

Kurzanleitung Scriptdeveloper V3.05.06 (15. Dezember 2018)

Der Scriptdeveloper (SDV) soll ein Hilfsmittel im Alltag bei der Erstellung von Homematic Scripten und deren Tests darstellen. Ein gewisses Wissen über Scripterstellung sowie den Aufbau einer CCU wird vorausgesetzt.

Die Software läuft auf Windows PC, ist bei nicht kommerzieller Nutzung Freeware und ist nicht an die Nichtverwendung einer Raspberrymatic oder sonstiger Einschränkungen gebunden.

Da mittlerweile aber schon einige tiefgreifende Operationen möglich sind, sind Löschfunktionen erst nach Drücken von Unlock  zugänglich.

Trotzdem an der Stelle der Hinweis, welcher auch beim ersten Start des Programmes bestätigt werden muss:

Dies ist eine BetaTestversion.

Die Verwendung dieser Software erfolgt auf eigenes Risiko
Der Autor dieser Software übernimmt keine Haftung für direkte oder indirekte Schäden, welche sich aus der Benutzung dieser Software ergeben sollten.
Eine kommerzielle Nutzung dieser Software ist untersagt

Ich bin einverstanden (Ja, Nein, wobei nein zum Programmende führt)

Hinweise über undokumentierte Methoden, die im Alltag nützlich sind aber ich bis jetzt auch noch nicht kannte, nehme ich gerne an und baue die auch gerne hier in das Programm mit ein.

Inhalt

Kurzanleitung Scriptdeveloper V3.05.06 (15. Dezember 2018)	1
1. Installation	3
1.1 Lizensierung	4
1.2 Systemvoraussetzungen	8
1.3 Was tut's bis jetzt	8
1.4 Bekannte Einschränkungen / Bugs	8
1.6 Changelog	9
1.6.1 Changelog 03.05.06 LZL	9
1.6.2 Changelog 03.05.01 LZL	9
1.6.3 Changelog 03.04.01 LZL	9
1.6.4 Changelog 03.03.01 LZL	9
2 Oberfläche	10
3 Scripteditor	11
3.1 Voreinstellungen Editor	12
3.2 Vervollständigen Funktion	13
4 Inspektor	14
4.1 Selektionswahl: DomScan	15
4.2 Selektionskriterium Types	18
4.3 Zusätzliche Selektionsbedingungen	19
4.4 Daten aus Inspektor in Editor übernehmen	25
4.4.1 Mehrfachauswahl als Enum String	27
4.5 Selektion von Selektion	29
4.6 Objekte löschen	30
4.7 Anwenderdefinierte Sichten	32
5 Backups	33
5.1 Räume	33
5.2 Gewerke	33
5.3 Systemvariablen	34
5.4 Devices und Kanäle	34
6 Kleine Helfer im Alltag	35
6.1 Umbenennen von Kanälen von Geräten	35

1. Installation

Das *.rar File in ein beliebiges Verzeichnis entpacken. Ein Installer ist nicht notwendig. In diesem Verzeichnis befindet sich auch das Konfigurationsfile SDV.INI. Bei der erstmaligen Verwendung muss dieses angepasst werden

```
[LAST]
DATEI=c:\MTH\Homematic\NewScript.hsc

[HOST]
CCU=192.168.2.19
NICKNAME=Benutzername           ← anpassen Benutzername / Nick
CCU1=192.168.2.XX              ← IP Adresse der 1. CCU
CCU2=192.168.2.XX              ← Wenn vorhanden, IP der 2. CCU
CUXD=CUxD.CUX2801001:5         ← CUxd KANAL (ist nötig, dafür braucht es kein pscp mehr)
LICENCE1=                        ← Lizenzschlüssel für 1. CCU
LICENCE2=                        ← Lizenzschlüssel für 2. CCU

[ENUM_NORM]                      ← Ab hier kommen dann interne Werte, Finger Weg
C1=65
C2=200
C3=293
C4=65

[ENUM_MAX]
C1=65
C2=200
C3=293
C4=65
```

BestandsNutzer:
Die Ini Datei hat sich ziemlich vergrößert. Hilfreich ist, in der mitgelieferten INI alle Schlüssel ab
[CFG_ChanView1_Methods]
Address=True....
Und in die bestehende INI Datei anzuhängen. Erspart eine Menge manueller Hakensetzen.

Warum CUxD ? Der SDV Version 2.x nutzte noch pscp für den Zugriff auf die Logdatei und auf das System. Dies war immer ein Schwachpunkt (zusätzliches Programm, Bestätigung Serverzertifikat. Dies wird jetzt mit CUxD realisiert. Es muss ein Kanal angegeben werden auf einem CUxD exec Gerät, auf das der SDV exclusiven Zugriff hat. Auf Systemen ohne CUxd kann der SDV nicht eingesetzt werden.

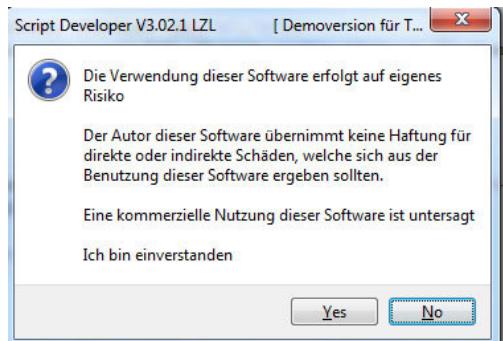
1.1 Lizenzyierung

Der SDV ist bei nicht kommerzieller Nutzung Freeware. Trotzdem habe ich mich entschlossen, aufgrund von Erfahrungen der Vergangenheit den Nutzerkreis oder die möglichen Features bestimmter Nutzer einzuschränken. Dies geschieht durch Vergabe von bis zu 2 Lizenzschlüsseln. Der SDV ist dadurch an bis zu 2 CCU / Raspberrymatic gepaart.

Wie arbeitet das ?

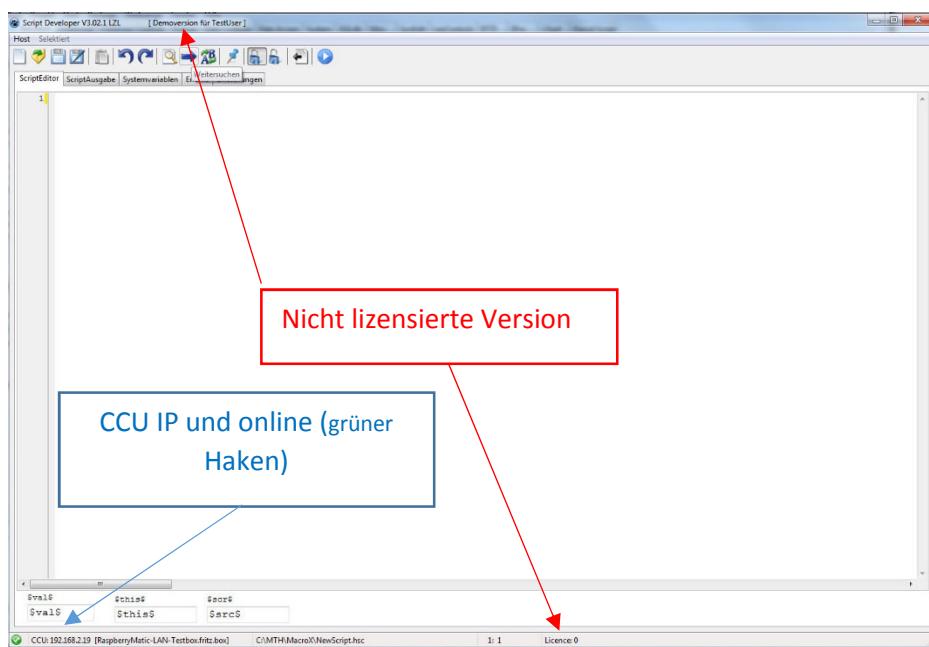
Der SDV telefoniert nicht nach Hause. Um eine Lizenz anzufragen ist folgender Weg einzuschlagen.

1. Die Konfigurationsdatei SDV.INI mit einem Editor öffnen.
2. Nickname Anpassen
3. IP der CCU 1 eintragen
4. IP der CCU 2 eintragen
5. CUXD Kanal eintragen
6. Konfiguration abspeichern
7. Script Developer starten

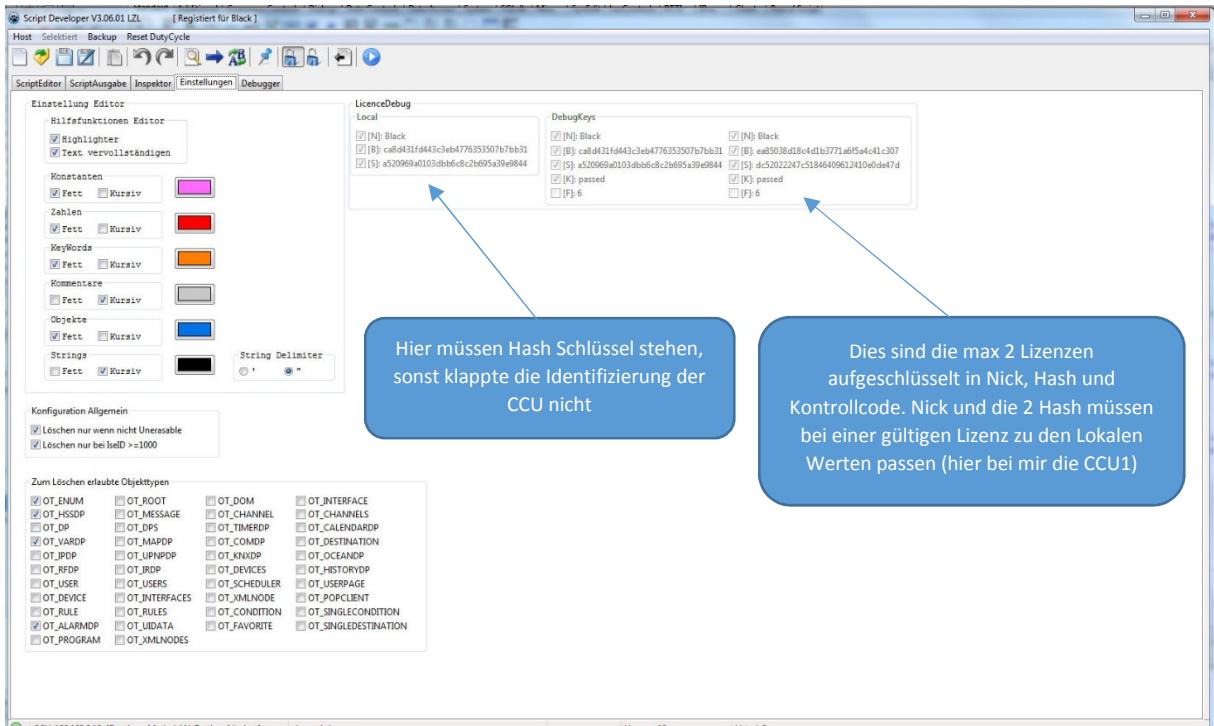


Bei allerersten Start muss dieses Fenster mit yes bestätigt werden. No führt so einem sofortigen Programmabbruch

Bei Bestätigung mit Yes startet nun zum erstenmal der SDV als Demoversion



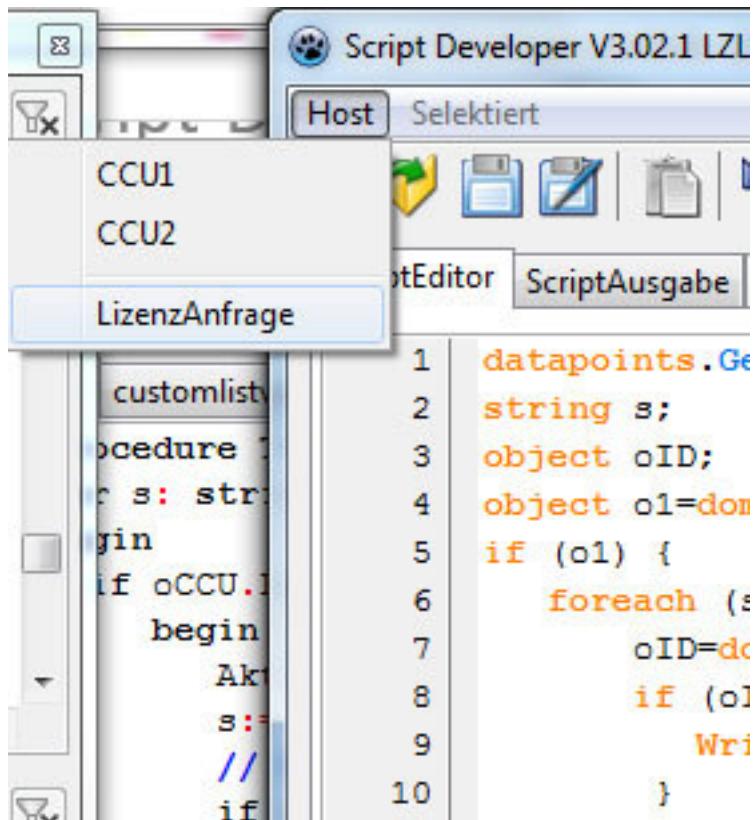
Wenn die CCU, für die der Schlüssel angefragt werden soll, als grün angezeigt wird, bitte vorher einmal unter dem Reiter Einstellungen kontrollieren



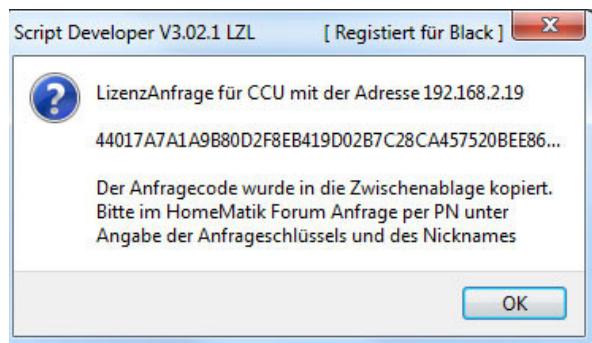
Aus einer Anfrage ohne unter local sinnige Einträge zu sehen lässt sich kein gültiger key generieren.

Aufgrund einer Programmungsgenauigkeit bei der Ausformulierung einer RegEx ist es leider sehr wahrscheinlich, dass alte Lizenzschlüssel vor 3.5.2 als ungültig erkannt werden. Sorry dafür, in dem Fall bitte die Anfrageschlüssel neu erstellen (wenn die Hashwerte gültig sind) und mir schicken. Der SDV 3.5.0 ist davon nicht betroffen, also am besten den 3.5.2 in ein neues Verzeichnis entpacken, die SDV.INI von dem 3.5.0 kopieren ins 3.5.2 Verzeichnis und neue Anfrage machen unter dem 3.5.2, da die alte INI nicht verändert wurde im alten Verzeichnis kann solange dann noch mit der 3.5.0 Version gearbeitet werden.

Für die weiteren Schritte muss der SDV mit der CCU verbunden sein und die CCU auch als online erkannt worden sein.



unter Host auf Lizenzanfrage drücken. Als nächstes öffnet sich ein Fenster mit einem Anfrage Hexstring.



Der Hexstring ist in die Zwischenanlage kopiert und kann in beliebige Text Dokumente eingefügt werden. Als nächste dann im Homematik.de Forum eine PN an mich schreiben mit dem String und Angabe des Nicknames, welcher zum Zeitpunkt der Lizenzanfrage in der INI Datei eingetragen war.

Was enthält dieser Hexstring ?

Kodiert und verschlüsselt: 1. den Nicknamen, 2. die Seriennummer des Funkmodules der verbundenen CCU , einen VerifizierCode von mir.

Die Seriennummer des Funkmodules ist nötig zur Verifizierung des Pairings. Diese wird bei mir nirgends gespeichert, mit diesem Hexschlüssel wird nach der Anfrage der LizenzLevel definiert und ebenfalls in einen Hexstring kopiert. Dieser dann zurückgesandte Hexstring wird unter Licence1 oder Licence2 in der INI Datei eingetragen. Es kann mit bis zu 2 CCU bearbeitet werden, sollte ein Lizenzlevel höherwertiger sein so gilt dieser höherwertige Level für beide CCUs.

Wer mit diesem Verfahren nicht einverstanden ist, möge bitte an dieser Stelle die PDF Datei schließen und kann die Dateien beruhigt löschen.

Geplant hab ich folgende Lizenzabstufungen

Level	Editor	Script ausführen	Highlighter Und Vervollständiger	Enums	SysVar	Programs	Backup Restore		
0	X								
1	X	X							
2	X	X	X						
3	X	X	X	X	X				
4	X	X	X	X	X	X			
5	X	X	X	X	X	X	X		
6									

1.2 Systemvoraussetzungen

Der SDV lief bisher in Testinstallionen unter WIN 7 64/32 bit, und unter Win 10 64bit. Da unter recht konservativen Compilereinstellungen übersetzt wurde, sollte er eigentlich unter allen Windows Version laufen (ab Win 7)

Auf der Homematic-Seite wurde bei mir auf einer Raspberrymatic 3.37.8.20181026 und 3.41.11.20181126 getestet. Allerdings ist noch nicht der Authorisierungsmechanismus über Nutzername/Passwort implementiert. Der SDV braucht Zugriff über Port :8181

Auf einer CCU sind die erzeugen internen Progs auch lauffähig, wenn Rega-Community eingestellt wird. Unter Legacy läuft es NICHT !

1.3 Was tuts bis jetzt

Der Editor funktioniert inkl. Suchen und Suchen / ersetzen. Der Highlighter und der Code vervollständiger arbeiten auch.

Undo / Redo arbeiten

Script ausführen arbeitet und liefert wie in der alten Version die antworten der CCU.

Enums und Sysvars arbeiten auch schon inkl Detaildaten und Editermöglichkeiten.

Darstellbarkeit zumindest der Grundmethoden aller Objekte

DomScan

Devices

Aufschlüsseln der MetaDaten

1.4 Bekannte Einschränkungen / Bugs

Auswahldialoge sind auf Englisch. Weiß ich, zurzeit benutze ich die in der Laufzeitumgebung integrierten Dialoge, und die sind leider trotz Landeseinstellung englisch.

Folding im Editor arbeitet noch nicht. Wenn der Rest läuft gucke ich da mal nach.

Kommentare im Script müssen als !- geschrieben werden. Kann man sich dran gewöhnen, das anzupassen wäre ein Haufen Aufwand, da EQ3 ja klugerweise Negation und Kommentar mit demselben Zeichen bedacht hat. Hurra. Ich kann jedenfalls mit dem !- gut leben, folglich ist die Chance, das ich das ändere, recht gering: xD

Aufgrund dessen, dass als Middleware bei mir IOBroker läuft und ich die Diagramm und die History Funktion der CCU nicht nutze, werde ich diese im SDV auch nicht ausprogrammieren.

1.6 Changelog

1.6.1 Changelog 03.05.06 LZL

Multithreading eingeführt für CCU Zugriff, Ping, etc (sollte keine Hänger mehr geben)
TSynHighlighterClass umgeschrieben, im Gegensatz zur originalen Version arbeitet meine nun CaseSensitive wie auch die CCU
Weitere Methoden eingefügt in Highlighter, Autocomplete und Detailansicht.
Detailansicht für Datenpunkte, Alarme, Systemvariablen, Devices und Channels komplettiert
Auf Sonderwunsch unseres Stammtischs Programme schon mal provisorisch mit ProgramCopyID Test eingefügt
Ein paar Standartfilter geschrieben und der Version im Rar beigefügt (*flt Dateien)
Umbenennen von DeviceKanälen nach dem Namen des übergeordneten Devices
Backup und Restore von Devicekanälen
Temporäre Lizenzen nun möglich

1.6.2 Changelog 03.05.01 LZL

Einige Programmfehler beseitigt.
Endlich die Codierung zwischen LZL und dem WebServer richtig in Griff bekommen
Device Objekt in Auswahl hinzu und Detailansicht
Systemvariablen Objekt vervollständigt
Alarm Objekt angelegt und vervollständigt
Metadaten Aufschlüsselung
Backup Methoden für Räume, Gewerke und Systemvariablen

1.6.3 Changelog 03.04.01 LZL

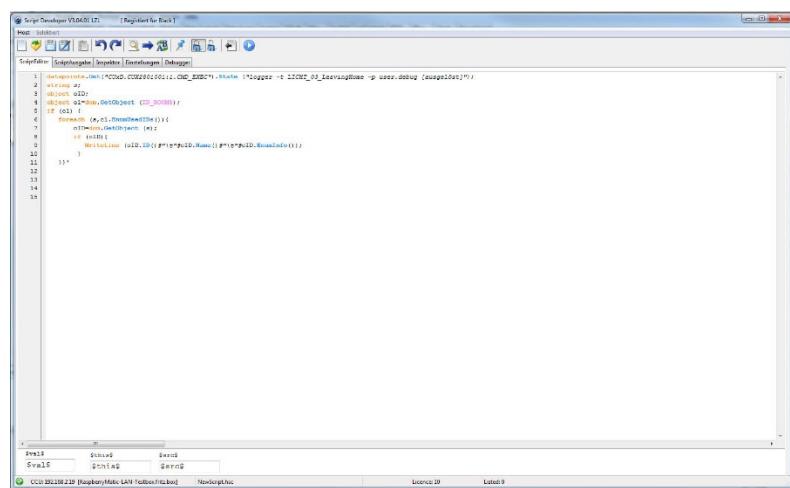
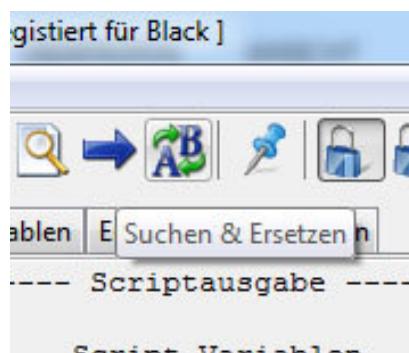
Einige Programmfehler beseitigt.
Variable Fenstergössen im Inspektor
Anzeige zur CCU Nummer die Hinterlegte IP
Scriptnamen Anzeige ohne Pfade (Länge der Anzeige)
Löschen von Objekten
Selektionen aus Selektionen
Sichern und Zurückholen von Selektionsfeldern (PIN)
Merken von Selektionen und Übernahme als Enum in den Scripteditor
Rekursives Auflösen der Detaildarstellung
Einige interne Änderungen, um die nächsten Steps der Roadmap effektiver zu ermöglichen

1.6.4 Changelog 03.03.01 LZL

Listendarstellung Sortieralgorithmus geändert
Kleine Programminkonsistenzen beseitigt.
Feld Objecttyp in Listendarstellung hinzu
Schreibfehle rbei \$src\$ beseitigt.
ObjectSelektion hinzugefügt

2 Oberfläche

Zu fast allen Funktionen sind die Hint parametriert, so dass es da Hilfestellung gibt.



Im Menüreiter **Scripte** finden sich die Einstellungen zum Anlegen eines neuen Scriptes zum Laden eines bestehenden Scriptes und zum Speichern eines Scriptes im Scripteditor sowie speichern unter.

In der Statuszeile finden sich Information über:

1. IP der Host CCU
2. DateiNames des Scriptes im Scripteditor
3. Anzahl der Elemente in der Listendarstellung

3 Scripteditor

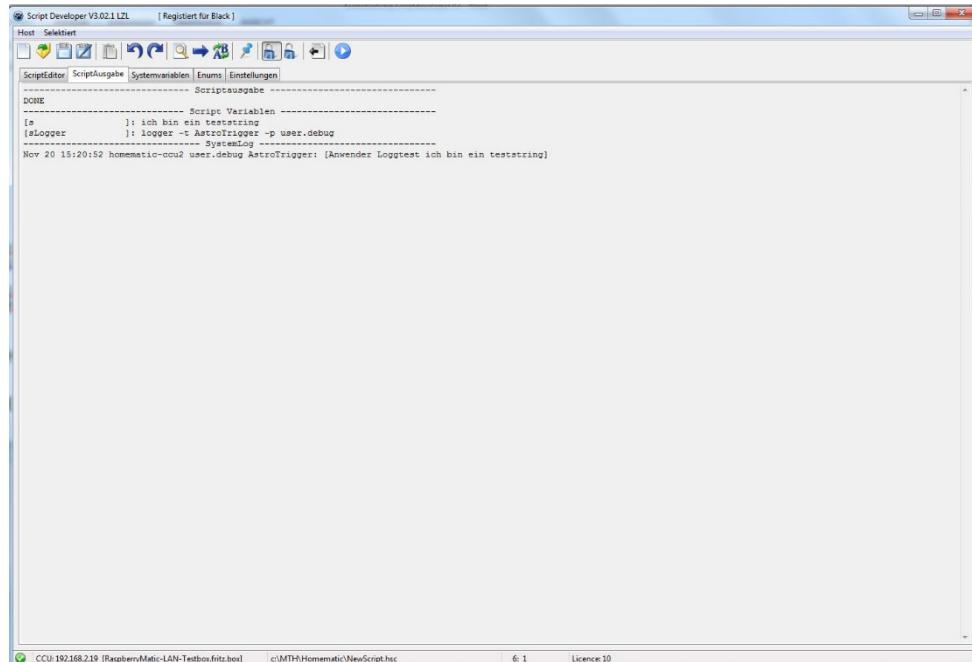
Im Scripteditor werden die Scripte geschrieben oder geladen, die mittels Run Script oder  an die CCU zum Ausführen gesendet werden. Das Scriptergebnis wird dann im Reiter Ausgabe angezeigt. Dieses kleine TestScript zum Beispiel:

```
string s= "ich bin ein teststring";
string sLogger      = "logger -t AstroTrigger -p user.debug ";

datapoints.Get("CUXD.CUX2801001:1.CMD_EXEC").State (sLogger # "[Anwender Loggtest " # s # "]");
WriteLine ("DONE");
```

Hier testweile aus State

Erzeugt folgende Ausgabe:



The screenshot shows the 'Scriptausgabe' tab of the Script Developer software. The output window displays the following text:

```
DONE
----- Script Variablen -----
[s           ]; ich bin ein teststring
[sLogger     ]; logger -t AstroTrigger -p user.debug
----- Systemlog -----
Nov 20 15:20:52 homematic-ccu2 user.debug AstroTrigger: [Anwender Loggtest ich bin ein teststring]
```

Script Ausgabe stellt alles dar, was in dem Script mit Write, WriteLine oder Derivaten zur Ausgabe gebracht wurde,

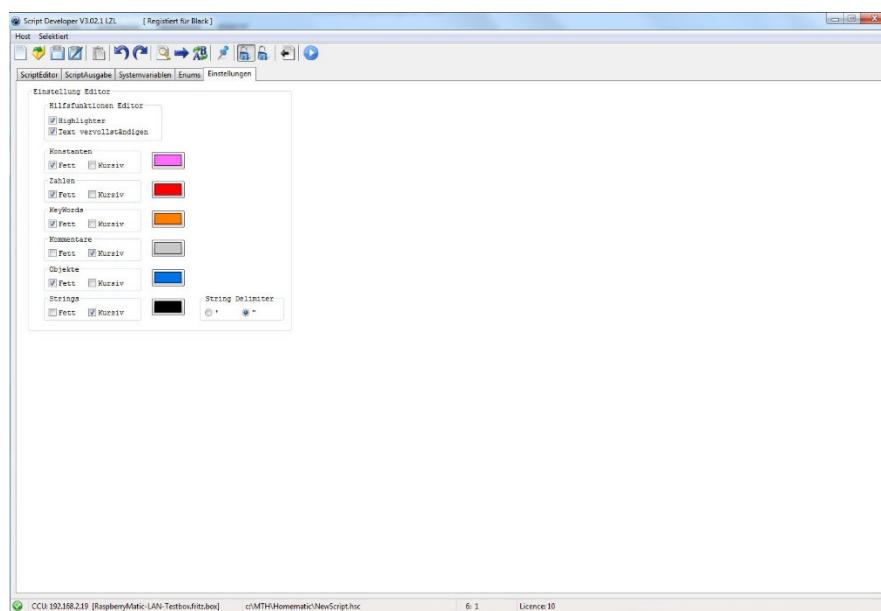
Unter lokale Script variablen stehen die Variablen welche im Script definiert wurden mit ihren Namen. In dem Fall hier sind das die Beiden String Variablen s und sLogger.

Wurde via Userlog ein Eintrag im Logfile erzeugt, so wird dieser nach Scriptende auch hier angezeigt.

Sollte in dem Script ein Fehler sein (hier testweise State zu Stat geändert) erhält man die gleiche Ausgabe wie im Syslog:

```
[----- Fehler im Script -----]
Jun 15 12:41:49 homematic-raspi local0.err ReGaHss: Error: IseESP::SyntaxError= Error 1 at
row 4 col 88 near ^ (sLogger # "[Anwender Logtest " # s # "]");^M WriteLine ("DONE");^M
[iseESP.cpp:1121]
Jun 15 12:41:49 homematic-raspi local0.err ReGaHss: Error: ParseProgram: SyntaxError=
(sLogger # "[Anwender Logtest " # s # "]");^M WriteLine ("DONE"); [iseESP.cpp:374]
```

3.1 Voreinstellungen Editor



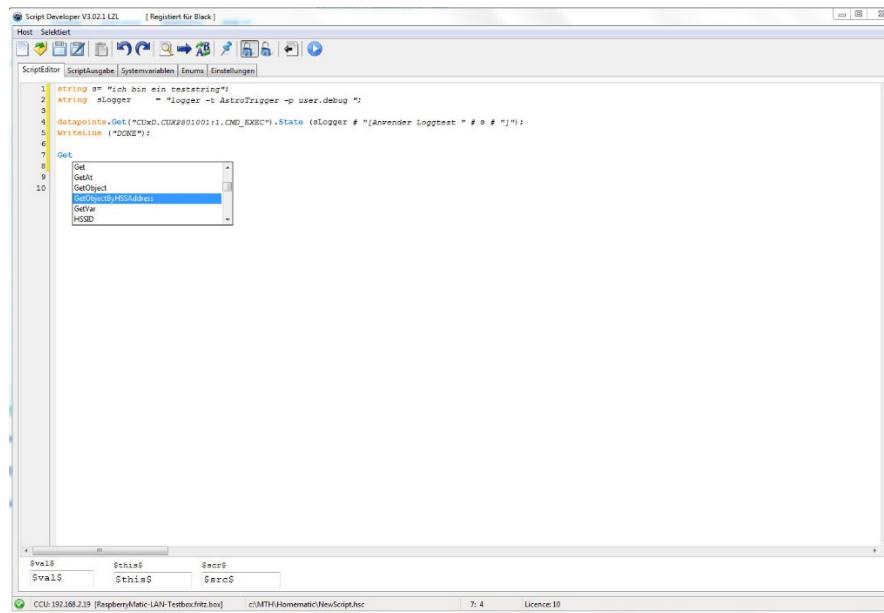
Hier kann nach Vorliebe der Highlighter konfiguriert oder auch ausgeschaltet werden

Ebenso lässt sich die Vervollständigen Funktion an oder abwählen.

Vorbedingung natürlich: Lizenzlevel muss vorhanden sein.

3.2 Vervollständigen Funktion

Methoden und Konstanten Namen muss man sich auswendig merken. Der Editor verfügt über einen Auto Vervollständiger. Man schreibt den Wortanfang, hier z.B Get , drückt Strg+Space und wählt in dem sich öffnenden Menü die passende Funktion aus.

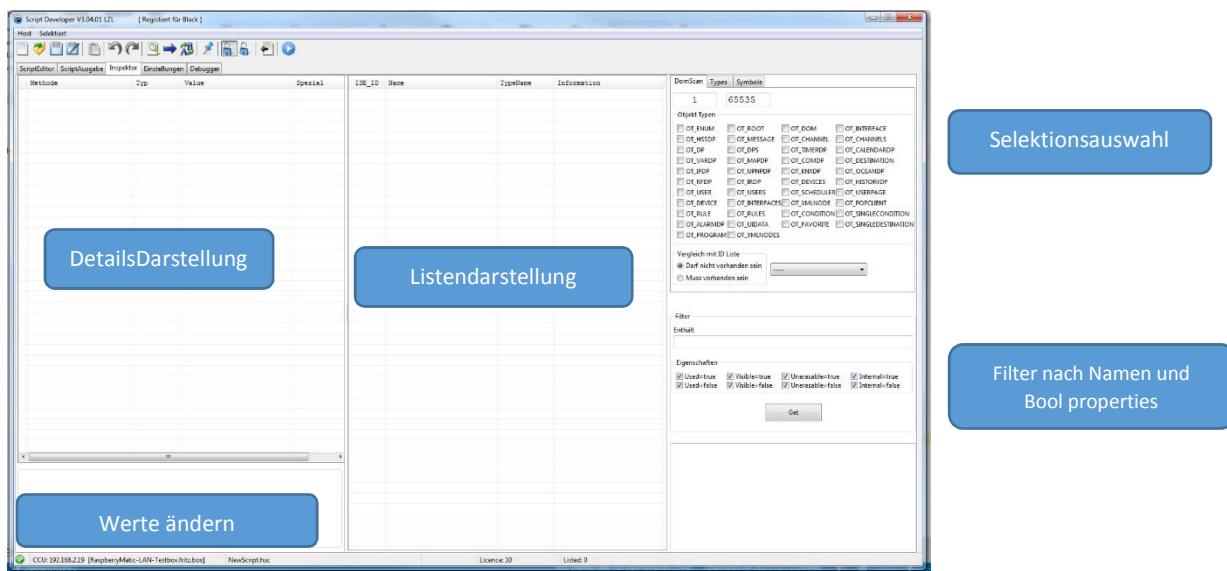


Nach Druck von Enter erscheint das Wort im Editor.

4 Inspektor

Der Inspektor dient zum Suchen, Anzeigen und Ändern von Objekten auf der CCU/Raspberry-Matik.

Es existieren verschiedene Selektionskriterien.



Filteroptionen:

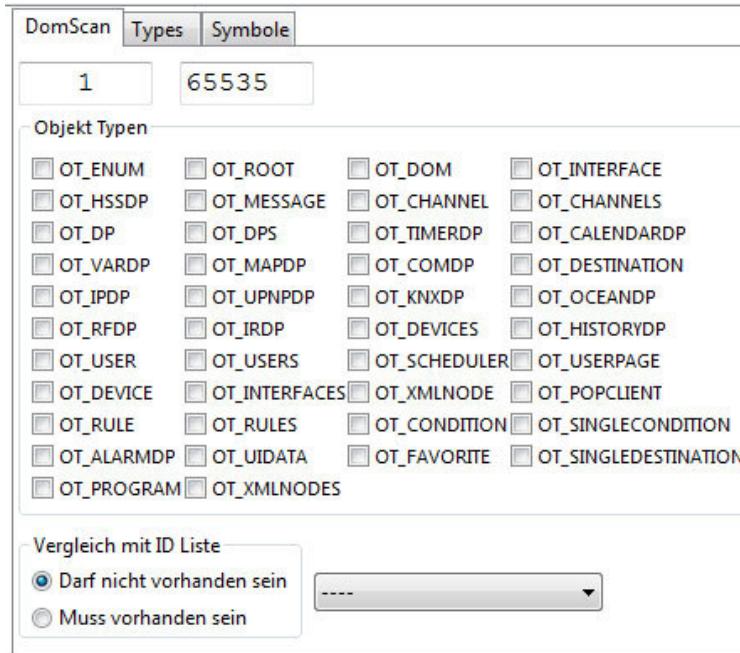
Auswahl der Aufzählungen (Räume, Gewerke, Favoriten, Interfaces , Systemvariablen sind bisher implementiert)

Enthält: leerer Eintrag = es wird nicht nach enthaltener Buchstabensequenz selektiert
Eingegebener Text. Die Systemvariable muss im Namen die Buchstabensequenz enthalten.

Eigenschaften: Es wird nach den Eigenschaften Used, Visible, Unerasable und Internal selektiert.
Am Beispiel used:

1. Kein Haken bei Used= false und kein Haken bei Used= true
Die Eigenschaft Used wird bei der Auswahl nicht beachtet
 2. Haken bei Used= false und Haken bei Used= true
Die Eigenschaft Used wird bei der Auswahl nicht beachtet
 3. Kein Haken bei Used= false und Haken bei Used= true
Um gelistet zu werden muss das Objekt die Eigenschaft Used=true haben
 4. Haken bei Used= false und kein Haken bei Used= true
Um gelistet zu werden muss das Objekt die Eigenschaft Used=false haben

4.1 Selektionswahl: DomScan



Eingabe des Scan Bereiches der IseID's (hier von 1-65535)

Achtung

Schrott Eingabe von Millionenwerten werden die CCU lahmlegen. Der SDV ist schließlich kein Spielzeug sondern ein Werkzeug, man sollte schon wissen, was man tut.

Damit ein Objekt Selektiert wird, muss es die angeklickte Objekteigenschaft haben.

Mehrfachangaben sind möglich

Beispiel für Suchen aller Objekte mit der Eigenschaft OT_DEVICE im Bereich der ISE_Nummern 1-65535

IseID	Name	TypeName	Information
1	Gateway	DEVICE	
1012	IP-RCON-10_BlaBla00-XX	DEVICE	
1239	CCU-ETIME	DEVICE	
1414	CCU-ALARMUSER	DEVICE	
2718	CCU-EXCER	DEVICE	
2844	BB-1C-SwD-PH	CALENDARDP	
3649	Zeitmando	CALENDARDP	
3864	BB-1C-SwD-PH	DEVICE	
4029	CCU-ETIME_COU7070VAR0001	DEVICE	
5971	Zeitmando	CALENDARDP	
6676	Zeitmando	CALENDARDP	
7418	RmTP-BSL_001A5A9A628010	DEVICE	

IseID: 1
 Types: 65535
 Object Type:
 OT_ENUM OT_ROOT OT_DOM OT_INTERFACE
 OT_HSSDP OT_MESSAGE OT_CHANNEL OT_CHANNELS
 OT_DP OT_DPS OT_TIMERDP OT_CALENDARDP
 OT_VARDP OT_MAPDP OT_COMDP OT_DESTINATION
 OT_IPDP OT_UPNPDP OT_KNXDP OT_OCEANDP
 OT_RFDP OT_IRDP OT_DEVICES OT_HISTORYDP
 OT_USER OT_USERS OT_SCHEDULER OT_USERPAGE
 OT_DEVICE OT_INTERFACES OT_XMLNODE OT_POPCLIENT
 OT_RULE OT_RULES OT_CONDITION OT_SINGLECONDITION
 OT_ALARMDP OT_UIDATA OT_FAVORITE OT_SINGLEDESTINATION
 OT_PROGRAM OT_XMENODES

Filter:
 Enthalt:

 Eigenschaften:
 Used=true Used=false Unusable=true Unusable=false Internal=true
 Used=false Used=true Unusable=false Unusable=true Internal=false

Anklicken eines Wertes in der Listdarstellung öffnet die Detaildarstellung des Objektes.

Ebenso ist es möglich, im DomScan Bereich Einträge zu suchen, welche beispielsweise nicht in den Aufzählungen gelistet sind.

Hier Beispielsweise: Scanlauf über alle Objekte aus DOM mit der Eigenschaft VARDP, die aber nicht unter ID_SYSTEM_VARIABLES gelistet sind:

The screenshot shows the Script Developer interface with a search results table and a sidebar. The table lists various objects with their properties. The sidebar includes a tree view of object types and a filter section for comparing IDs.

Methode	Type	Value	Special	ID_ID	Name	TypeName	Information
*	integer	3		11	Gateway-IP	VARDP	true
* Name	string	Root devices		1704	ActiveState	VARDP	true
Type	integer	1003		1781	ActiveState	VARDP	true
TypeName	string	RDSP		2077	-ActiveState	VARDP	false
IsRoot	boolean	true		2214	-ActiveState	VARDP	false
Internal	boolean	false		2217	-ActiveState	VARDP	false
Visible	boolean	true		3401	ActiveState	VARDP	true
Unvisible	boolean	false		3626	ActiveState	VARDP	true
EnabledWithData	string			3723	ActiveState	VARDP	true
DPInfo	string			3920	ActiveState	VARDP	true
IsVisible	boolean	true		4216	ActiveState	VARDP	true
Value	string			4323	ActiveState	VARDP	true
ValueID	string			4707	ActiveState	VARDP	true
ValueIndex	integer	1		5852	ActiveState	VARDP	true
ValueIndex0	string			6790	ActiveState	VARDP	true
ValueIndex1	string	0		6930	ActiveState	VARDP	true
ValueIndex2	string	1		7147	ActiveState	VARDP	true
ValueIndex3	string	2		7388	New program-ActiveState	VARDP	false
ValueIndex4	string	3		7389	New program-ActiveState	VARDP	false
ValueIndex5	string	4		7377	New program-ActiveState	VARDP	false
ValueIndex6	string	5		6515	-	VARDP	-
D:_EnumDevicePrograms	string						

Hier tauchen dann einige interne Datenpunkte auf, im dem Falle sind die –ActiveState keine Leichen, sondern der Anwahl Punkt Programm aktiv unter Programme.

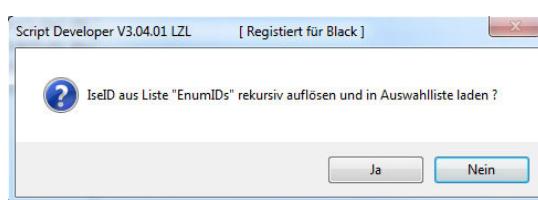
Weiterhin ist auch rekursives Arbeiten nun möglich

Hier Root Device, welches die Einträge der gelisteten Geräte enthält.

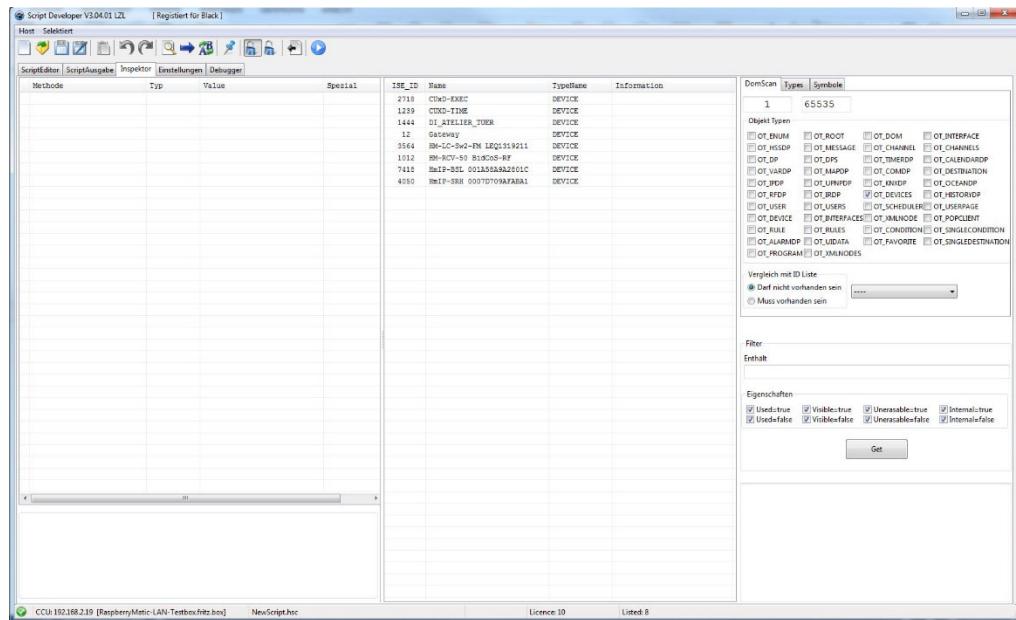
The screenshot shows the Script Developer interface with a search results table and a sidebar. The table lists various objects with their properties. The sidebar includes a tree view of object types and a filter section for comparing IDs.

Methode	Type	Value	Special	ID_ID	Name	TypeName	Information
*	integer	3		9	Root devices	DEVICE	
* Name	string	Root devices		2718	DEVICE	CDAD-EMC	
Type	integer	19		1229	DEVICE	CDAD-TIME	
TypeName	string	DEVICES		1444	DEVICE	DL_1000T_1	
IsRoot	boolean	true		12	DEVICE	DL-1000T	
Internal	boolean	false		3044	DEVICE	DL-1000T-1M	
Visible	boolean	true		1612	DEVICE	DL-BV-50-Bi	
Unvisible	boolean	false		7429	DEVICE	DLB-SB-901	
EnabledWithData	string			6533	DEVICE	DLB-SB-901	
DPInfo	string						

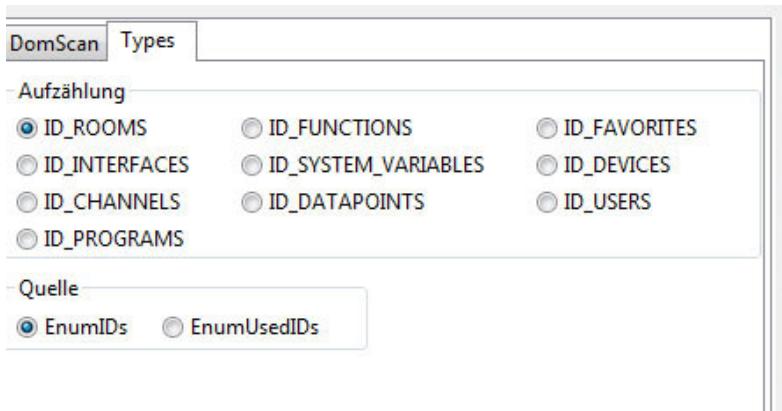
Das „R“ in der ersten Spalte zeigt an, dass ein Rekursiver Aufruf möglich ist. Doppelklicken auf die Zeile öffnet eine Sicherheitsabfrage



Wurde mit JA bestätigt, so wird die EnumList nun Aufgelöst und als neue Auswahlliste zur weiteren Bearbeitung zur Verfügung gestellt.



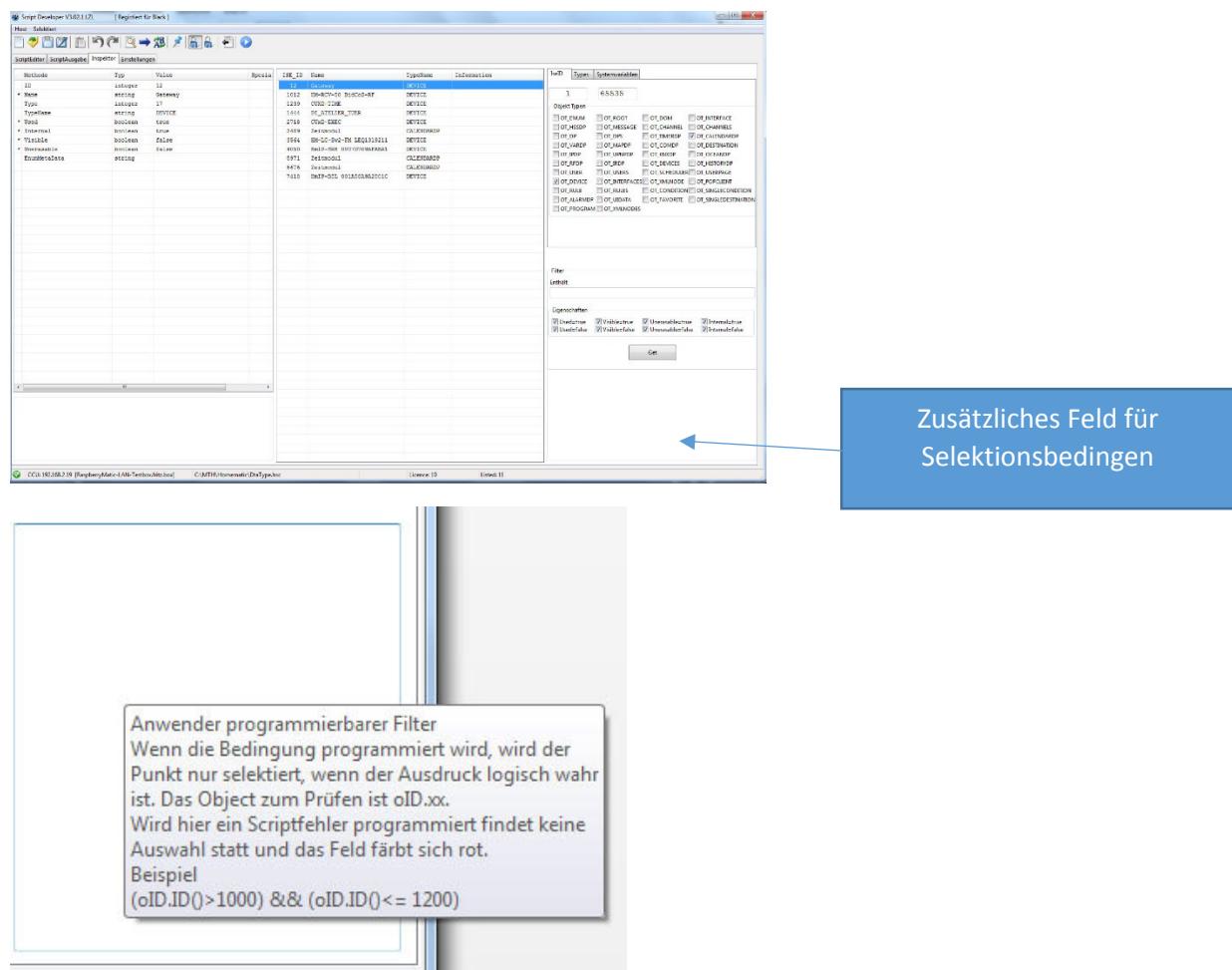
4.2 Selektionskriterium Types



Hierbei wird wie schon in der Version 3.2 in festen Bereichen Gesucht und selektiert. Schneller und einfacher zu handeln als die Objekt Selektion, dafür nicht so umfangreich.

4.3 Zusätzliche Selektionsbedingungen

Durch Druck auf Get wird die Liste gemäß Selektion von der CCU angefordert, aufbereitet und dargestellt. (Lizenzlevel vorausgesetzt)



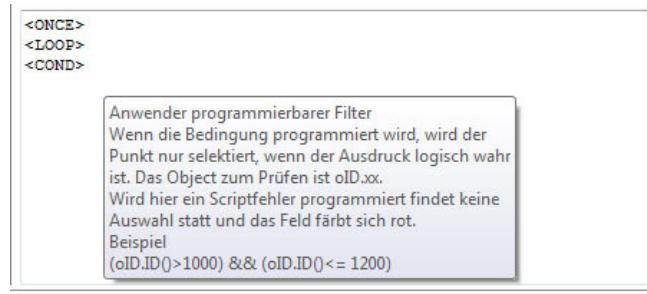
Filter sind ein mächtiges Werkzeug zum komplexen Eingrenzen und für Komplexe Abfragen.

Für die Filter existiert mittlerweile ein Kontext Menü mit rechter Maustaste:



Filter löschen entfernt sämtliche Filterbedingungen

Neuer Filter legt von der Syntax einen neuen, leeren Filter an



Mit Filter laden und speichern lassen sich nun Anwenderfilter als *.flt Datei im Verzeichnis des SDV abspeichern.

Ein Filter besteht aus den 3 Abschnitten:

<ONCE> Der Text dahinter wird am Anfang des internen Abfragescriptes quasi im einmaligen Durchlauf eingefügt. Normalerweise stehen hier Definitionen, welche nicht bei jedem Durchlauf aktualisiert werden müssen

<LOOP> Der Text dahinter wird im Zyklischen Durchlauf des Programmes innerhalb der Programmschleife eingefügt.

<COND> der Text hinter COND wird in die IF Abfrage eingefügt, welche letztlich das Objekt zur Darstellung in der Liste selektiert.

Vereinfachter Ablauf: so sieht vereinfacht das Listenselektionsprogramm aus:

```
object oID;
string s;
foreach (s,Schleifenbedingung) {
oID= dom.GetObject (s);
if (oID) {
    if (ElementBedingung) {WriteLine („Element in Liste: „ # oID.ID () );
}
}
```

Ein Anwenderdefinierter Filter wird dann in diese Grundschleife so eingebaut:

```
object oID;
string s;
ONCETEXT;

foreach (s,Schleifenbedingung) {
oID= dom.GetObject (s);
if (oID) {
    LOOPTEXT;
    if (ElementBedingung && (CONDTEXT)) {WriteLine („Element in Liste: „ # oID.ID () );
}
}
```

An diesem Kleinen Filter mal verdeutlicht:

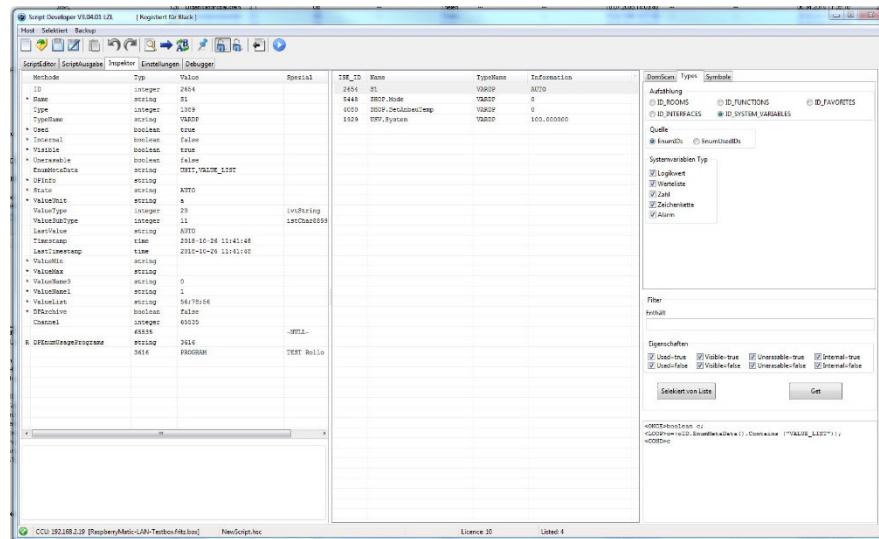
```
<ONCE>boolean c;
<LOOP>c=(oID.EnumMetaData().Contains ("VALUE_LIST"));
<COND>c
```

Daraus generiert der Scriptdeveloper folgende interne Filterabfrage:

```
object oID;
string s;
boolean c;

foreach (s,Schleifenbedingung) {
oID= dom.GetObject (s);
if (oID) {
    c=(oID.EnumMetaData() .Contains ("VALUE_LIST"));
    if (ElementBedingung && (c)) {WriteLine („Element in Liste: „ # oID.ID () );
}
}
```

Filtert aus der Gruppe der Systemvariablen alle, in deren Eigenschaft EnumMetadata das Wort VALUE_LIST vorkommt.



So lassen sich dann Filter in epischer Komplexität basteln, die man über die RegaDom stülpen kann. Zu beachten, die folgenden Variablennamen sind schon Intern vorbelegt:

object oID: darf benutzt werden, ist der Bezug auf das Objekt, welches im Filter überprüft werden soll

var v: intern benutzt zur Typerkennung: Fingers weg

string sInfo: intern benutzt zur Listengenerierung: Fingers weg

boolean b: interner Filter, auch Finger weg

string done: auch interne Benutzung, auch Finger weg

Die Filterbedingung wird in HM Script ausformuliert. Das gefundene Object kommt nur in die Liste, wenn die ausformulierte Bedingung True ist. Das Teil ist mächtig, aber auch nicht ungefährlich, man kann auch Müll als Bedingung schreiben. Dabei kommt dann aber eine Warnung:

```
<ONCE>boolean c;  
<LOOP>c=(oID.EnumMetaData().Contains ("VALUE_LIST");  
<COND>c
```

Anwender progr
Wenn die Bedin
Punkt nur selekt
ist. Das Object zu

Bedingung ist falsch: erzeugt Scriptfehler
(Klammer zu fehlt). In dem fall färbt sich nach
Druck auf Get das Feld rot

Der rar Datei liegen Standardmäßig nun schon mal 2 Filter bei:

PROGRAM_GeisterProg_CopyID - Filter um Geisterprogramme mit gesetzter CopyID zu finden

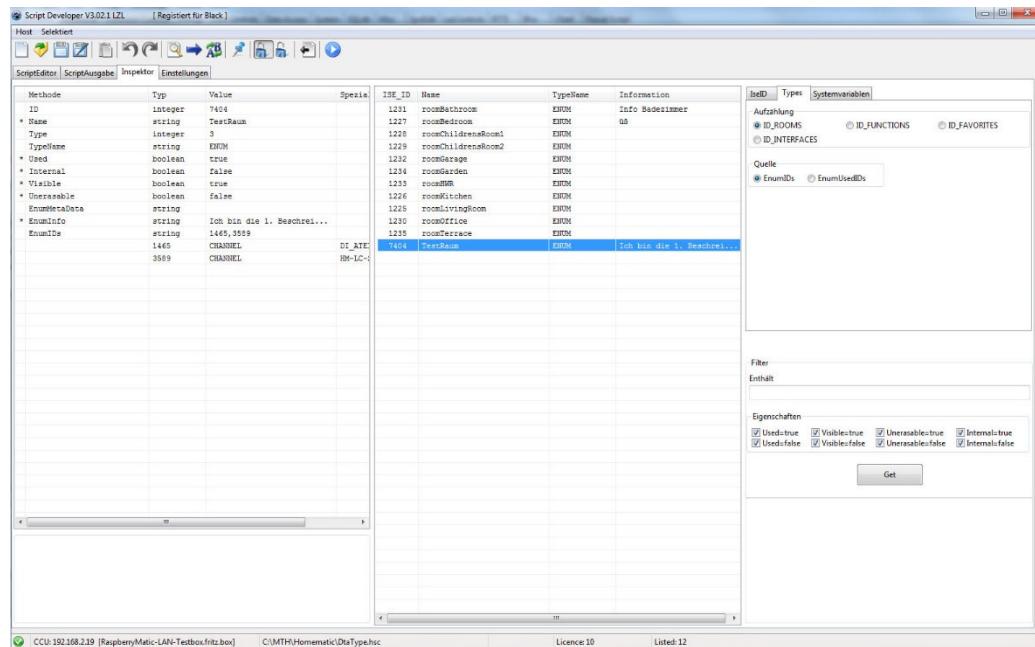
SYSVAR_VerwaisterChannel – Filter um Systemvariablen zu finden, deren Channel verweis in Nirvana zeigt

Durch Click auf die Beschreibungszeile IsID bzw Name können die Felder entsprechend sortiert werden.

Click auf eine selektierte Aufzählung öffnet im Detailfenster die Methodenansicht des Objektes

Changelog V3.03xx

Da die internen Sortieralgorithmen suboptimal arbeiteten, hat das ListView Object neue selektive Sortieralgorithmen bekommen. IsID sortiert nun wie man erwartet nach Integer aufsteigend, Name sortiert alphabetisch aufsteigend, TypeName sortiert alphabetisch, sind die Typenames gleich, wird innerhalb gleicher Typenames nach IsID numerisch sortiert.



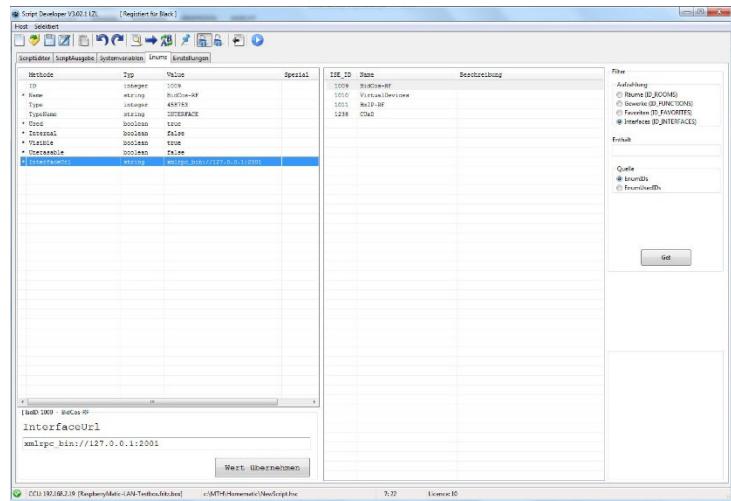
Die Spaltenbreite kann sowohl in Normalansicht als auch in Maximize separat eingestellt werden
(Das Programm sollte sich die Breiten merken und je nach Darstellungsart automatisch wieder einstellen, sollte...)

Dargestellt werden die Methode, der Vartype und die Property.

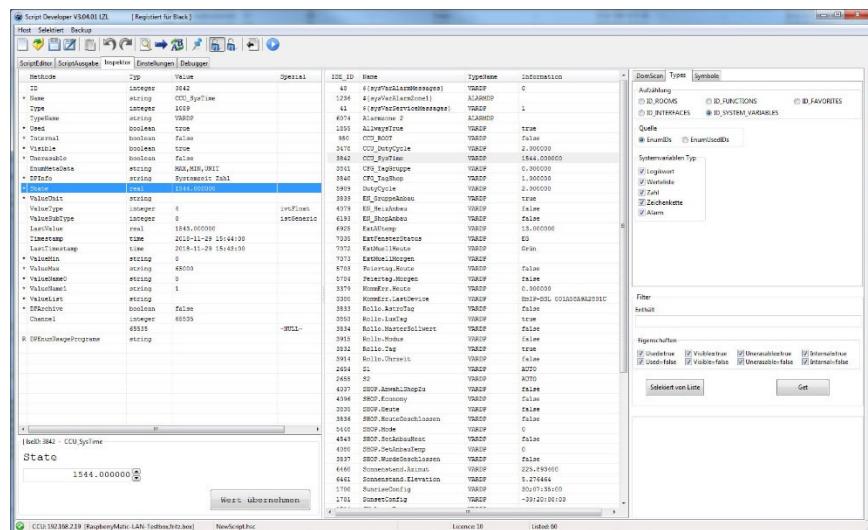
Bei den Aufzählungen wird jeweils eine Rekursionsstufe aufgelöst, um an die Detailinformationen zu kommen. Hier die Liste der Channels, die diesen Raum verwenden, aufgelöst in die ID, der Typ (hier Channels und der Name des Channels)

Properties, die in der ersten Zeile mit einem Stern (*) gekennzeichnet sind, können in ihrem Wert geändert werden.

Dazu auf die Zeile klicken



Nach Click auf Wert übernehmen wird der Wert in der CCU geändert. Also Vorsichtig mit dieser Funktion umgehen, hier gibt es kein redo.

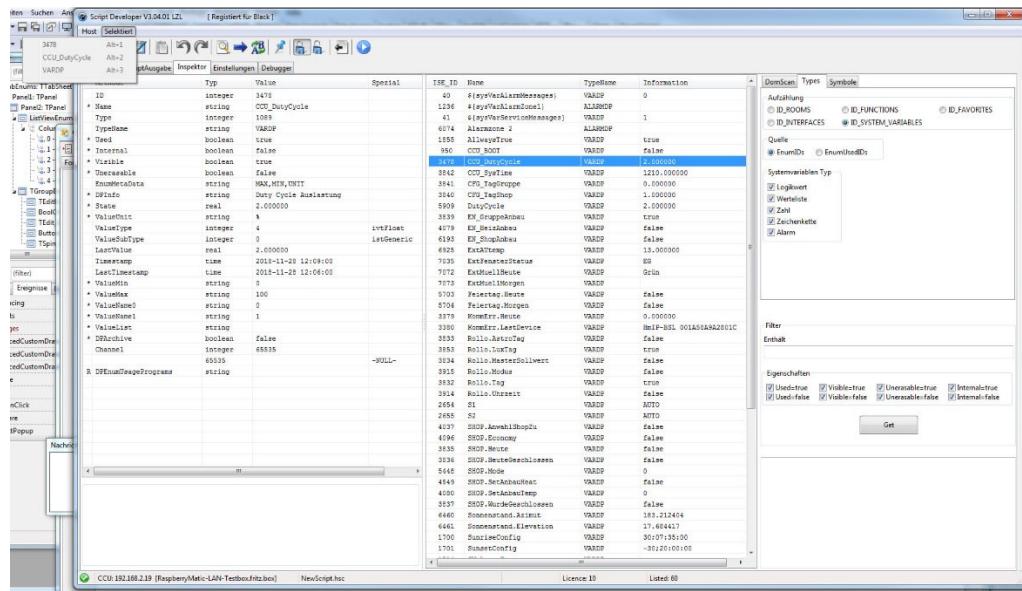


4.4 Daten aus Inspektor in Editor übernehmen

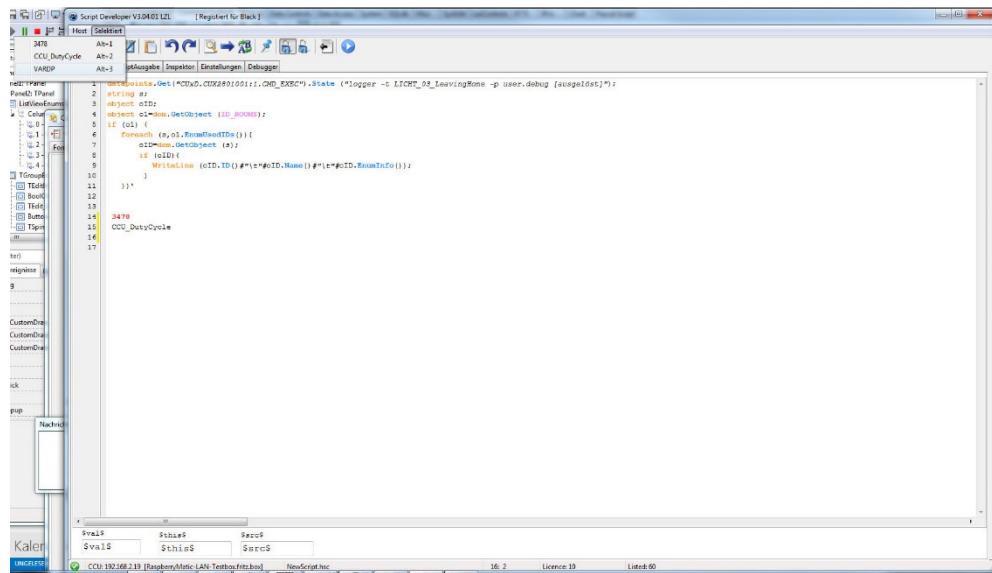
Zur schnelleren und auch möglichst fehlerfreien Bearbeitung besteht die Möglichkeit, Daten aus dem Inspektor direkt in den Editor zu übernehmen.

Immer wenn im Inspektor in den beiden Listviews auf eine Eigenschaft geklickt wurde, stehen diese Daten dann im Editor unter Selektiert zu Verfügung.

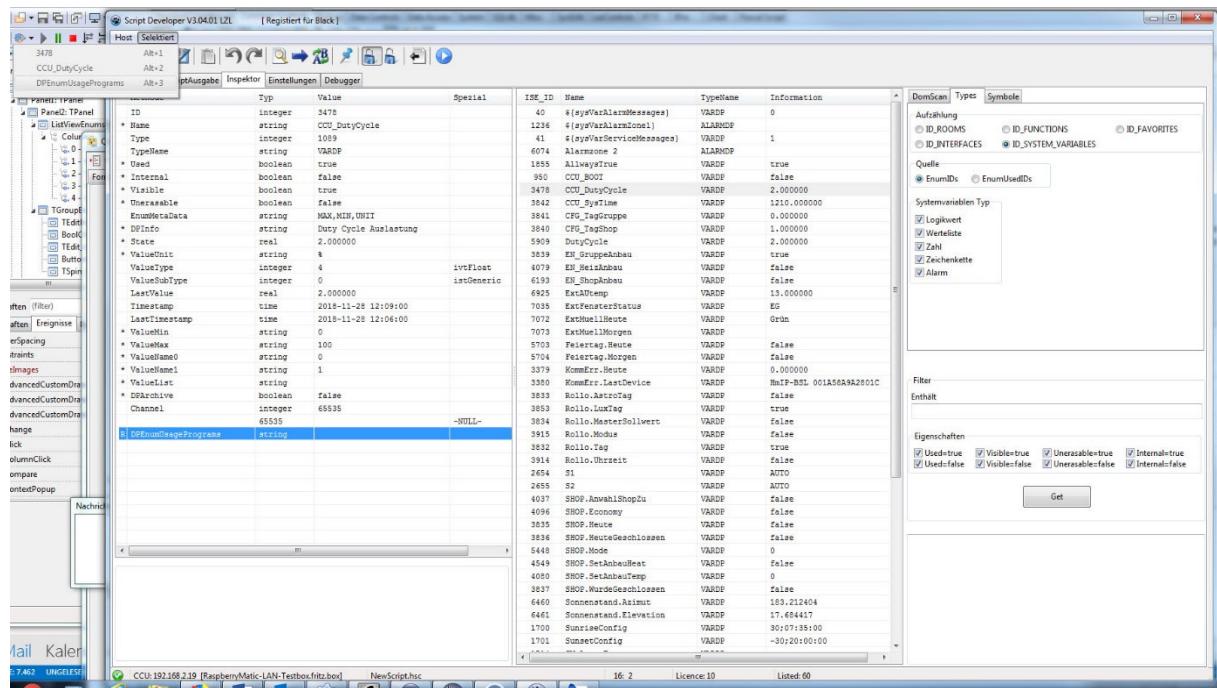
Hier Klick auf die Systemvariable



Unter selektiert sind die Eigenschaften herausgefiltert worden und lassen sich im Editor entweder durch das Menü selektiert oder durch die Kurztasten Alt-1: ID, Alt-2: Name und Alt 3: Eigenschaft bzw. Methode einfügen.



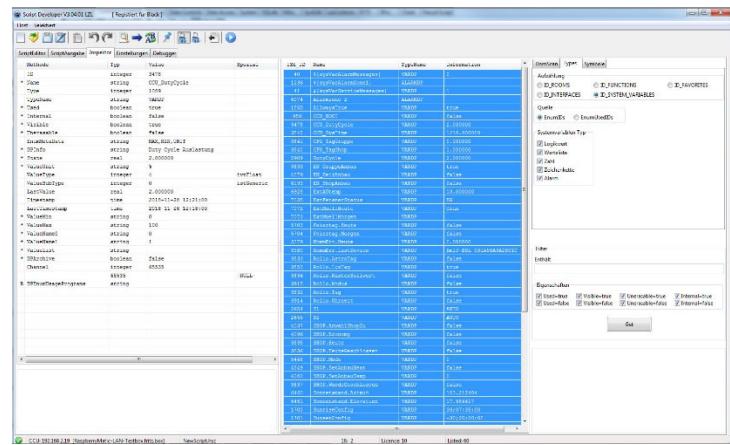
Im Detailauswahlfeld wird bei klicken auf die Methode auch noch der Methodenname gespeichert, der sich dann auch durch Alt-2 einfügen lässt



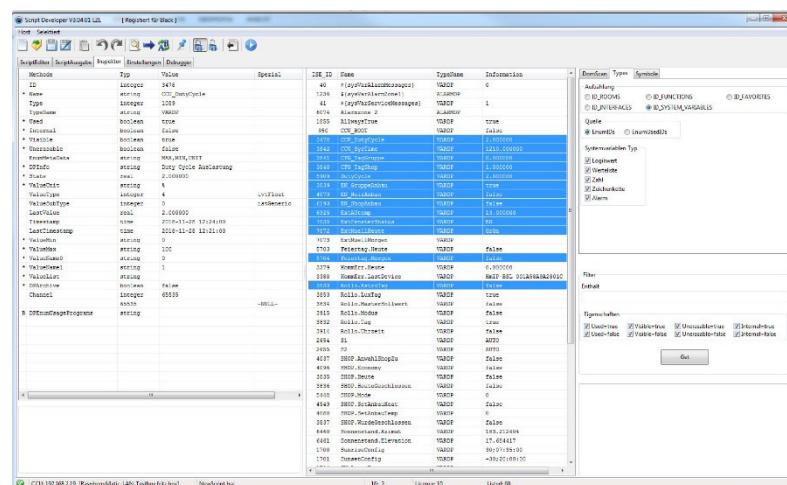
4.4.1 Mehrfachauswahl als Enum String

Es lassen sich im Hauptauswahlfeld Mehrfachselektionen vornehmen.

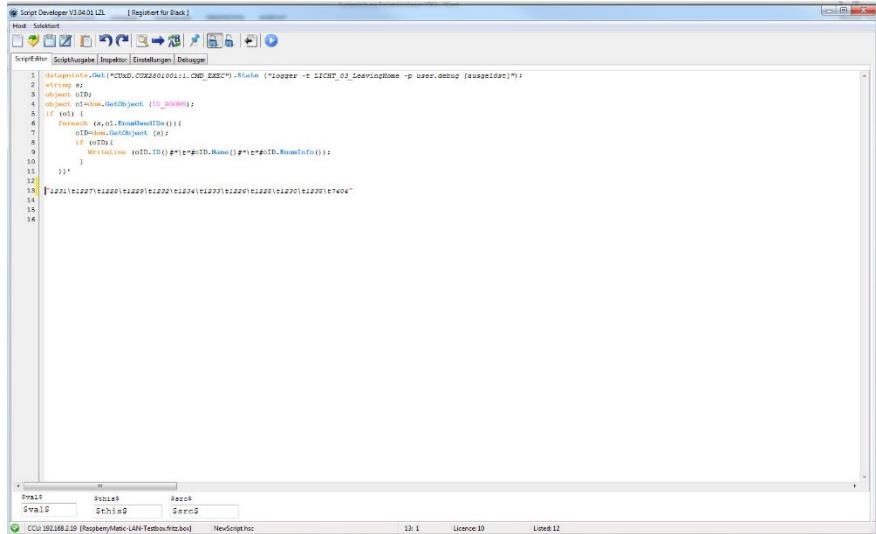
Ctrl-A : alle auswählen.



Oder die Übliche Mausbedienung:



Mit der Taste markt sich der Inspektor die Auswahl, welche sich dann Script Konform im Editor Als ID-Enum durch die Einfügen Clipboard Taste, welche nach dem Pin Druck nicht mehr grau ist, lassen sie die selektierten ID,s im Editor einfügen (z.B. zur Verarbeitung in einem Script als foreach)



Die Pinliste funktioniert nicht nur mit dem Editor, auch im Inspektor lässt sich eine mit dem Pin gemerkte Selektionsliste wieder in die Auswahl laden:

Mit rechter Maustaste im Mittleren Feld die Funktion „Einfügen aus Pinliste“ anwählen und die Sicherheitsabfrage bestätigen,

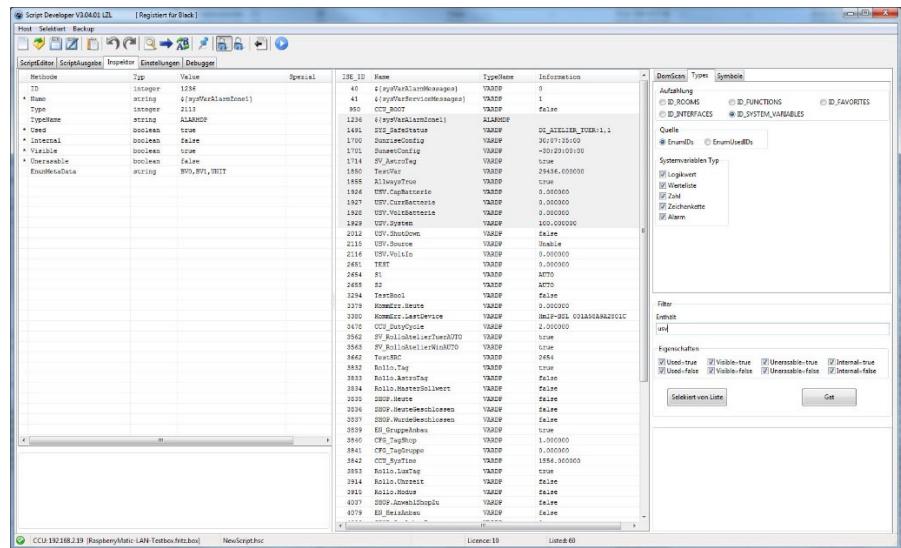


dann befinden sich die Selektierten Elemente wieder im Mittleren Feld.

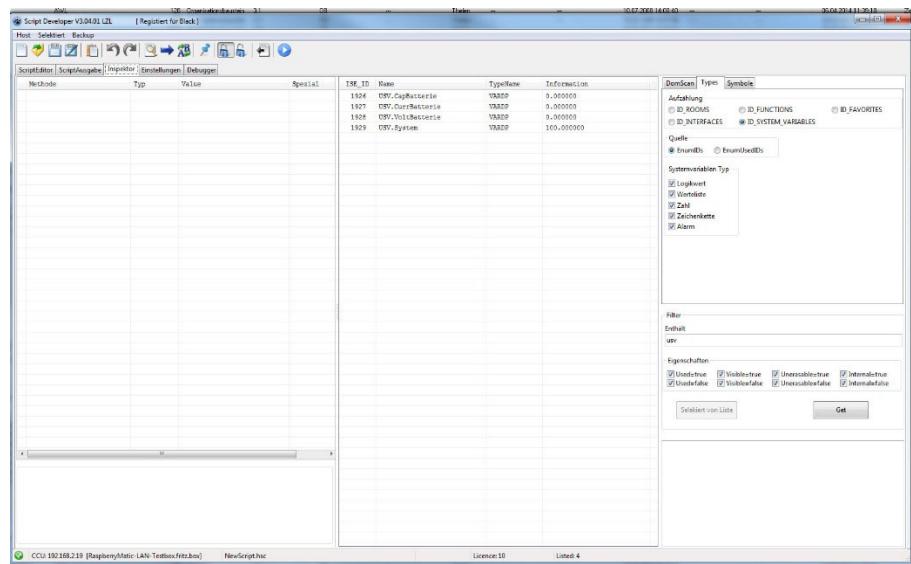
4.5 Selektion von Selektion

Befinden sich Daten in der Listendarstellung, so können daraus Bereiche selektiert werden und über diesen manuell selektierten Bereich die Auswahlfilter geschickt werden.

Hier Beispiel



Selektierter Bereich von Systemvariablen, die hier darauf gefiltert werden sollen, dass der Name den String „usv“ enthält. Es müssen 4 Sysvars gefunden werden, die IDS 2012,2115,2116 werden hier nicht berücksichtigt, da diese sind selektiert sind. Bei <Druck auf: Selektiert von Liste: ergibt sich dann

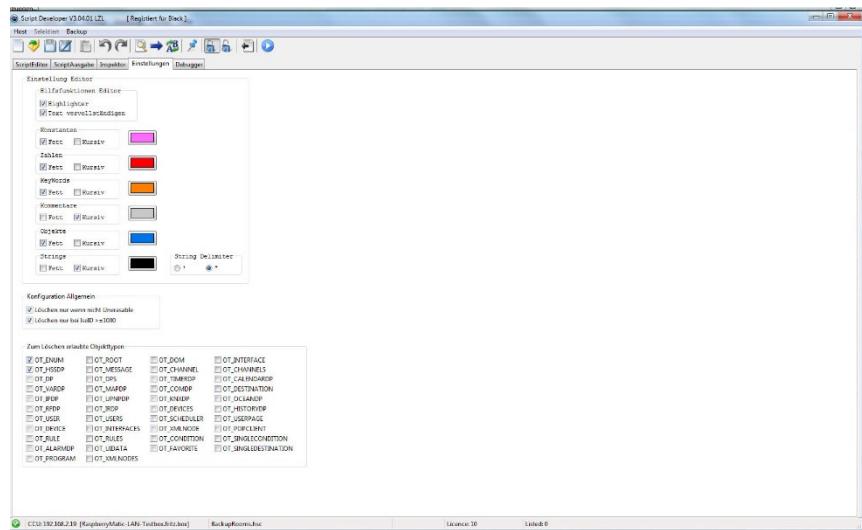


4.6 Objekte löschen

Objekte können vom SDV direkt auf der CCU gelöscht werden. Die Verantwortung, welche Objekte gelöscht werden, obliegt dem jeweiligen Anwender. Für die Löschfunktion gibt es KEIN Redo. Bevor derartige Bearbeitungen gemacht werden, IMMER vorher ein Backup machen.

Redo geht nur über restore !

Um Versehentliches löschen zu verhindern, sind ein paar Schutzmechanismen eingebaut. Generell sind Löschfunktion blockiert, wenn das Schloss in der Menüleiste auf zu steht. Um Löschen generell Freizugeben muss das schloss auf „Offen“ stehen.



Unter Einstellungen befinden sich noch ein paar Einstellungen, die Löschmöglichkeiten eingrenzen:

Löschen nur wenn nicht Unerasable: Jedes Objekt auf der CCU hat eine Property namens unerasable. (unlösbar) Ist der Haken gesetzt, geht löschen nur wenn das Objekt nicht auf unerasable = checked steht. Um nicht löscharbe Elemente zu löschen entweder:

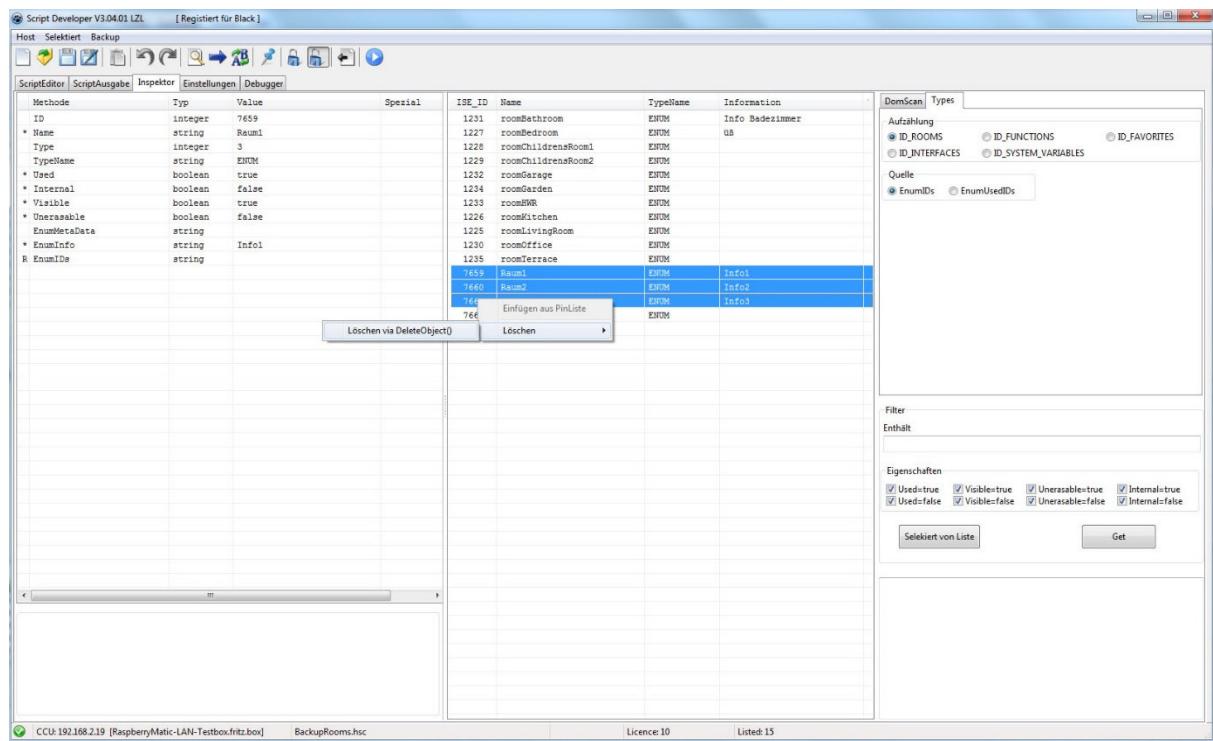
Im Inspektor unter Detailsview die Property entfernen (gilt nur für das Objekt), oder hier den Haken wegmachen (gilt für alle)

Löschen nur wenn ID \geq 1000. Dieser haken verhindert, dass man versehentlich Interne IDs der CCU (normalerweise unter kleiner 1000 angelegt) löscht. Will man in dem Bereich löschen, muss der hier explizit manuell unchecked werden.

Die Einstellungen werden NICHT gespeichert, bei jedem Neustart des SDV sind diese beiden Einstellungen wieder checked.

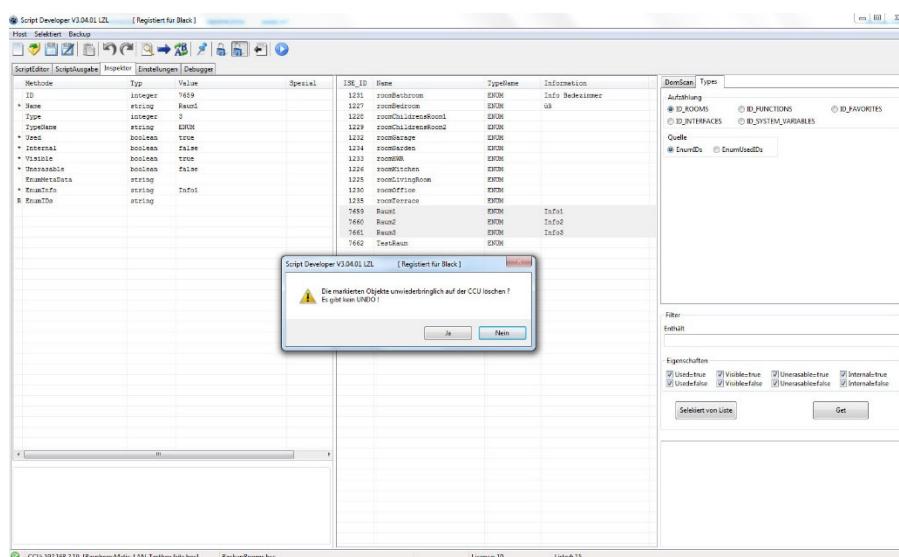
Löschen von Objekten. Die Letzte Sicherheit: ein zu lösches Objekt muss einen hier gecheckten Objekttyp haben, sonst wird es nicht gelöscht.

Löschen läuft so ab:



Objekte filtern und markieren, rechte Maustaste, Löschen, Löschen via DeleteObject ()

Mehrfachselektion ist möglich

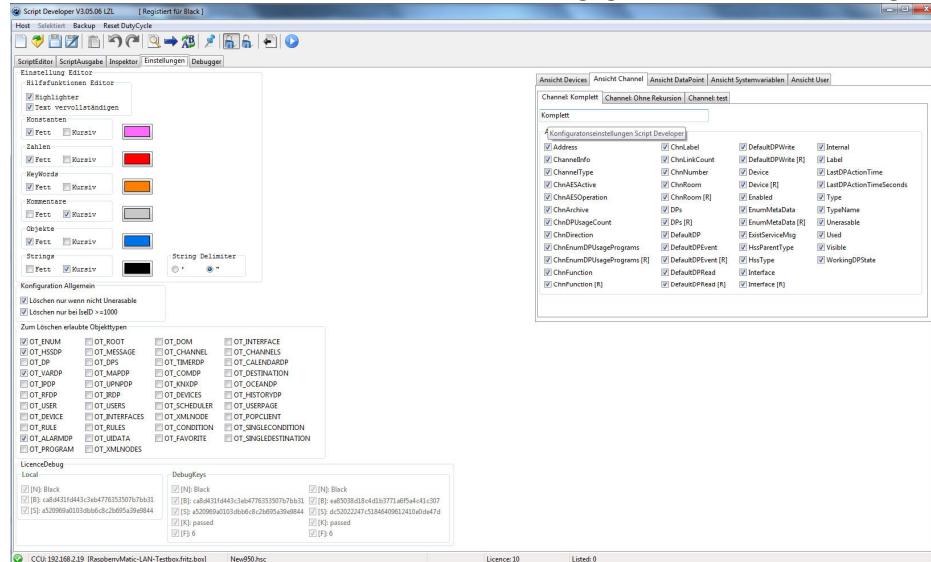


Nach dieser Sicherheitsabfrage sind die Objekte dann weg.. Zurück geht's dann nur mit restore.

4.7 Anwenderdefinierte Sichten

Die Detailansichten können stellenweise sehr umfangreich sein und auf den ersten Blick mit Information zuwerfen. Deshalb ist es möglich, für manche Objekte drei Anwenderspezifische Sichten zu definieren.

Es werden dann in der Detailansicht nur die freigegebenen Methoden dargestellt.



Sichten können mit eigenen Namen versehen werden.

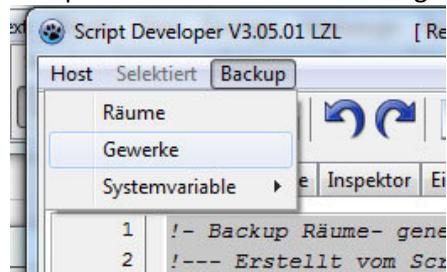
Die Entstellungen werden beim Verlassen gespeichert. Die jeweils geöffnete Sicht wird dann für das gefundene Objekt angewendet.

5 Backups

Von relevanten Objekten können Backups gemacht werden. Diese ersetzen KEIN richtiges SystemBackup an der CCU !!!

Vielmehr dienen diese im Falle eines Umzuges von einem alten System auf ein Neusystem als Hilfestellung, wenn man das alte Systembackup nicht benutzen will (Loswerden von in den Jahren angesammelten Leichen), oder aber ein inkonsistentes System.

Den passenden Lizenzlevel vorausgesetzt, findet sich die Backups hier:



Devices müssen VORHER manuell umgezogen worden sein über ablernen und neu anlernen. Und die Geräte müssen, damit die Backups von Räumen und Gewerken sinnig arbeiten können, wieder ihre „alten“ Namen haben.

Siehe dazu auch die passende EQ3 Dokumentation. Der SDV legt keine neuen Devices oder Direktverbindungen an.

5.1 Räume

Auf dem Bestandssystem wird ein Restore Programm erzeugt und lässt sich anschließend auf dem PC abspeichern. Der vorgeschlagene Dateiname ist dabei backup_Rooms_ + datum und Uhrzeit der Generierung.

Auf dem „Neusystem“ lässt sich dieses Programm über den SDV via Scripteditor dann starten. Dabei passiert folgendes:

Es wird geprüft, ob dort schon ein Raum mit dem Namen „XX“ existiert. Wenn ja, gut, wenn nein, wird dieser Raum neu anlegt, mit Namen und Beschreibung versehen und in ID_ROOMS eingehängt. Waren dem alten Raum Kanäle zugeordnet, so versucht der SDV nun diese Kanäle des Altsystems über ihren Kanalnamen zu identifizieren. Ist dieses erfolgreich, so wird dieser Kanal dem Raum hinzugefügt.

5.2 Gewerke

Auf dem Bestandssystem wird ein Restore Programm erzeugt und lässt sich anschließend auf dem PC abspeichern. Der vorgeschlagene Dateiname ist dabei backup_Functions_ + datum und Uhrzeit der Generierung.

Auf dem „Neusystem“ lässt sich dieses Programm über den SDV via Scripteditor dann starten. Dabei passiert folgendes:

Es wird geprüft, ob dort schon ein Gewerk mit dem Namen „XX“ existiert. Wenn ja, gut, wenn nein, wird dieses Gewerk neu anlegt, mit Namen und Beschreibung versehen und in ID_FUNCTIONS eingehängt.

Waren dem alten Gewerk Kanäle zugeordnet, so versucht der SDV nun diese Kanäle des Altsystems über ihren Kanalnamen zu identifizieren. Ist dieses erfolgreich, so wird dieser Kanal dem Gewerk hinzugefügt.

5.3 Systemvariablen

Der komplizierteste Part.

Auf dem Bestandssystem wird ein Restore Programm erzeugt und lässt sich anschließend auf dem PC abspeichern. Der vorgeschlagene Dateiname ist dabei backup_Sysvars_ + datum und Uhrzeit der Generierung.

Auf dem „Neusystem“ lässt sich dieses Programm über den SDV via Scripteditor dann laden.

Hierbei können noch folgende Einstellungen in dem Programm Kopf vorgenommen werden:

```
----- Scriptausgabe -----
!-      Backup SystemVariablen vom 06.12.2018 13:21:02
!-      Erstellt mit Script Developer V3.04 by Black 2018
!----- Diese Zeilen Anpassen -----
boolean bcreate= true; !- Anlegen, wenn noch nicht existierte
boolean bupdate= true; !- Wert Updaten, wenn vorhanden und gleicher Typ
boolean barchive= false; !- false: immer restore mit DPArchive (false), true: restore mit
altem Wert

bcreate:
true: wenn die Systemvariable noch nicht existiert wird diese angelegt und in ID_SYSTEM_VARIABLES
eingehängt.
False: wenn die Systemvariable noch nicht existierte, wird auch nix gemacht.

bupdate:
true: wenn die Systemvariable schon existierte und diese den gleichen Typ hat, wird der State wert
aus dem Backup in die variable geschrieben. Wenn nicht der gleiche Typ- passiert nix
false: wenn die Systemvariable schon existiert- wird nix gemacht

barchive: (nur bei Neuanlage)
true: beim Restore wird die Archiv Option der Systemvariable aus dem Backup genommen.
False: es wird immer ohne Archiv Option angelegt beim Restore.
```

Der SDV unterscheidet dabei von sich aus zwischen Alarm und Systemvariable. Bei Alarm wird nicht der Zustand (AllsArmed) verändert. Heisst: bei Neu Anlage sind die Alarne immer scharf, auch wenn dieser Alarm vorher im Alt System über AIArm (false) unscharf geschaltet wurde !

Zugeordnete Channels werden ebenfalls versucht zu rekonstruieren, so sich der Kanal über den alten Kanalnamen identifizieren lässt (s.a. Räume und Gewerke)

5.4 Devices und Kanäle

Bei diesem Backup werden die Namen der Kanäle und Geräte gesichert. Die Identifikation erfolgt später über das Interface und die Seriennummer, die der Kanäle durch Durchiterieren und Vergleich mit ChnNumber Methode.

Hilfreich beim Umzug von einem System auf ein anderes System. Nachdem die Geräte abgelernt und am neuen System MANUELL !!!! angelernt wurden, kann das Restore Programm die alten Namen anhand der Seriennummern wiederherstellen. Anschließend können die Raum / Gewerk und Systemvariablen Restore gemacht werden.

6 Kleine Helfer im Alltag

6.1 Umbenennen von Kanälen von Geräten

Wer hatte nicht schon alles die Freude, z.B. an einem neu angelernten IP Gerät mit 14 Kanälen die Namen neu zu vergeben. Dies geht nun schneller.

Das Device wird selektiert und der Name der Device geändert.

Anschliessend rechte maustaste auf das Device in der Listendarstellung und Punkt auswählen:

ISE_ID	Name	TypeName	Information
2718	CUxD-EXEC	DEVICE	CUxD
1239	CUXD-TIME	DEVICE	CUxD
1444	DI_ATELIER_TUER	DEVICE	BidCos-RF
12	Gateway	DEVICE	---
3564	HM-LC-Sw2-FM LEQ1319211	DEVICE	BidCos-RF
1012	HM-RCV-50 BidCoS-RF	DEVICE	BidCos-RF
74	Einfügen aus PinListe		HmIP-RF
76	Löschen		HmIP-RF
75			HmIP-RF
40	Kanäle von diesem Device umbenennen		HmIP-RF

Rückfrage mit Ja bestätigen und die Kanäle werden so benannt:

Device: DeviceName

Kanal0 : DeviceName:0

Kanal1 : Devicename:1

Etc...