DIVUS VISION API

DIVUS VISION API - Handbuch

Version 4.24

REV01-20240528

ALLGEMEINE INFORMATIONEN

DIVUS GmbH
Pillhof 51
I-39057 Eppan (BZ) - Italien

Änderungen des Handbuchs behalten wir uns ohne Vorankündigung vor. Die Fehlerfreiheit und Richtigkeit der in diesem Dokument und auf den mitgelieferten Speichermedien enthaltenen Daten können wir nicht garantieren. Anregungen zu Verbesserungen sowie Hinweise auf Fehler sind uns jederzeit willkommen. Die Vereinbarungen gelten auch für die speziellen Anhänge zu diesem Handbuch.

Die Bezeichnungen in diesem Dokument können Marken sein, deren Benutzung durch Dritte für eigene Zwecke die Rechte der Inhaber verletzen können.

Benutzerhinweise: Bitte lesen Sie das Handbuch vor dem ersten Einsatz und bewahren Sie es zur späteren Verwendung sorgfältig auf.

Zielgruppe: Das Handbuch ist für Anwender mit Vorkenntnissen in der PC- und Automatisierungstechnik geschrieben.

DARSTELLUNGSKONVENTIONEN

[TASTE]	Tasteneingaben des Benutzers werden in eckigen Klammern dargestellt, z.B. [STRG] oder [ENTF]
COURIER	Bildschirmausgaben werden in der Schriftart Courier beschrieben, z.B. `C:\>`
COURIER FETT	Tastatureingaben durch den Benutzer sind in Schriftart Courier fett beschrieben, z.B. **`C:\>DIR`**
"..."	Namen von auszuwählenden Schaltflächen, Menüs oder anderen Bildschirmelementen werden "..." wiedergegeben.
PIKTOGRAMME	Im Handbuch sind folgende Piktogramme zur Kennzeichnung bestimmter Textabschnitte verwendet:
	*Achtung!* Möglicherweise gefährliche Situation. Sachschäden können die Folge sein.
	*Notizen* Tipps und ergänzende Hinweise
	*Neu* Kennzeichnet Änderungen und neue Features

INHALTSVERZEICHNIS

1 Einleitung

1.1 ALLGEMEINE EINFÜHRUNG

Dieses Handbuch beschreibt die VISION-API (Application Programming Interface) - eine Schnittstelle, über die VISION von externen Systemen angesprochen und gesteuert werden kann.

In der Praxis bedeutet dies, dass Sie Systeme wie

MQTT Explorer (https://www.microsoft.com/store/... zum Testen),
Home Assistant (https://www.home-assistant.io/) oder
Node-RED (https://nodered.org/)

um die von VISION verwalteten Elemente zu steuern oder ihren Status auszulesen.

Der Zugriff und die Kommunikation erfolgen über das MQTT-Protokoll, das so genannte Topics verwendet, um einzelne Funktionen oder Funktionsgruppen anzusprechen oder über Änderungen an diesen informiert zu werden. Dazu wird ein MQTT-Server (Broker) eingesetzt, der die Sicherheit und die Verwaltung/Verteilung der Nachrichten an die Teilnehmer übernimmt. In diesem Fall befindet sich der MQTT-Server direkt auf dem DIVUS KNX IQ und ist speziell für diesen Zweck konfiguriert.

Obwohl die VISION API auch ohne Programmierkenntnisse genutzt werden kann, ist diese Funktionalität für fortgeschrittene Nutzer geeignet.

1.2 VORAUSSETZUNGEN

Wie im VISION-Handbuch erläutert, muss der API-Benutzer standardmäßig erst aktiviert werden, um ihn nutzen zu können - der API-Zugang funktioniert nur über die Authentifizierungsdaten des Api-Benutzers. Was die Benutzerrechte betrifft, so kann die Aktivierung für diese Funktionalität dann entweder für alle oder für einzelne Elemente konfiguriert werden. Siehe dazu Kap. .

Natürlich benötigen Sie auch ein VISION-Projekt, in dem die Elemente, die Sie von außen steuern wollen, vollständig konfiguriert sind und die Verbindung zu ihnen erfolgreich getestet wurde. Um einzelne Elemente über die API ansprechen zu können, muss deren Element-ID bekannt sein: Diese wird am unteren Rand des Einstellungsformulars des Elements angezeigt. Siehe dazu Kap. 1.4.

1.3 SICHERHEIT

Aus Sicherheitsgründen ist der API-Zugang nur lokal möglich (d.h. nicht über die Cloud). Das Sicherheitsrisiko bei der Aktivierung des API-Zugangs ist daher gering. Dennoch sollten sicherheitsrelevante Elemente für den API-Zugang nicht aktiviert oder explizit verweigert werden.

1.4 MQTT UND SEINE BEGRIFFE - KURZE ERLÄUTERUNG

Der Hauptunterschied zwischen MQTT und dem klassischen Client-Server-Modell, bei dem der Client Daten anfordert und dann ändert, liegt in den Konzepten ***subscribe*** und ***publish***. Die Teilnehmer können Daten veröffentlichen (publish) und sie so anderen zur Verfügung stellen, die sie - bei Interesse - abonnieren können (subscribe). Diese Architektur ermöglicht es, den Datenaustausch auf ein Minimum zu reduzieren und dennoch alle interessierten Parteien auf dem Laufenden zu halten. Mehr über die Details erfahren Sie hier:

BROKER

In MQTT ist der ***Broker*** für die zentrale Verwaltung und Verteilung aller Nachrichten zuständig. Obwohl MQTT-Server und MQTT-Broker keine Synonyme sind (Server ist ein weiter gefasster Begriff für eine Rolle, die auch MQTT-Clients spielen können), ist in diesem Handbuch immer der Broker gemeint, wenn von MQTT-Server die Rede ist. Das DIVUS KNX IQ selbst spielt im Rahmen dieses Handbuchs die Rolle des MQTT-Brokers / MQTT-Servers.

TOPICS

Ein MQTT-Server verwendet sogenannte ***Topics*** (Themen): eine hierarchische Struktur, mit der Daten kategorisiert, verwaltet und veröffentlicht werden.

PUBLISH

Das Veröffentlichen (***publish***) dient in erster Linie dazu, Daten über Topics für andere Teilnehmer verfügbar zu machen. Wenn Sie einen Wert ändern wollen, schreiben Sie in das gewünschte Topic mit der gewünschten Wertänderung, ebenfalls mit einer Publishing-Aktion. Das Zielgerät oder der MQTT-Server, lesen die gewünschte Änderung, die sie betrifft, und übernehmen sie entsprechend. Um zu überprüfen, ob die Änderung übernommen wurde, können Sie im abonnierten Topic nachsehen, ob sich die Änderung dort wiederspiegelt - wenn alles geklappt hat.

SUBSCRIBE

Die Kunden wählen die Topics aus, die sie interessieren: Dies wird als Abonnement (***subscribe***) bezeichnet. Jedes Mal, wenn sich ein Wert in/unter einem Topic ändert, werden alle abonnierten Clients informiert - d.h. ohne dass sie explizit nachfragen müssen, ob sich etwas geändert hat oder welcher der aktuelle Wert ist.

CLIENT_ID

Sie können einen separaten Kommunikationskanal mit dem MQTT-Server öffnen, indem Sie eine beliebige eindeutige Zeichenfolge namens ***client_id*** in ein Topic eingeben. Die client_id muss in dem Topic verwendet werden, um Werte zu bearbeiten. Dies dient zur Identifizierung der Herkunft jeder Änderung, hilft bei eventuellen Fehlern und hat keine Auswirkungen auf die anderen Clients, da die entsprechenden Antworten des Servers, einschließlich eventueller Fehlercodes und Nachrichten, ebenfalls nur das Topic mit der gleichen client_id (und damit im Normalfall nur diesen Client) erreichen.

Die client_id ist eine eindeutige Zeichenkette, die aus einer beliebigen Kombination der Zeichen 0-9, a-z, A-Z, "-", "_" besteht..

STATUS / REQUEST

In der Regel enthalten die Subscribe-Topics des MQTT-Servers des DIVUS KNX IQ das Schlüsselwort ***status***, während die Publish-Topics das Schlüsselwort ***request*** enthalten. Diejenigen mit status werden automatisch aktualisiert, sobald es eine externe Wertänderung gibt oder sobald eine Wertänderung vom Client selbst über ein Publish angefordert und erfolgreich durchgeführt worden ist. Die Topics werden weiter unterteilt in solche vom Typ *status/get* oder *request/get* (zum Auslesen) und solche vom Typ *status/set* oder *request/set* (zum Setzen).

PAYLOAD

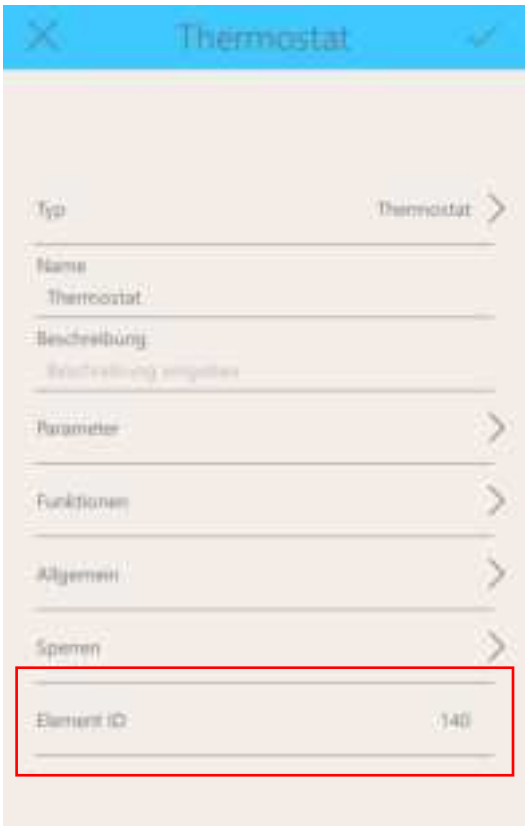
Wertänderungen und andere optionale Parameter werden dem Topic mit der sogenannten Payload hinzugefügt. Hier sind die Parameter der einzelnen Elemente (Element-Id, Name, Typ, Funktionen) und spezielle Parameter (uuid, Filter) zu verwenden. Obwohl es mehrere Möglichkeiten gibt, wird die Nutzlast in diesem Handbuch als JSON formatiert dargestellt. JSON verwendet Klammern und Kommas zur Darstellung von Daten beliebiger Struktur und minimiert so die Größe der zu übertragenden Datenpakete. Weitere Einzelheiten zu Nutzdaten finden Sie weiter unten im Handbuch.

FILTER

Für spezielle Zwecke ist es möglich, nach der Art der Funktion zu filtern, z. B. um nur Ein/Aus-Schalter, d. h. 1-Bit-Schalter, anzusprechen. Zu diesem Zweck wird der Parameter ***filters*** in der Nutzlast verwendet. Die Filterung ist derzeit nur nach Funktionstyp möglich.

ELEMENT-ID

Um einzelne Elemente ansprechen zu können, wird deren ***Element-ID*** benötigt. Diese ist in VISION im Menü der Elementeigenschaften zu finden oder kann auch direkt aus den Daten abgelesen werden, die im allgemeinen Abonnement des MQTT-Explorers vor jedem verfügbaren Element angezeigt werden (die Elemente sind dort alphabetisch nach Element-ID aufgelistet).



2 Konfiguration für den API-Benutzer-Zugriff

2.1 KONFIGURATION VON VISION FÜR DEN API-BENUTZERZUGRIFF

Gehen Sie in VISION als Administrator zu Konfiguration - Benutzer-/API-Zugangsverwaltung, klicken Sie auf Benutzer/API-Zugang und klicken Sie mit der rechten Maustaste auf API-Benutzer (oder halten Sie die Taste gedrückt), um das Bearbeitungsfenster zu öffnen.

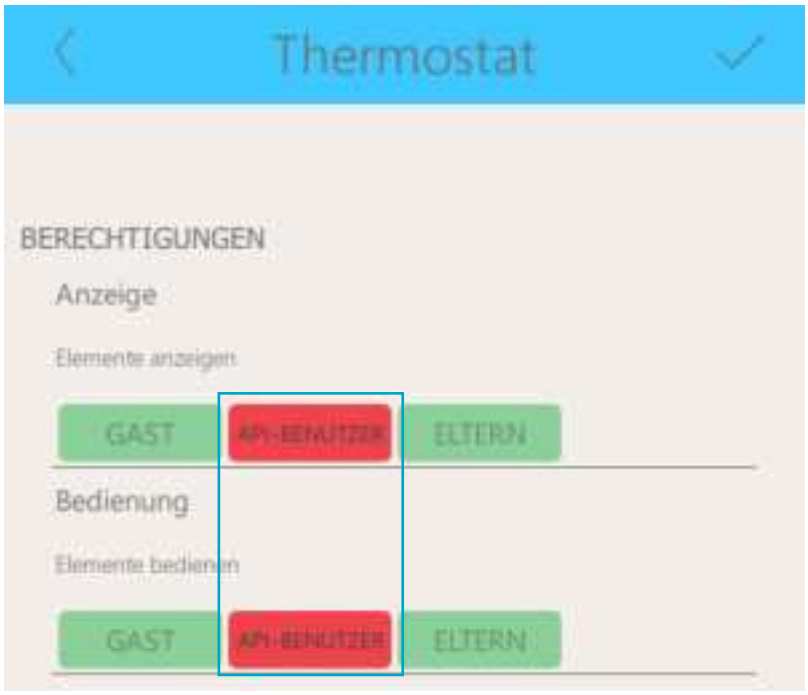
Dort finden Sie die folgenden Parameter und Daten:

Aktivieren (Checkbox)	Der Benutzer wird hier erstmal aktiviert. Standard ist deaktiviert	
Benutzername	Dieser String wird für den Zugriff per API gebraucht	kopieren Sie ihn von hier heraus
Kennwort	Dieser String wird für den Zugriff per API gebraucht	kopieren Sie ihn von hier heraus
Installations-ID	Dieser String wird für den Zugriff per API gebraucht	kopieren Sie ihn von hier heraus
Rechte	Hier können die Standard-Rechte für das Lesen und Schreiben der Werte der VISION-Elemente definiert werden d.h. das was hier definiert wird, gilt für alle bestehenden und zukünftig angelegten Elemente. Falls Sie nur einzelne Elemente freigeben möchten, sollten Sie die Standard-Rechte hier nicht ändern.	

2.2 BERECHTIGUNGEN FÜR EINZELNE ELEMENTE

Es wird empfohlen, den API-Zugriff nicht auf das gesamte Projekt, sondern nur auf die gewünschten Elemente zu gewähren. Gehen Sie wie folgt vor:

1. Melden Sie sich bei VISION als Administrator an.
2. Wählen Sie das gewünschte Element aus und öffnen Sie dessen Einstellungsmenü (rechte Maustaste oder gedrückt halten, dann Einstellungen)
3. Aktivieren Sie unter dem Menüpunkt Allgemein - Berechtigungen die Option "Standardberechtigungen überschreiben" und gehen Sie dann zum Unterpunkt Berechtigungen, der die Berechtigungsmatrix anzeigt



4. Aktivieren Sie hier die Bedienungsberechtigung, die auch direkt die *Anzeige*-Berechtigung aktiviert. Wenn Sie über den API-Zugang nur Daten lesen wollen, genügt es, das Anzeigerecht zu aktivieren.

5. Wiederholen Sie das gleiche Verfahren für alle Elemente, auf die Sie zugreifen möchten.

3 Verbindung über MQTT

3.1 EINLEITUNG

Als Beispiel wird der Zugriff über die MQTT API des DIVUS KNX IQ mit einer relativ einfachen, kostenlosen Software namens MQTT Explorer (siehe Kap. 1.1) demonstriert, die für Windows, Mac und Linux verfügbar ist. Grundlegende Kenntnisse und Erfahrungen mit MQTT werden vorausgesetzt

3.2 FÜR DIE VERBINDUNG ERFORDERLICHE DATEN

Wie bereits erwähnt (siehe Abschnitt 2.1), sind der Benutzername und das Passwort des API-Benutzers erforderlich.

Im Folgenden finden Sie eine Übersicht über alle Daten, die vor dem Aufbau einer Verbindung erfasst werden müssen:

Benutzername	Auszulesen auf der Detailseite des API-Benutzers
Passwort	Auszulesen auf der Detailseite des API-Benutzers
IP-Adresse	Auslesen in den Launcher-Einstellungen unter Allgemein - Netzwerk - Ethernet (oder über Synchronizer)
Port	8884 (dieser Port ist für diesen Zweck reserviert)

3.3 ERSTE VERBINDUNG MIT DEM MQTT EXPLORER UND ALLGEMEINE ANMELDUNG

Normalerweise unterscheidet MQTT zwischen den Aktivitäten subscribe und publish. MQTT Explorer vereinfacht dies, indem er bei der ersten Verbindung automatisch alle verfügbaren Topics (Topic #) abonniert. Dadurch ist nach einer erfolgreichen Verbindung direkt im linken Bereich des MQTT-Explorer-Fensters der Baum zu sehen, der zu allen verfügbaren Elementen führt (d.h. der API-Benutzer hat Zugriff). Um weitere Subscribe-Topics einzutragen oder das # mit einem spezifischeren Topic zu ersetzen, geht man im Verbindungsfenster auf Erweitert.

Das rechts oben angezeigte Topic sieht etwa so aus:

```
local/api/v1/7f4x0607849x444xxx256573x3x9x983/status/elements/objects_list
```

wo 7f4x0607849x444xxx256573x3x9x983 der API-Benutzername ist und objects_list enthält alle verfügbaren Elemente.

Dieses Topic wird immer auf dem neuesten Stand gehalten, d.h. alle Wertänderungen werden dort in Echtzeit wiedergegeben. Wenn Sie nur einzelne Elemente abonnieren möchten, geben Sie hinter objects_list/ die Element-ID des gewünschten Elements ein.

Hinweis: Diese Art des Abonnements entspricht in etwa der Logik hinter den KNX-Rückmeldeadressen; sie zeigt den aktuellen Status der Elemente an und kann zur Überprüfung verwendet werden, ob die gewünschten Änderungen erfolgreich durchgeführt wurden. Wenn Sie Daten nur auslesen, aber nicht ändern wollen, ist diese Art des Abonnements ausreichend.

Ein einzelnes einfaches Element sieht in JSON-Notation etwa so aus:


```
{
  "data": {
    "functions": {
      "onoff": {
        "value": "1"
      }
    }
  },
  "name": "Licht Küche Ein/Aus",
  "type": 0
  }
}
```


Hinweis: Alle Werte haben die oben gezeigte Syntax z.B. `{ "value": "1" }` als Ausgabe der Subscribe-Topics, während man in die Payload zur Änderung eines Werts d.h. bei Publish-Topics direkt den Wert schreibt die Klammern und „value" fallen aus z.B. `"onoff": "1"`.

4 Erweiterte Befehle

4.1 EINLEITUNG

Im Allgemeinen gibt es 3 Arten von Topics:

1. Topic(s) zum Abonnieren, um die verfügbaren Elemente zu sehen und um Wertänderungen in Echtzeit zu erhalten

2. Topic(s) zum Abonnieren, um die Antworten auf die Veröffentlichungsanfragen (des Kunden) zu erhalten

3. Publish-Topic(s), um Elemente mit ihren Werten zu erhalten oder zu setzen

Wir werden uns später auf diese Arten beziehen, indem wir die hier gezeigte Nummerierung verwenden (z.B. Topics vom Typ 1, 2, 3). Weitere Details in den folgenden Abschnitten und in Kap. **Error! Reference source not found.**.

4.1.1 TOPICS ABONNIEREN, UM DIE VERFÜGBAREN ELEMENTE ZU SEHEN UND WERTÄNDERUNGEN IN ECHTZEIT ZU ERHALTEN

Diese wurden bereits in Kap. **Error! Reference source not found.** beschrieben.

4.1.2 TOPICS ABONNIEREN, UM DIE ANTWORTEN AUF DIE PUBLISH-ANFRAGEN DES CLIENTS ZU ERHALTEN

Diese Art von Topics ist optional. Sie erlaubt es .

- . einen eindeutigen Kommunikationskanal mit dem MQTT-Server zu öffnen, indem eine beliebige client_id verwendet wird. Mehr dazu in Kap. **Error! Reference source not found.**

- . das Ergebnis von Publish-Anfragen für das entsprechende Topic zu empfangen: Erfolg oder Misserfolg mit Fehlercode und Meldung.

Es gibt verschiedene Topics, um Antworten auf Veröffentlichungsbefehle zu erhalten oder zu setzen. Der entsprechende Unterschied in den Pfaden ist der Teil "status/get" gegenüber "status/set". Mehr dazu in Kap. **Error! Reference source not found.**.

Sobald Sie die benötigten Topics für Ihr System gefunden haben, können Sie diesen Schritt überspringen und direkt die Topics veröffentlichen.

4.1.3 TOPICS VERÖFFENTLICHEN, UM ELEMENTE MIT IHREN WERTEN ZU ERHALTEN ODER ZU SETZEN

Für diese Topics wird ein ähnlicher Pfad wie für das Abonnieren verwendet - die einzige Änderung ist das Wort "request" anstelle des für das Abonnieren verwendeten "status". Die vollständigen Topicspfade werden später in Kap. **Error! Reference source not found.** gezeigt.

Ein Get-Topic fordert zum Lesen der Elemente und Werte des MQTT-Servers auf. Die Nutzlast kann verwendet werden, um auf der Grundlage des Funktionstyps der Elemente zu filtern.

Ein Set-Topic fordert dazu auf, einige Teile eines Elements zu ändern, wie in seiner Nutzlast angegeben.

4.2 PRÄFIX FÜR BEFEHLE UND ENTSPRECHENDE ANTWORTEN

4.2.1 KURZE ERLÄUTERUNG

Alle Befehle, die an den MQTT-Server gesendet werden, haben einen gemeinsamen Anfangsteil, nämlich:

```
local/api/v1/[API username]
```

die wir als allgemeines Präfix bezeichnen können.

Abonnieren Sie Topics, um die Antworten auf unsere Wertänderungsanfragen zu erhalten, die alle mit

```
[general prefix]/[client_id]/status/
```

gefolgt von get oder set, je nach Zweck (abfragen oder setzen).

```
[general prefix]/[client_id]/status/get

[general prefix]/[client_id]/status/set
```

Die Publish-Topics beginnen alle mit

```
[general prefix]/[client_id]/request/
```

und wieder, gefolgt von get oder set.

```
[general prefix]/[client_id]/request/get

[general prefix]/[client_id]/request/set
```

4.2.2 AUSFÜHRLICHE ERLÄUTERUNG

Die Echtzeit-Topics (Typ 1) haben das allgemeine Präfix (siehe oben), gefolgt von

```
/status/elements/objects_list
```

und objects_list, die, wie der Name schon sagt, die verfügbaren Elemente enthält, kann von einem # gefolgt werden, um sie alle anzuzeigen, oder von einer Element-ID, um nur ein bestimmtes Element anzuzeigen. z.B.:

```
local/api/v1/[API username]/status/elements/objects_list/#          //zeigt alle
```

oder

```
local/api/v1/[API username]/status/elements/objects_list/140          //zeigt nur das Element mit ID 140
```

Mit ein paar Änderungen, ausgehend von den obigen Topics, erhalten wir diejenigen vom Typ 2, um unseren eigenen Kanal zu abonnieren, der durch eine client_id unserer Wahl gekennzeichnet ist. Wir fügen also die client_id nach dem API-Benutzernamen ein

```
local/api/v1/[API username]/[client_id]/
```

und je nach der Art der Antworten, die wir erhalten möchten, verwenden wir get oder set nach dem Schlüsselwort status, z. B.

```
local/api/v1/[API username]/[client_id]/status/get/elements/objects_list/140
```

oder

```
local/api/v1/[API username]/[client_id]/status/set/elements/objects_list/140
```

Hinweis: diese Topics sind, wie zuerst erwähnt, im Fall vom MQTT Explorer unter *Erweitert* einzutragen.

Hinweis: diese Topics geben nichts aus, bis wir anfangen, darin etwas zu veröffentlichen.

Für Topics des Typs 3 zum Abrufen oder Setzen von Werten für die Elemente von KNX IQ müssen Sie nur Status durch Anforderung in den Topics des Typs 2 ersetzen, z. B.:

```
local/api/v1/[API username]/[client_id]/request/get/elements/objects_list/140
```

oder

```
local/api/v1/[API username]/[client_id]/request/set/elements/objects_list/140
```

Bei Set-Befehlen spielt natürlich die Payload die Hauptrolle, da sie die gewünschten Änderungen (d.h. geänderte Werte für die Funktionen des Elements) enthält.

Warnung: Verwenden Sie niemals die Option retain in Ihren Befehlen des Typs 3, da dies zu Problemen auf der KNX-Seite führen kann.

4.3 BEISPIEL: VERÖFFENTLICHUNG ZUM ÄNDERN EINES WERTS EINES EINZELNEN ELEMENT

Der einfachste Fall ist, dass Sie den Wert eines der im allgemeinen Abonnement angezeigten Elemente ändern möchten (siehe vorheriges Kapitel).

Im Allgemeinen besteht die Prozedur zum Ändern/ Schalten einer Funktion von VISION über MQTT aus 3 Schritten, von denen nicht alle unbedingt notwendig sind, aber wir empfehlen dennoch, sie wie beschrieben auszuführen.

1. Das Topic, das die zu ändernde Funktion enthält, wird mit einer eigenen client_id abonniert (subscribe)

2. Das zu bearbeitende Topic wird zusammen mit der Payload mit den gewünschten Änderungen unter Verwendung der in 1. gewählten client_id veröffentlicht (publish).

3. Zur Kontrolle kann man dann die Antwort im Topic (1.) sehen - d.h. ob (2.) funktioniert hat oder nicht

4. Im allgemeinen Subscribe, in dem alle Werte bei Änderungen aktualisiert werden, können Sie die gewünschte(n) Wertänderung(en) sehen, wenn alles gut geklappt hat.

Die Schritte dazu sind:

1. Wählen Sie ein client_id, z. B. "Divus", und fügen Sie es in den Pfad nach dem API-Benutzernamen ein.

```
local/api/v1/7f4x0607849x444xxx256573x3x9x983/Divus/status/set/elements/objects_list/41
```

   Dies ist das vollständige Topic für das Abonnieren Ihres eigenen Kommunikationskanals mit dem MQTT-Server. Damit teilen Sie dem Server mit, wo Sie die Antworten auf die Änderungen erwarten, die Sie zu senden beabsichtigen. Beachten Sie den Status/Set-Teil, der a. definiert, dass es sich um ein Subscribe-Topic handelt und b. dass es die Antworten auf Set-Befehle erhält.

2. Das Publish-Topic ist dasselbe, außer dass die Status/Request-Schlüsselwörter ausgetauscht werden

```
local/api/v1/7f4x0607849x444xxx256573x3x9x983/Divus/request/set/elements/objects_list/41
```

3. Worin die Änderung bestehen soll, steht in der Nutzlast. Hier sind einige Beispiele.

   - Ausschalten eines Elements, das die Funktion Ein/Aus hat (1 Bit):

```
{
   "data": { "functions": { "onoff": 0}
}
```

   - Einschalten eines Elements, das die Funktion Ein/Aus hat (1 Bit). Wenn mehrere solcher Befehle vom selben Client gestartet werden, kann außerdem der uuid-Parameter ("unique ID", ist in der Regel ein 128-Bit-String, formatiert als Hex-Wert mit 8-4-4-4-12 Ziffern) verwendet werden, um die Antwort der entsprechenden Abfrage zuzuordnen, da dieser Parameter - falls in der Abfrage vorhanden - auch in der Antwort zu finden ist.

```
{
   "uuid": "550e8400-e29b-41d4-a716-446655440000",
   "data": { "functions": { "onoff": 1}
}
```

   - Einschalten und Einstellen der Helligkeit eines Dimmers auf 50%

```
{
   "uuid": "550e8400-e29b-41d4-a716-446655440001", // optional
   "data": { "functions": { "onoff": 1, "dimming": "50" }
}
```

     Die Antwort auf das oben gezeigte und abonnierte Topic (genauer gesagt, seine Nutzlast) lautet dann zum Beispiel.

```
{
   "uuid": "550e8400-e29b-41d4-a716-446655440001", // nur wenn im subscribe gesetzt
   "data": {
     "functions": {
```

```
          "onoff": "202",
          "dimming": "404"
      }
    }
}
```

Die obige Antwort ist ein Beispiel für den Fall einer korrekten Nutzlast, obwohl das Element keine Dimmfunktion hat.

Wenn es schwerwiegendere Probleme gibt, die dazu führen, dass die Nutzlast nicht korrekt interpretiert werden kann, sieht die Antwort wie folgt aus (z. B.):

```
 {
    "uuid": "550e8400-e29b-41d4-a716-446655440001", // only if set in subscribe
    "error": "400",
    "message": "BadRequest, Non-compliant topic"
}
```

Siehe Kap. 5.1 für eine Erleuterung der Fehler-Codes und -Meldungen aber im Allgemeinen sind, wie bei http, 200 Codes positive Antworten, während 400 negative sind.

4.4 BEISPIEL: VERÖFFENTLICHUNG ZUR ÄNDERUNG DER WERTE MEHRERER ELEMENTE

Das Verfahren ist ähnlich wie das zuvor gezeigte, um ein einzelnes Element zu ändern. Der Unterschied besteht darin, dass Sie die element_id in den Topics weglassen und dann den Satz der element_ids vor den Daten in der Payload angeben. Siehe die Syntax und Struktur unten.

1. wie zuvor beginnen Sie mit dem Abonnieren des eingestellten Themas unter Verwendung einer benutzerdefinierten client_id (hier "Divus")

```
local/api/v1/7f4x0607849x444xxx256573x3x9x983/Divus/status/set/elements/objects_list
```

2. die Bearbeitung erfolgt dann in dem entsprechenden Publish-Topics

```
local/api/v1/7f4x0607849x444xxx256573x3x9x983/Divus/request/set/elements/objects_list
```

mit einer wie folgt strukturierten Nutzlast:

```
{
   "uuid":"request_uuid",
   "data": {
      "8": {
               "functions" {
                  "onoff":"1",
                  "dimming":"50"
               }
          },
      "9": {
               "functions" {
                  "onoff":"1"
```

```
                }
            }
    }
}
```

4.5 FILTER NACH FUNKTIONSTYP IN ABFRAGEN

Mit dem Parameter filters in der Nutzlast können nur die gewünschte(n) Funktion(en) eines Elements angesprochen werden. Die Ein/Aus-Funktion eines Schalters oder Dimmers wird z.B. "onoff" genannt und der entsprechende Filter wird folgendermaßen definiert:

```
{

   "filters": { "functions": ["onoff"]}

}
```

Die Antwort sieht dann z. B. so aus

```
{
   "data": {
       "functions": {
         "onoff": {
            "value": "1"
         }
       },
       "name": "Dimmer FL",
       "type": 1
    }
}
```

Die eckige Klammer zeigt an, dass Sie auch nach mehreren Funktionen filtern können, z. B.

```
{

   "filters": { "functions": ["temperature", "humidity"]}

}
```

führt zu einer Antwort wie dieser:

```
{
  "data": {
    "functions": {
      "humidity": {
        "value": "58.6"
      },
      "temperature": {
        "value": "19.87"
```

```
      }
    },
    "name": "Thermostat",
    "type": 5
  }
}
```

5 Anhang

5.1 FEHLER-CODES

Fehler in der MQTT-Kommunikation führen zu einem numerischen Code. Die folgende Tabelle hilft bei der Aufschlüsselung.

Aktion	Fehler-Code	Bedeutung
Publish	202	Accepted (function processed) - N.B. das ist nicht ein Fehler-Code also OK
Abfrage	400	Incorrect query (der Inhalt der Abfrage enthält einen Fehler) Non-compliant payload (Fehler in der Nutzlast)
Abfrage	401	Not authorised (Probleme mit den Rechten des API-Benutzers)
Publish	404	Not found (bezieht sich auf Element-ID oder Funktion)
Abfrage	405	Topic not valid (Fehler im Topic/Pfad)
Abfrage	409	Conflict (Fehler beim Filtertyp)

5.2 PARAMETER DER NUTZLAST

Die Payload unterstützt je nach Kontext unterschiedliche Parameter. Die folgende Tabelle zeigt, welche Parameter in welchen Topics vorkommen können.

Topic-Typ	Parameter	Unter-Parameter	Erklärung

Abfrage	`data`		Enthält die verfügbaren Funktionen mit den jeweiligen Werten
			`{`
			`"data": {`
			`"functions": {`
			`"cooling_onoff": {`
			`"value": "0"},`
			`…`
			`}`
			`}`
			`}`

		functions	Enthält eine oder mehrere Funktionen mit dem jeweiligen Wert. Welche Funktionen zur Verfügung stehen, hängt vom Elementtyp und seiner Konfiguration ab. Sie können dies in den Subscribe-Topics sehen. `{ "functions": { "cooling_onoff": { "value":"0" }, "temperature": { "value":"23.48" } } }`
	`name`		Name des Elements, wie in VISION definiert `"name": "Thermostat"`
	`type`		Numerischer Wert, der den Elementtyp charakterisiert, z. B. Ein/Aus-Elemente haben den Typ 0, Dimmer-Elemente den Typ 1, Thermostat-Elemente den Typ 5 usw.. `"type": 5`
subscribe or request	`uuid`		Optionaler Parameter, um die zukünftige Antwort der Anfrage zuordnen zu können (da die gleiche ID in der Antwort zu finden ist). Spielt eine Rolle, wenn viele Abfragen gleichzeitig vom Client zum Server losgehen können und die Gefahr besteht, dass die Antworten nicht den Abfragen zugeordnet werden können. `{ "uuid": "550e8400-e29b-41d4-a716-446655440001" }`
	`filters`		Optionaler Parameter zur Filterung (derzeit nur nach Funktionstyp), wenn Sie nur bestimmte Funktionen direkt ansprechen wollen. Es können mehrere Funktionen zur Filterung verwendet werden, die durch ein Komma getrennt werden. Der gesamte Filter wird dann gelesen als "gebe nur die Funktionen und ihre Werte zurück, die ich hier aufliste (von einem oder mehreren Elementen, je nach Topic). `{ "filters": { "functions": ["onoff"]} }`

5.3 VERSIONSHINWEISE

5.3.1 VERSION 1.00

Neu:

- Erste Veröffentlichung

NOTIZEN