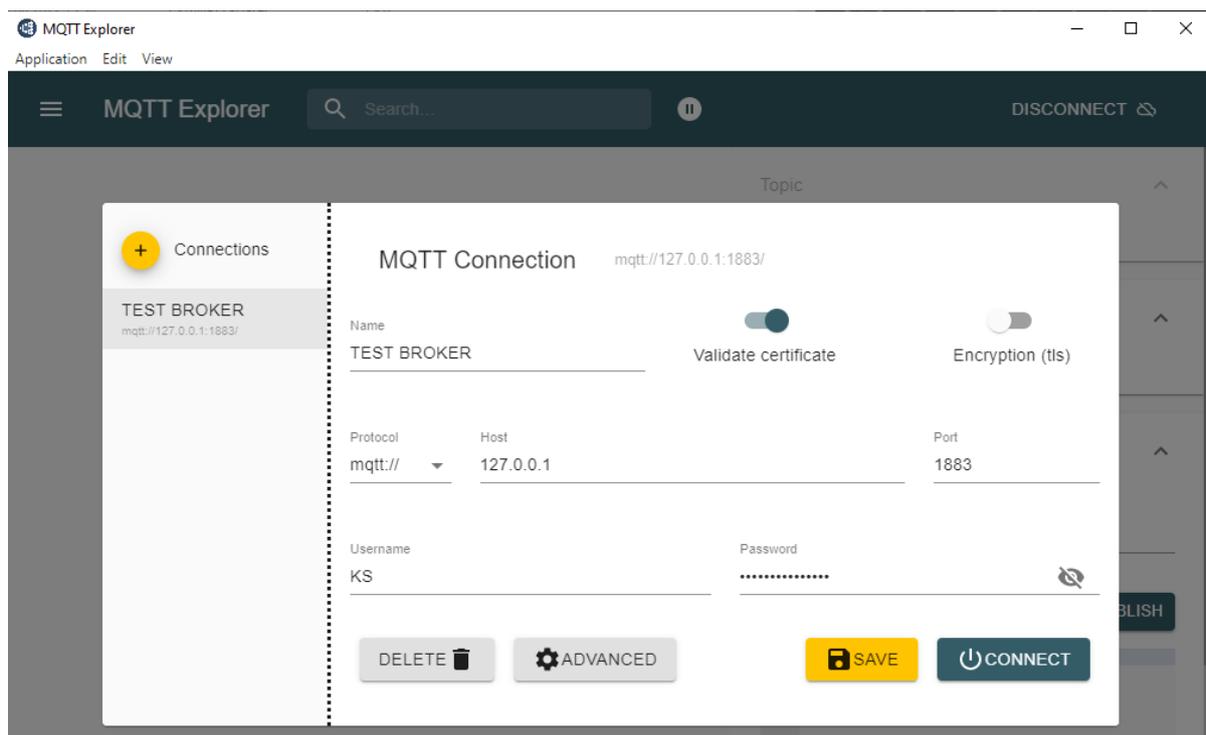
zB „C:\Program Files\mosquitto“

Dort tippt man nun **mosquitto_passwd -c passwd USERNAME** ein.

Danach vergibt man für den MQTT Broker ein Passwort nach seinem Wunsch.

- In der Computerverwaltung von Windows unter **Dienste** den Mosquitto Broker wieder starten.

Mit dem MQTT Broker eine Testverbindung herstellen



Wenn der MQTT Server auf dem PC läuft, ist die IP Adresse 127.0.0.1 (Localhost)
Durch Eingabe von Username und Passwort sollte eine Verbindung zum MQTT Server aufgebaut werden.

Befehle für die Kommunikation von WDP über MQTT

Nachrichten im MQTT-System bestehen immer aus einem Topic und dem Nachrichteninhalt. Beispiele:

- Bei Auslösung des Digitalsystem-Stop in WDP wird an das Topic „WDP/State“ die Nachricht „stop“ gesandt
- Bei Auslösung des Digitalsystem-Go in WDP wird an das Topic „WDP/State“ die Nachricht „go“ gesandt

Die folgenden Topics werden von Win-Digipet standardmäßig gesandt/ausgewertet:

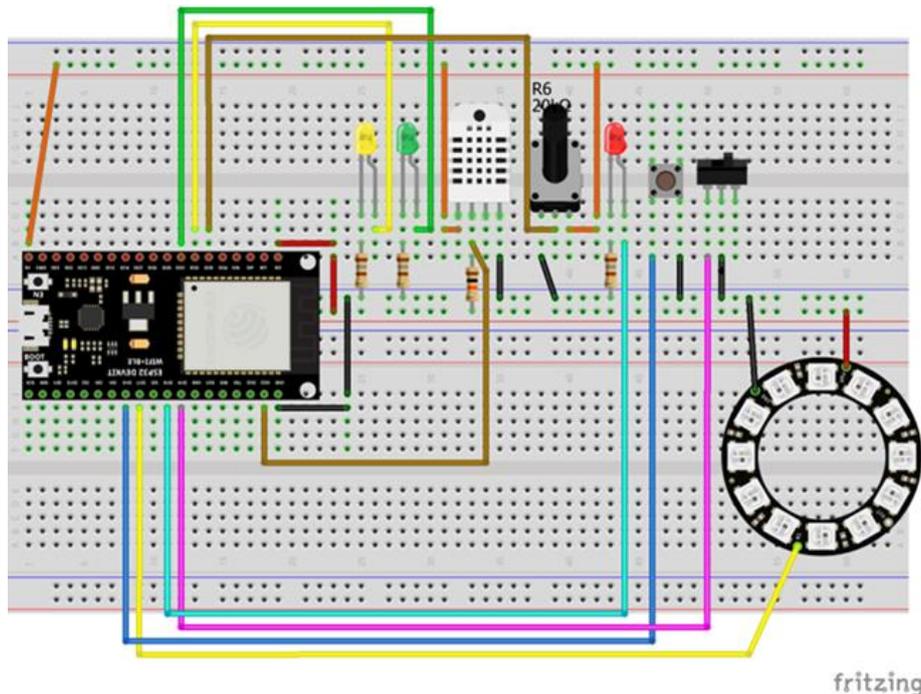
Topic	Richtung	Inhalt/Erläuterung
WDP/OnlineState	Senden	„online“ wenn WDP gestartet „offline“ wenn WDP beendet
WDP/Clock	Senden	Aktuelle Modellbahnzeit
WDP/InitTime	Senden	Einmaliges Senden der aktuellen Systemzeit in Unixseconds (Sekunden seit dem 01.01.1970), hiermit kann ein Gerät erkennen, dass WDP gestartet wurde und ggf. seinen aktuellen Status Senden
WDP/Cmd/SD/Nr/State	Senden	Nr im Topic wird ersetzt durch die Nummer eines Magnetartikels in WDP welcher dem MQTT-Digitalsystem zugeordnet wurde. Der Nachrichteninhalt ist eine Zahl von 0..x, diese Zahl stellt die aktuelle Magnetartikel-Stellung (beginnend bei 0 dar) bzw. im Falle eines Zählers den aktuellen Zählerwert Einstellung Digitalformat: MQTT WDP-Protokoll
WDP/Evt/SD/Nr/State	Empfang	Gegenstück zu „WDP/Cmd/SD/Nr/State“ Hierüber kann Win-Digipet Änderungen der Stellung des Magnetartikels mit der Nummer Nr empfangen. Nachrichteninhalt analog zur von Win-Digipet ausgesandten Nachricht Einstellung Digitalformat: MQTT WDP-Protokoll
WDP/Evt/Counter/ID_z/State WDP/Evt/Counter/X_Y/State	Empfang	Hierüber kann Win-Digipet Änderungen des Wertes eines Zählers empfangen Zähler wird identifiziert entweder über seine ID-Nr. z oder seine Koordinaten X_Y
WDP/Evt/FB/Nr/State	Empfang	Rückmelder-Nachricht des Rückmelders Nr an Win-Digipet Nachrichteninhalt standardmäßig 0/1 (0 -> frei, 1 bzw. alle anderen Werte als 0 -> besetzt)
WDP/Evt/FB/Nr/I_A WDP/Evt/FB/Nr/I_mA	Empfang	Stromverbrauch-Nachricht des Rückmelders Nr an Win-Digipet Meldung des Stromverbrauchs eines Rückmelders in A oder mA, Wert stets ohne Einheit zu Senden
WDP/Evt/FB/Nr/U_V	Empfang	Spannungs-Nachricht des Rückmelders Nr an Win-Digipet Meldung der Spannung eines Rückmelders in V, Wert stets ohne Einheit zu Senden

WDP/Evt/FB/Nr/T_C	Empfang	Temperatur-Nachricht des Rückmelders Nr an Win-Digipet Meldung der Temperatur eines Rückmelders in °C, Wert stets ohne Einheit zu Senden
WDP/Evt/PwSen/Nr/I_A WDP/Evt/PwSen/Nr/I_mA	Empfang	Stromverbrauch-Nachricht des Boosters/der Strommessung-Einrichtung Nr an Win-Digipet Meldung des Stromverbrauchs eines Boosters/einer Strommessung-Einrichtung in A oder mA, Wert stets ohne Einheit zu Senden
WDP/Evt/PwSen/Nr/U_V	Empfang	Spannungs-Nachricht des Boosters/der Strommessung-Einrichtung Nr an Win-Digipet Meldung der Spannung eines Boosters/einer Strommessung-Einrichtung in V, Wert stets ohne Einheit zu Senden
WDP/Evt/Memo/ID/State	Empfang	Memo mit ID ändern/erstellen Nachrichteninhalt kann JSON-Format haben, Beispiel: <pre>{ "text": "meintext", "duration": "0", "windowID": "1" }</pre> duration = Anzeigedauer in Minuten (kann auch weggelassen werden -> Wert 0 = dauerhaft) windowID = Memo-Fenster-ID Wertebereich 1-10 (kann auch weggelassen werden -> bekannter Wert für ID, sonst 1) Eine Nachricht welche nicht im JSON-Format ist d.h. kein { } am Anfang/Ende wird als reiner Text interpretiert für das Memo
WDP/Evt/Memo/ID/Delete	Empfang	Memo mit der ID löschen (analog STW), Meldungsinhalt muss die Zeichenfolge <i>delete</i> sein
WDP/Evt/Text/x_y/State	Empfang	Text an Stelle x/y im Gleisbild ändern Nachrichteninhalt muss JSON-Format haben, Beispiel: <pre>{ "text": "neuer Text", "bold": "1" }</pre> bold = Fett an/aus mit Wert 0/1 (kann auch weggelassen werden -> keine Änderung) weitere Attribute: italic = kursiv, strikeout = durchgestrichen, underline = unterstrichen, color1 = Zahl-RGB-Wert Text normal, color2 = RGB-Wert wenn RMK belegt (z.B. 255 rot, grün 65280) Eine Nachricht welche nicht im JSON-Format ist d.h. kein { } am Anfang/Ende wird als reiner Text interpretiert für das Gleisbild

WDP/Evt/PwSen/Nr/T_C	Empfang	Temperatur-Nachricht des Boosters/der Strommessung-Einrichtung Nr an Win-Digipet Meldung der Temperatur eines Boosters/einer Strommessung-Einrichtung in °C, Wert stets ohne Einheit zu Senden
WDP/Custom/...		Alle Nachrichten welche mit diesem Topic-Text beginnen werden seitens WDP (auch in Zukunft) niemals vorbelegt werden und können somit frei in eigenen Projekten/Ideen genutzt werden
WDP/....	Senden/ Empfang	Alle weiteren Nachrichten welche mit dem in WDP eingegebenen Topic (in diesem Beispiel WDP) beginnen gelten als reserviert für spätere Erweiterungen

Demo-Schaltung 1

Um dem ambitionierten Elektronik-Bastler den Einstieg zu erleichtern wurde eine kleine Schaltung für einen ESP32 Mikrokontroller entworfen. Die Verdrahtung auf einem Breadboard als auch der Arduino-Quellcode sowie ein zugehöriges Win-Digipet Projekt werden zur Verfügung gestellt.



Folgende Bauteile wurden verwendet:

- 1 x ESP32
- 3 x LED 5mm in rot/gelb/grün
- 3 x Widerstand 180Ω als Vorwiderstand für die LED
- 1 x Widerstand 10kΩ (dient als Pull-up-Widerstand)
- 1 x Potentiometer 20kΩ (möglichst nicht logarithmische Charakteristik)
- 1 x Taster
- 1 x Umschalter (Ein/Aus reicht)
- 1 x Temperatur/Feuchte-Sensor DHT22
- 1 x sogenannter NeoPixel Ring mit 12 RGB-LED WS2812B
- 2 x Breadboard

Für unseren Aufbau wurden Bauteile via Reichelt Elektronik bzw. Amazon bezogen

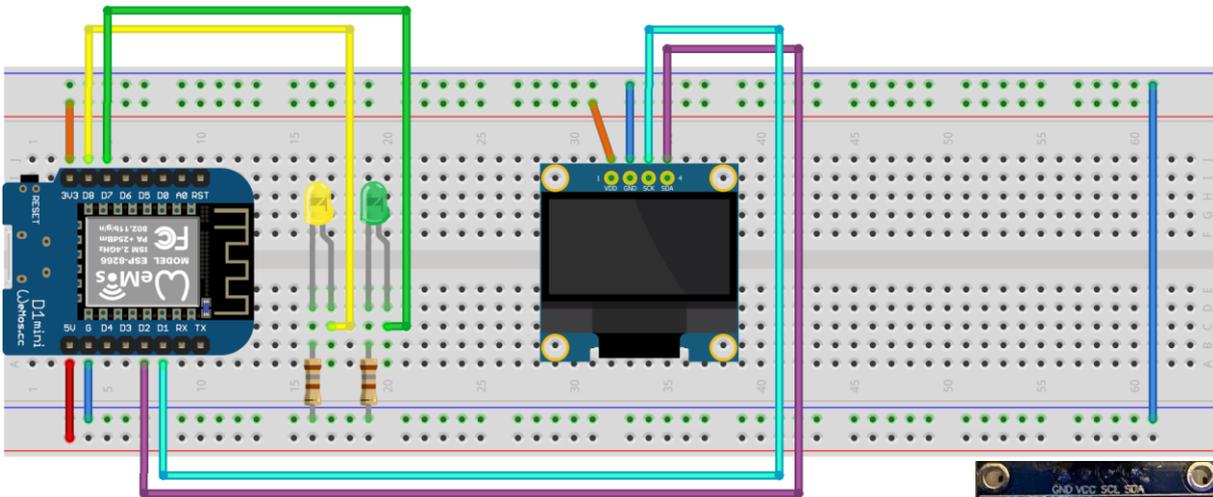
Ziel dieser kleinen DEMO Schaltung ist es die Möglichkeiten der Kommunikation zwischen WDP und einem Arduino Mikrokontroller zu verdeutlichen. Mit WDP können somit Schaltaktionen über Magnetartikel gesendet werden, die der Arduino ausführt. Es können analoge Regelwerte an WDP übertragen werden. Weiters können Schalterstellungen und Taster Auslösungen an WDP gesendet und von WDP verarbeitet werden.

Noch ein Hinweis zu den Schaltungen. Durch die Verwendung von NeoPixel und Displays, sollte auf die Verwendung gut stabilisierter Netzteile mit ausreichender Leistung geachtet werden. Einige gute Erläuterungen und Hinweise kann man hier finden:<https://arduino-hannover.de/2018/07/25/die-tuecken-der-esp32-stromversorgung/>

Demo-Schaltung 2

Die zweite Schaltung dient der Verdeutlichung von „Custom“-Befehlen. Hierbei werden über einen Stellwerker Beispiel-Texte an ein LCD-Display übertragen. Folgende Behle werden genutzt:

- WDP/Custom/Display/1/Row/1 -> im Display Nr.1 Text in Zeile 1 schreiben
- WDP/Custom/Display/1/Row/2 -> im Display Nr.1 Text in Zeile 2 schreiben
- WDP/Custom/Display/1/Row/3 -> im Display Nr.1 Text in Zeile 3 schreiben
- WDP/Custom/Display/1/Row/4 -> im Display Nr.1 Text in Zeile 4 schreiben
- WDP/Custom/Display/1/Row/5 -> im Display Nr.1 Text in Zeile 5 schreiben
- WDP/Custom/Display/1/Clear (mit beliebigem Nachrichteninhalte) -> Display leeren
- WDP/Custom/Display/1/Line (0 oder 1 als Nachrichteninhalte) -> Trennlinien zwischen den Zeilen zeichnen



- 1 x Wemos D1 mini
- 2 x LED 5mm in gelb/grün
- 2 x Widerstand 180Ω als Vorwiderstand für die LED
- 0,96 Zoll OLED SSD1306 Display I2C 128 x 64 Pixel
- 1 x Breadboard



Beispielausgabe

Für unseren Probeaufbau wurden Bauteile via Reichelt Elektronik bzw. Amazon bezogen.

Achtung: es gibt Varianten des OLED wo die Pins GND und VCC eine andere Reihenfolge als im Schaltbild haben, siehe auch Beispielbild rechts.

Wichtig ist, dass beim gleichzeitigen Betrieb mehrerer Arduino-Projekte, jedes Gerät in der config.h eine andere „clientid“ bekommt.

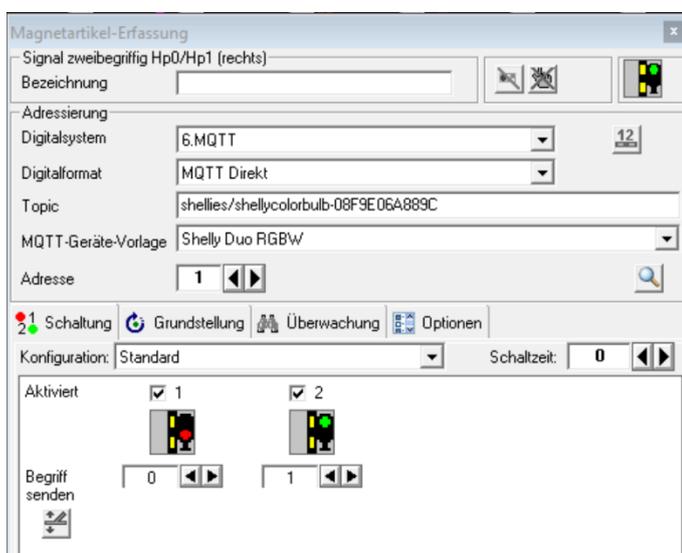
Einbindung von Drittanbieter-Produkten via MQTT in Win-Digipet

Neben der Einbindung von selbst erstellter Hardware wollen wir auch eine Einbindung von Drittanbieter-Produkten via MQTT in WDP erklären. Wir haben hier Möglichkeiten geschaffen einige gängige Produkte einzubinden, sofern die MQTT-Umsetzungen der Produkte mit den MQTT-Möglichkeiten in Win-Digipet kompatibel sind.

Wählt man in den Einstellungen eines Magnetartikel/Zählers im Gleisbildeeditor von Win-Digipet das Format „MQTT Direkt“ dann kann man einige über Vorlagen zur Verfügung gestellte Drittanbieter-Produkte über Win-Digipet steuern. Für den Anfang sind dies vor Allem Produkte des im Smarthome-Bereich weit verbreiteten Herstellers Shelly, da diese beim Autor auch bereits im breiten Einsatz sind. Eine Erweiterung um weitere Produkte ist denkbar sofern der Aufwand vertretbar ist und es sich im vom Protokoll her umsetzen lässt.

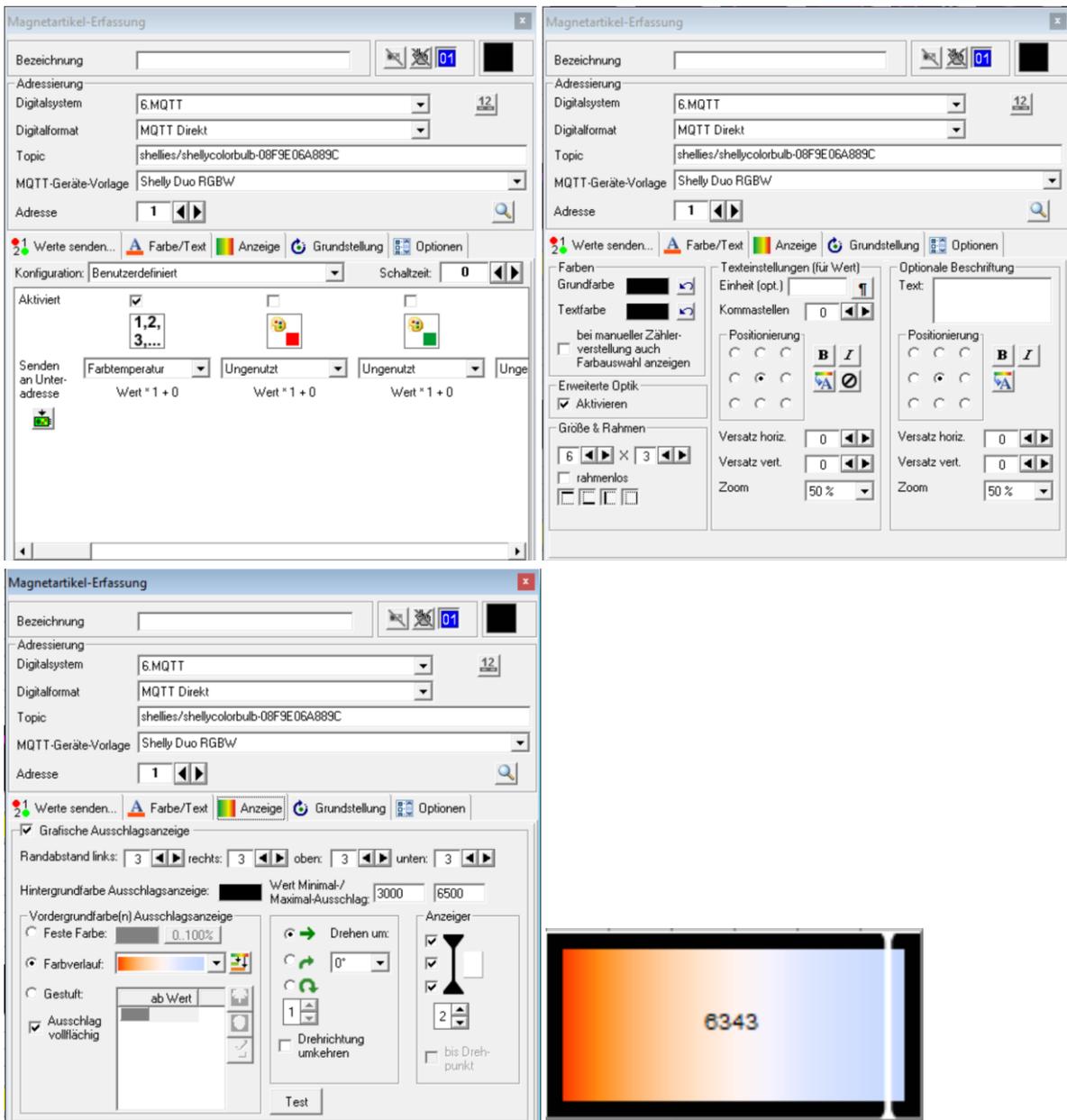
Voraussetzung ist natürlich, dass das jeweilige Drittanbieter-Produkt beim gleichen MQTT-Broker wie Win-Digipet angemeldet und das Topic unter welchem das Produkt erreichbar ist bekannt ist. Außerdem muss bei den neueren Shelly-Produkten in der MQTT-Konfiguration der Haken bei „Generic status update over MQTT“ aktiviert sein.

Im Folgenden sind einige Beispielkonfigurationen dargestellt:



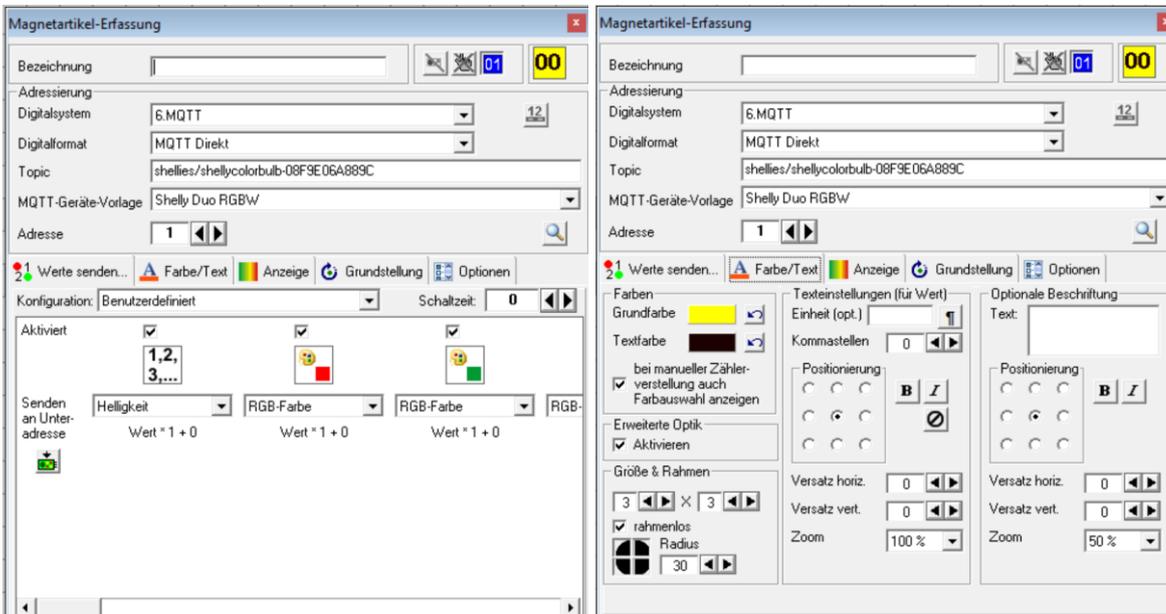
Signal-Symbol zum Ein-/Ausschalten einer Shelly Duo RGBW-Glühbirne welche via Topic ‚shellies/shellycolorbulb-08F9E06A889C‘ erreichbar ist. Hat ein Gerät z.B. ein Dimmer zum Anschluss von Glühbirnen mehrere Ausgänge erreicht man den ersten Ausgang via Adresse 1, den zweiten via Adresse 2 usw.

Eine besondere Funktion bei den Shellies hätte noch ein Taster auf Adresse 255 mit Begriff 255. Dahinter verbirgt sich dann das Anfordern des aktuellen Status des Shellies (als z.B. sofortige Aktualisierung nochmal der Messwerte die im folgenden gezeigt werden, die werden zwar periodisch ausgesandt, aber so erzwingt man ein Update).



Wieder die gleiche Shelly Duo RGBW Lampe wie im vorherigen Beispiel, diesmal aber bei einem Zähler hinterlegt über den man die Farbtemperatur der Lampe einstellen kann zwischen 3000 und 6500 Kelvin.

Zur Testausendung einmal oben rechts in das dunkle Feld klicken und eine Farbtemperatur eintippen und OK drücken.

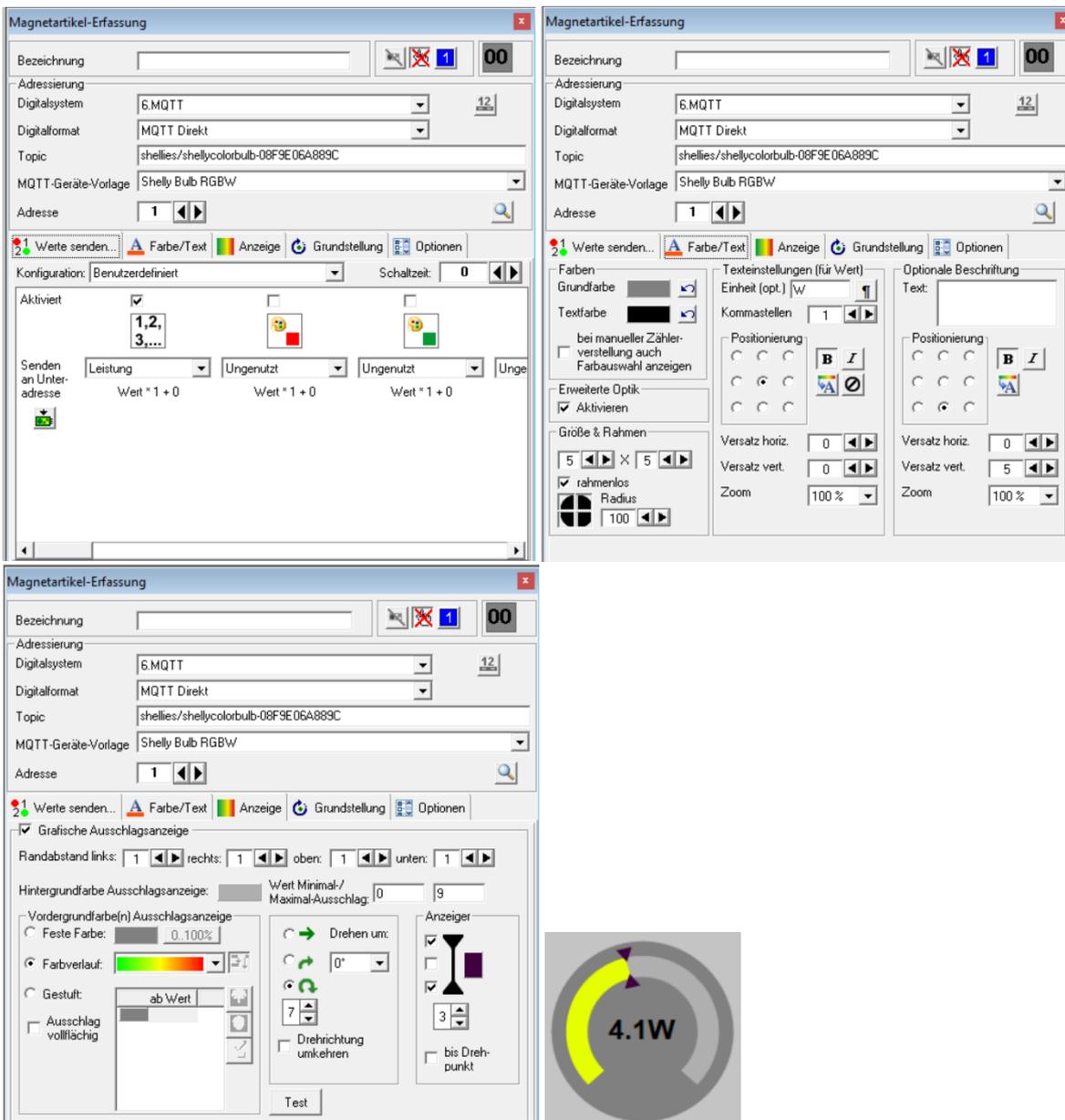


73

Und wieder die gleiche Shelly Duo RGBW Lampe wie im vorherigen Beispiel, diesmal bei einem weiteren Zähler hinterlegt dessen Wert dann die Helligkeit der Lampe in % beeinflussen kann. Und die Hintergrundfarbe des Zählers bestimmt die RGB-Farbe der Lampe (Wichtig: Wert RGB-Farbe ist zu hinterlegen bei dem roten, grünen und blauen Farbkanal in der Konfiguration, siehe Bild).

Zur Testaussendung der Helligkeit einmal oben rechts in das dunkle Feld klicken und eine Helligkeit zwischen 0 und 100% eintippen und OK drücken.

Zur Testaussendung der Farbe einmal auf der zweiten Karte die Grundfarbe ändern und dann auf der ersten Karte den Button  betätigen.



Und wieder die gleiche Shelly Duo RGBW Lampe wie im vorherigen Beispiel, diesmal bei einem weiteren Zähler hinterlegt. Dessen Wert dient dann aber nicht dazu etwas an der Lampe zu steuern, sondern nur etwas anzuzeigen. In diesem Falle die Leistungsaufnahme der Lampe. Ob man das jetzt bei Lampe an der Modellbahn braucht, mag natürlich jeder selber entscheiden. Aber die Shellys gibt es unterschiedlichsten Ausführungen welche auch viele unterschiedliche Messwerte liefern (Spannung, Leistung, Strom, Temperatur). Wieso nicht mit so einem Gerät mal die Leistungsaufnahme der gesamten Modellbahnanlage überwachen und visualisieren?

Es wurden Vorlagen für alle zum Zeitpunkt der Erstellung vorhandenen Funk-Schalter, Dimmer, Leuchten, LED-Streifen-Controller der Firma Shelly hinterlegt.

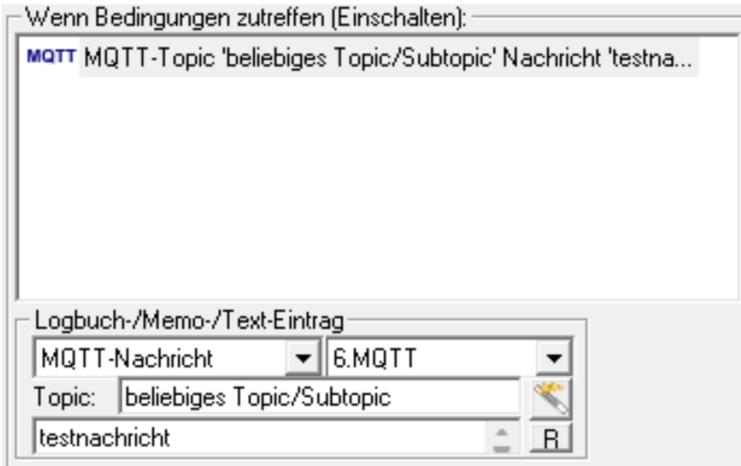
Ebenso gibt es Temperatur-Sensoren, so dass man die Raumtemperatur im Mobarraum sich darstellen lassen könnte usw. usw.

Noch mehr Freiheitsgrade, komplett individuell....

Und jetzt für diejenigen MQTT-Experten/Bastler die noch mehr Freiheitsgrade brauchen.

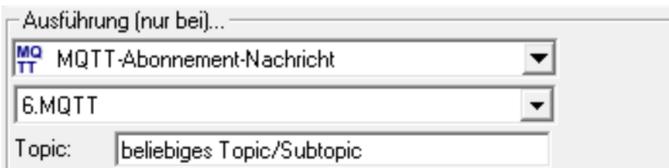
ACHTUNG: ABSOLUTES EXPERTENTUM!!!

Des Weiteren kann man über eine Schaltaktion im STW einen komplett freien Nachrichtentext an ein beliebiges Thema senden, Beispiel:



Diese Funktion und die folgenden Funktionen stehen nur in der Premium-Version zur Verfügung.

Will man wiederum den Inhalt einer beliebigen MQTT-Nachricht abgreifen, dann ist dies wie folgt möglich. Zunächst einmal muss man einen neuen STW-Eintrag anlegen und den Sonder-STW-Typ gemäß folgendem Bild wählen:



Hierüber wird das Abonnement des Thema angelegt. Den Inhalt kann man auswerten nutzen via:

1. Bedingung Stellwerkswärter-Status -> Auswahl des zuvor angelegten MQTT-Abonnement-STWs und dann „MQTT-Nachricht-Wert“, Beispiel:



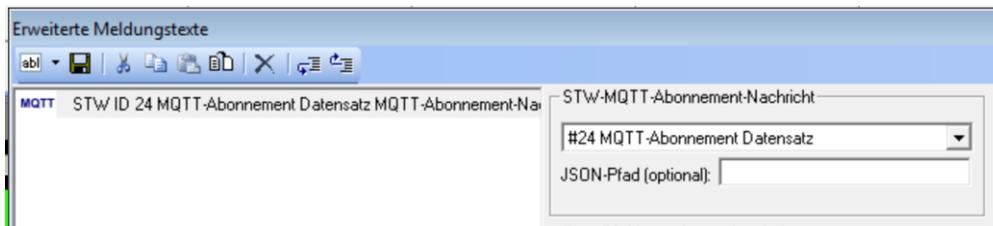
2. Berechnung Stellwerkswärter-Status -> Auswahl des zuvor angelegten MQTT-Abonnement-STWs und dann „MQTT-Nachricht-Wert“ analog zu 1.
3. Zusammengesetzten Textnachrichten (Erweiterte Meldungstexte), Beispiel Schaltaktion zum Setzen eines Textes im Gleisbild:



-> Zauberstab

-> Neue Zeile MQTT-Abonnement-Nachricht und dort wieder Verweis auf den zuvor angelegten MQTT-

Abonnement-STW, Beispiel:



Noch nicht erklärt wurde jetzt der auch den Screenshots zu sehende optionale Parameter JSON-Pfad. Viele MQTT-Nachrichten beinhalten mehrere Informationen, Beispiel Nachricht:

```
{"brightness":30,"ct":4447,"temperature":{"tC":39.1,"tF":102.4}}
```

Diese Nachricht erhält Informationen zur Helligkeit, Farbtemperatur und Gerätetemperatur einer Lampe.

Würde man nun keinen JSON-Pfad angeben, würde z.B. im Beispiel 3 der komplette Text inkl. geschweiften Klammern im Gleisbild ausgegeben. Gibt man aber nun folgende JSON-Pfade (bezogen auf das Beispiel):

- brightness -> Ausgabe des Wertes 30 (Helligkeit der Lampe)
- ct -> Ausgabe des Wertes 4447 (Farbtemperatur der Lampe)
- temperature.tC-> Ausgabe des Wertes 39.1 (Temperatur der Lampe in °C)
- temperature.tF -> Ausgabe des Wertes 102.4 (Temperatur der Lampe in °F)