


## Kurzanleitung Scriptdeveloper V3.05.xx (06. 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.xx (06. Dezember 2018) .....	1
1. Installation .....	3
1.1 Lizenzierung .....	4
1.2 Systemvoraussetzungen .....	7
1.3 Was tuts bis jetzt .....	7
1.4 Bekannte Einschränkungen / Bugs .....	7
1.6 Changelog .....	8
1.6.1 Changelog 03.05.01 LZL .....	8
1.6.2 Changelog 03.04.01 LZL .....	8
1.6.3 Changelog 03.03.01 LZL .....	8
2 Oberfläche .....	8
3 Scripteditor .....	10
3.1 Voreinstellungen Editor .....	11
3.2 Vervollständigen Funktion .....	12
4 Inspektor .....	13
4.1 Selektionswahl: DomScan .....	14
4.2 Selektionskriterium Types .....	17
4.3 Zusätzliche Selektionsbedingungen .....	18
4.4 Daten aus Inspektor in Editor übernehmen .....	24
4.4.1 Mehrfachauswahl als Enum String .....	26
4.5 Selektion von Selektion .....	28
4.6 Objekte löschen .....	29
5 Backups .....	31
5.1 Räume .....	31
5.2 Gewerke .....	31
5.3 Systemvariablen .....	32

## 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

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 exklusiven Zugriff hat. Auf Systemen ohne CUXd kann der SDV nicht eingesetzt werden.

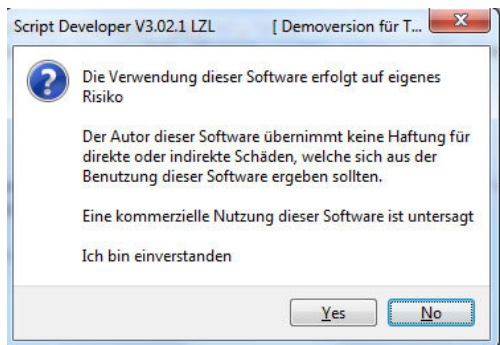
## 1.1 Lizenzierung

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 einzugrenzen. Dies geschieht durch Vergabe von bis zu 2 Lizenzschlüsseln. Der SDV ist dadurch an bis zu 2 CCU / Raspberrymatic gepairt.

Wie arbeitet das ?

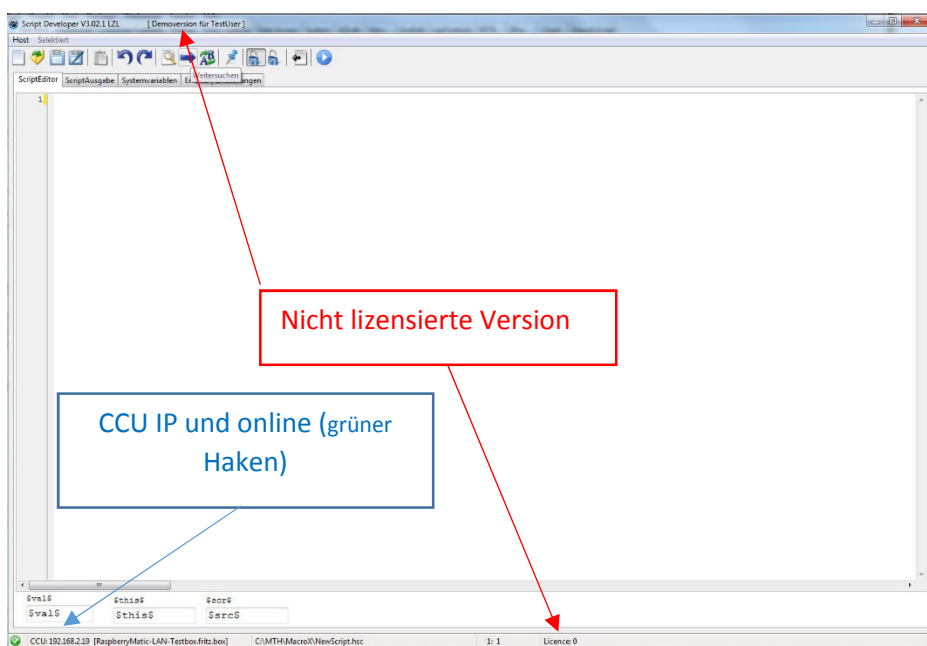
Der SDV telefoniert nicht nach Hause. Um eine Lizenz anzufragen ist folgender Weg einzuschlagen.

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

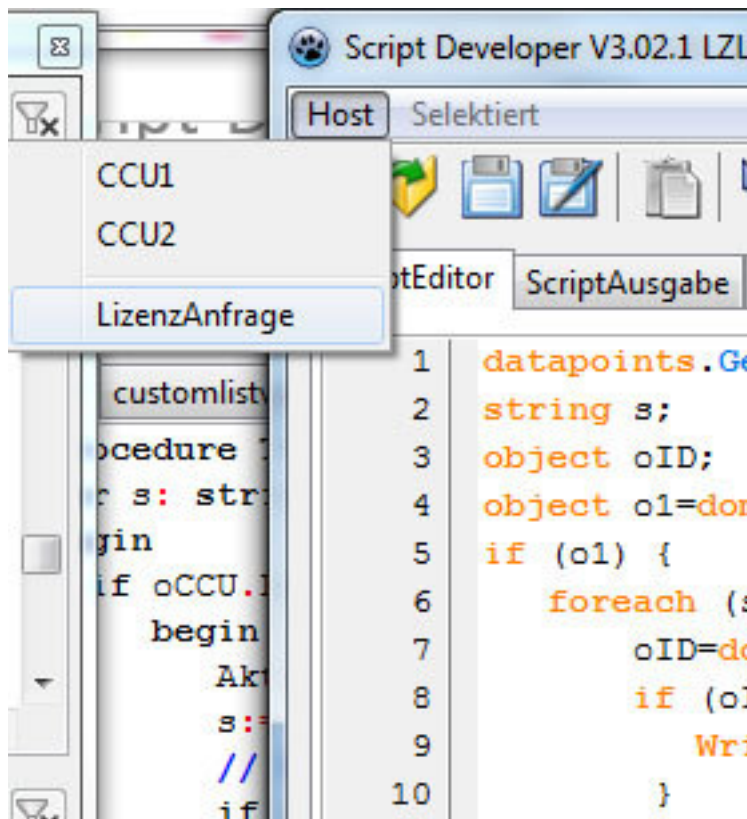


Bei allerersten Start muss dieses Fenster mit yes bestätigt werden. No führt so einem sofortigen Programmabbruch

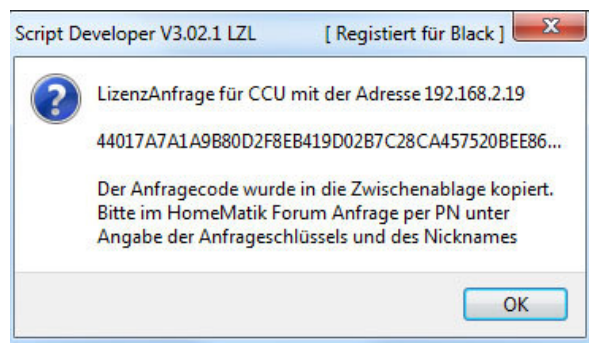
Bei Bestätigung mit Yes startet nun zum erstenmal der SDV als Demoversion



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 Testinstallationen 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 Nutzernamen/Passwort implementiert. Der SDV braucht Zugriff über Port :8181

Auf einer CCU sind die erzeugten 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 Editiermöglichkeiten.

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 03.05.01 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

### 1.6.2 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)  
Merkens 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

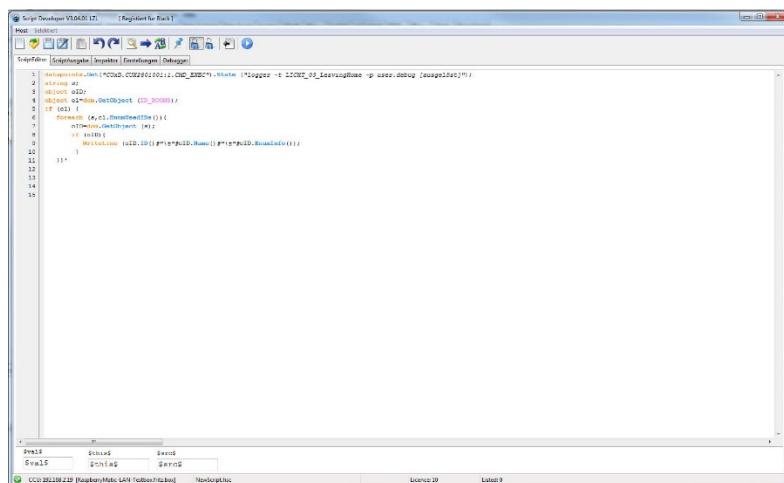
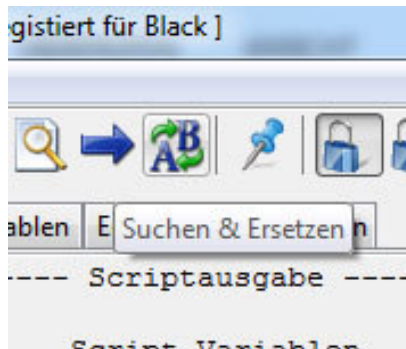
### 1.6.3 Changelog 03.03.01 LZL




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

## 2 Oberfläche

Zu fast allen Funktionen sind die Hint parametrisiert, 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

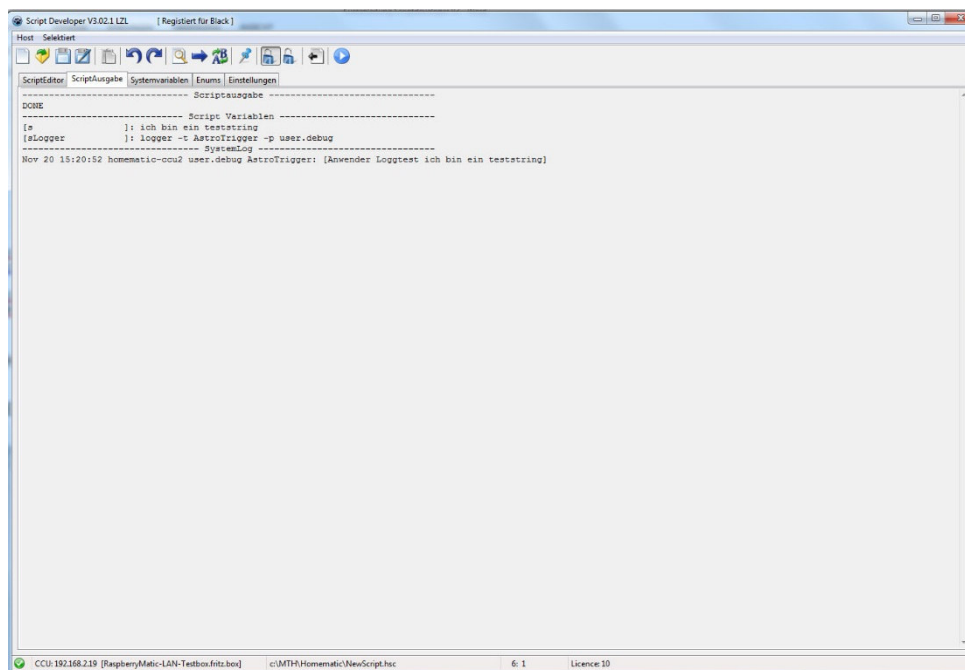
### 3 Scripteditor

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:



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

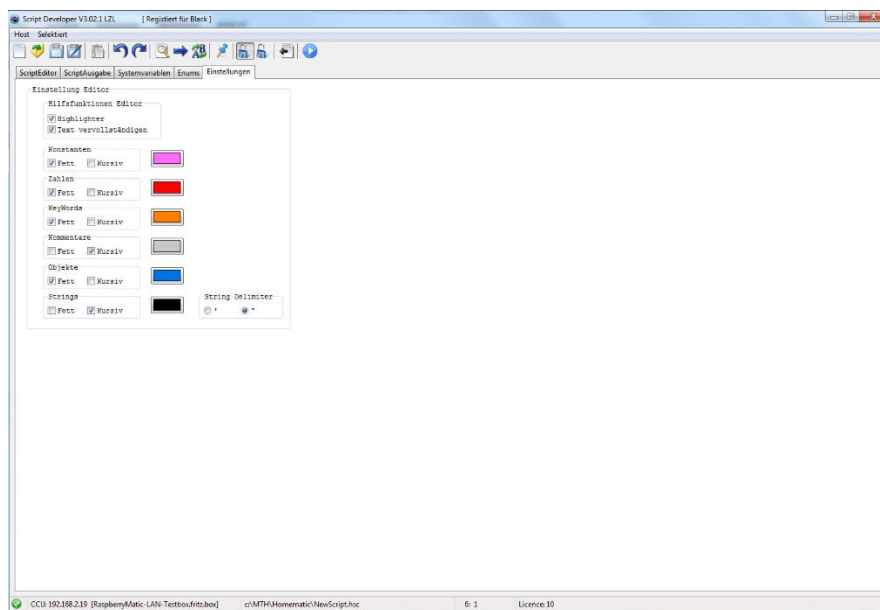
Unter lokale Script variablen stehen die Variablen welche im Script definiert wurden mit ihren Namen. In dem Fall hier sind das die Beiden String Variablen s und sLogger.

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 Loggtest " # s # "]);^M WriteLine ("DONE");^M
[iseESP.cpp:1121]
Jun 15 12:41:49 homematic-raspi local0.err ReGaHss: Error: ParseProgram: SyntaxError=
(sLogger # "[Anwender Loggtest " # s # "]);^M WriteLine ("DONE"); [iseESP.cpp:374]
```

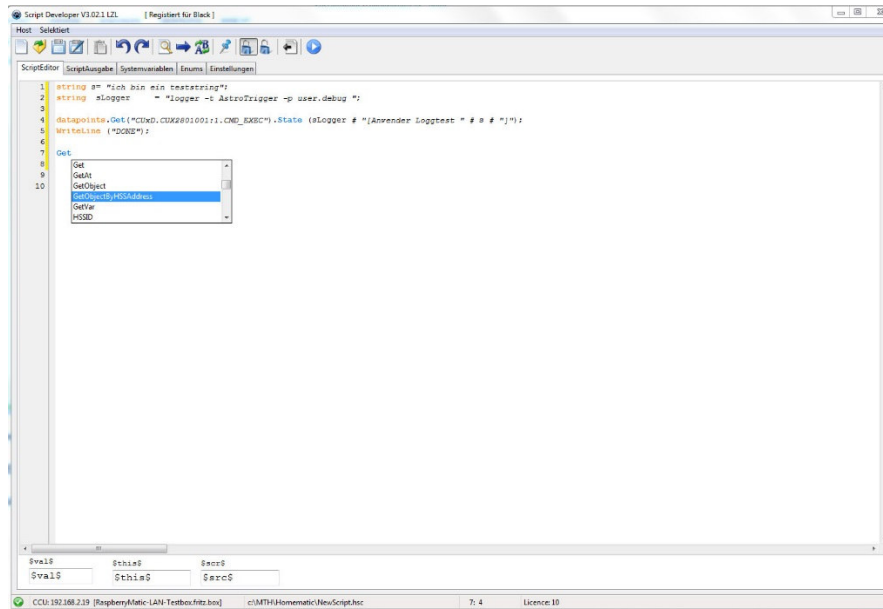
### 3.1 Voreinstellungen Editor



Hier kann nach Vorliebe der Highlighter konfiguriert oder auch ausgeschaltet werden.  
Ebenso lässt sich die Vervollständigen Funktion an oder abwählen.  
Vorbereitung natürlich: Lizenzlevel muss vorhanden sein.

### 3.2 Vervollständigen Funktion

Methoden und Konstanten Namen muss man sich nicht 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.

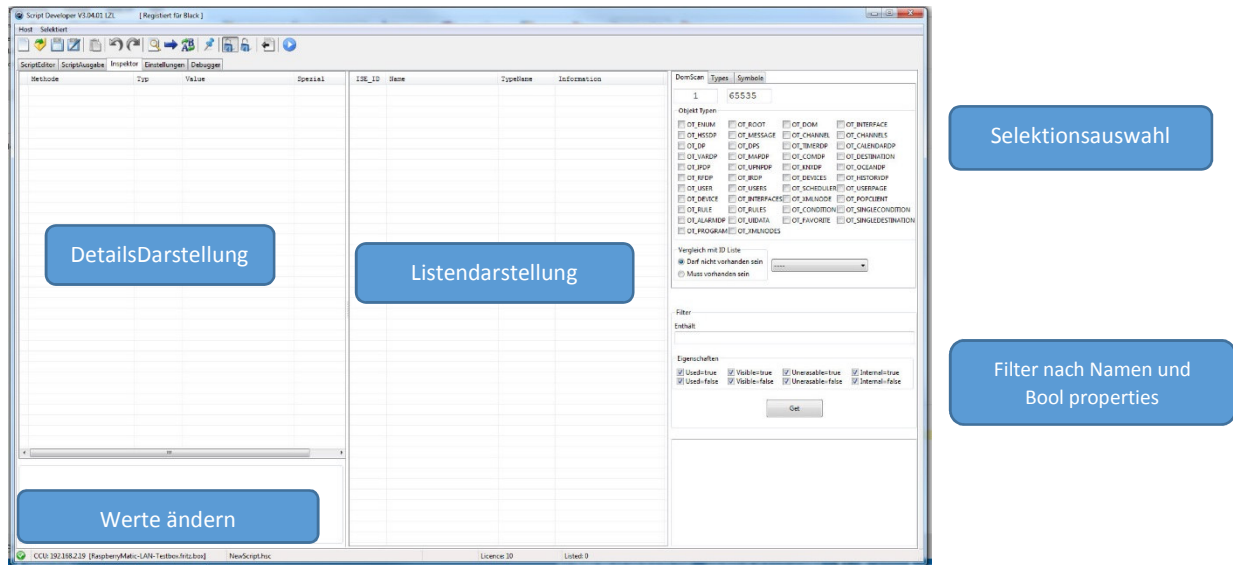


Nach Druck von Enter erscheint das Wort im Editor.

## 4 Inspektor

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

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

DomScan

Types

Symbole

1

65535

Objekt Typen

<input type="checkbox"/> OT_ENUM	<input type="checkbox"/> OT_ROOT	<input type="checkbox"/> OT_DOM	<input type="checkbox"/> OT_INTERFACE
<input type="checkbox"/> OT_HSSDP	<input type="checkbox"/> OT_MESSAGE	<input type="checkbox"/> OT_CHANNEL	<input type="checkbox"/> OT_CHANNELS
<input type="checkbox"/> OT_DP	<input type="checkbox"/> OT_DPS	<input type="checkbox"/> OT_TIMERDP	<input type="checkbox"/> OT_CALENDARDP
<input type="checkbox"/> OT_VARDP	<input type="checkbox"/> OT_MAPDP	<input type="checkbox"/> OT_COMDP	<input type="checkbox"/> OT_DESTINATION
<input type="checkbox"/> OT_IPDP	<input type="checkbox"/> OT_UPNPDP	<input type="checkbox"/> OT_KNXDP	<input type="checkbox"/> OT_OCEANDP
<input type="checkbox"/> OT_RFDP	<input type="checkbox"/> OT_IRDP	<input type="checkbox"/> OT_DEVICES	<input type="checkbox"/> OT_HISTORYDP
<input type="checkbox"/> OT_USER	<input type="checkbox"/> OT_USERS	<input type="checkbox"/> OT_SCHEDULER	<input type="checkbox"/> OT_USERPAGE
<input type="checkbox"/> OT_DEVICE	<input type="checkbox"/> OT_INTERFACES	<input type="checkbox"/> OT_XMLNODE	<input type="checkbox"/> OT_POPCLIENT
<input type="checkbox"/> OT_RULE	<input type="checkbox"/> OT_RULES	<input type="checkbox"/> OT_CONDITION	<input type="checkbox"/> OT_SINGLECONDITION
<input type="checkbox"/> OT_ALARMDP	<input type="checkbox"/> OT_UIDATA	<input type="checkbox"/> OT_FAVORITE	<input type="checkbox"/> OT_SINGLEDESTINATION
<input type="checkbox"/> OT_PROGRAM	<input type="checkbox"/> OT_XMLNODES		

Vergleich mit ID Liste

☒ Darf nicht vorhanden sein
 ☐ Muss vorhanden sein

----

Eingabe des Scan Bereiches der IsELD'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.

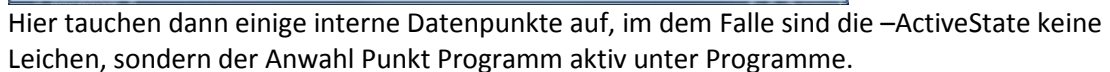
Damit ein Objekt Selektiert wird, muss es die angeklickte Objekteigenschaft haben.

Mehrfachangaben sind möglich

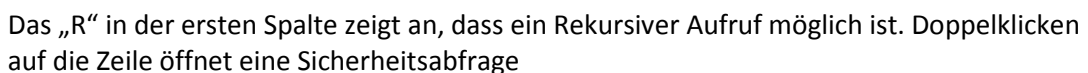
Beispiel für Suchen aller Objekte mit der Eigenschaft OT\_DEVICE im Bereich der ISE Nummern 1-65535

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

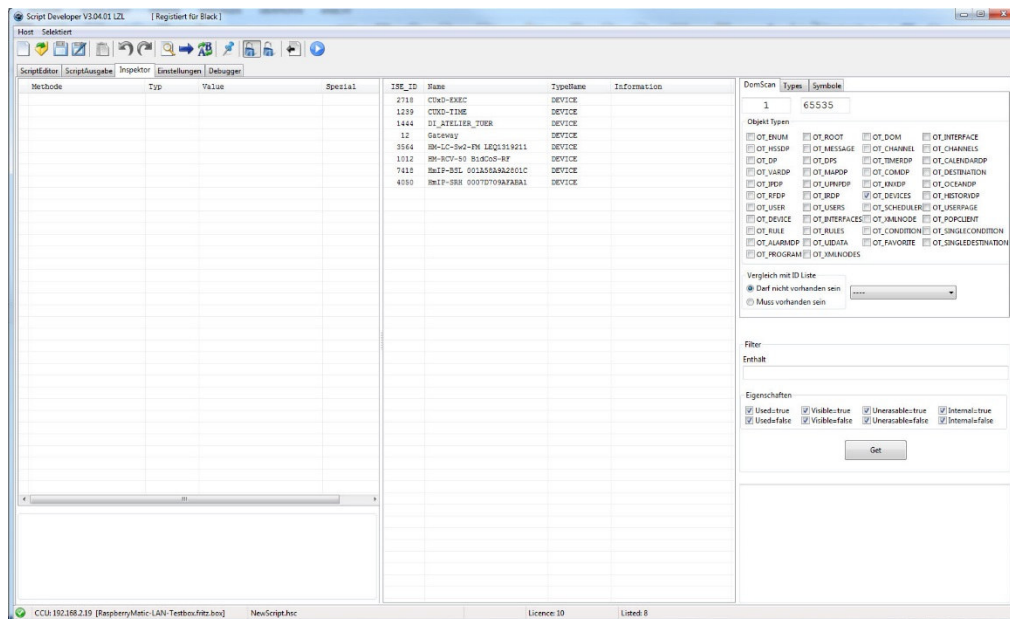
Hier Beispielsweise: Scanlauf über alle Objekte aus DOM mit der Eigenschaft VARDP, die aber nicht unter ID\_SYSTEM\_VARIABLES gelistet sind:



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



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





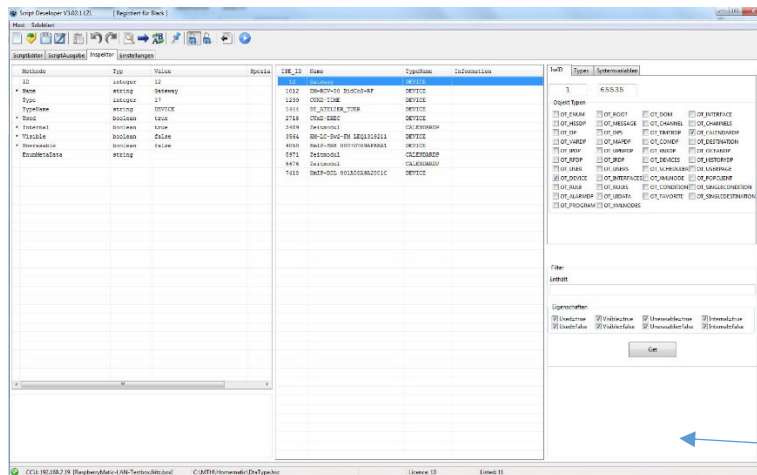
## 4.2 Selektionskriterium Types

The screenshot shows a software interface with three tabs: 'IselD', 'Types', and 'Systemvariablen'. The 'Types' tab is active. It contains two sections: 'Aufzählung' and 'Quelle'. In the 'Aufzählung' section, there are three radio buttons: 'ID\_ROOMS' (selected), 'ID\_FUNCTIONS', and 'ID\_FAVORITES'. Below it, there is a radio button for 'ID\_INTERFACES'. In the 'Quelle' section, there are two radio buttons: 'EnumIDs' (selected) and 'EnumUsedIDs'.

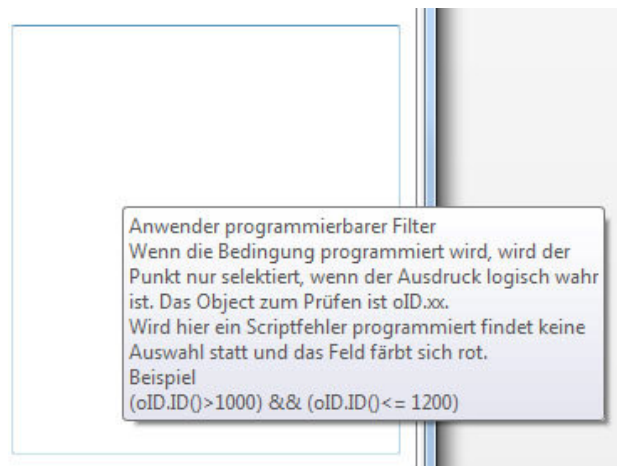
Hierbei wird wie schon in der Version 3.2 in festen Bereichen Gesucht und selektiert. Schneller und einfacher zu handeln als die Objekt Selektion, dafür nicht so umfangreich.

### 4.3 Zusätzliche Selektionsbedingungen

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

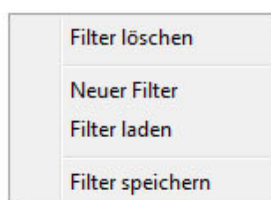


Zusätzliches Feld für Selektionsbedingen



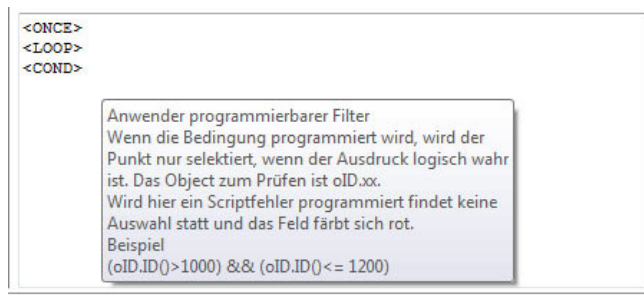
Filter sind ein mächtiges Werkzeug zum komplexen Eingrenzen und für Komplexe Abfragen.

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 () );
}
}
```

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

```
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 () );
}
}
```

```
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 Filder, 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 programmiert  
Wenn die Bedingung  
Punkt nur selektiert  
ist. Das Object zu

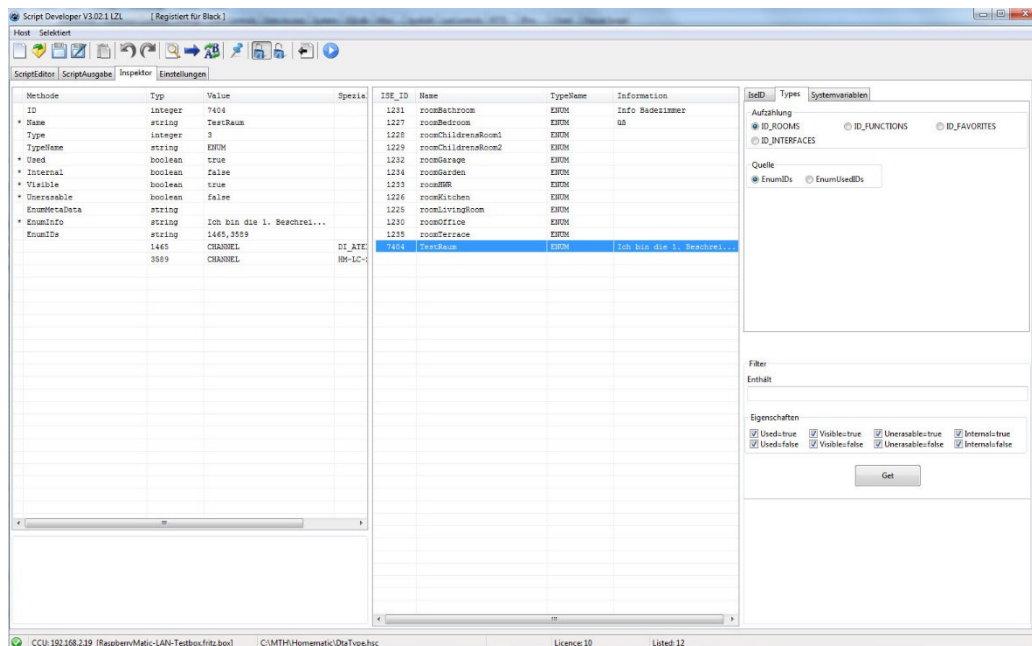
Bedingung ist falsch: erzeugt Scriptfehler  
(Klammer zu fehlt). In dem fall färbt sich nach  
Druck auf Get das Feld rot

Durch Click auf die Beschreibungszeile IseID 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 Sortialgorithmen suboptimal arbeiteten, hat das ListView Object neue selektive Sortialgorithmen bekommen. IseID sortiert nun wie man erwartet nach Integer aufsteigend, Name sortiert alphabetisch aufsteigend, TypeName sortiert alphabetisch, sind die Typenames gleich, wird innerhalb gleicher Typenames nach IseID numerisch sortiert.



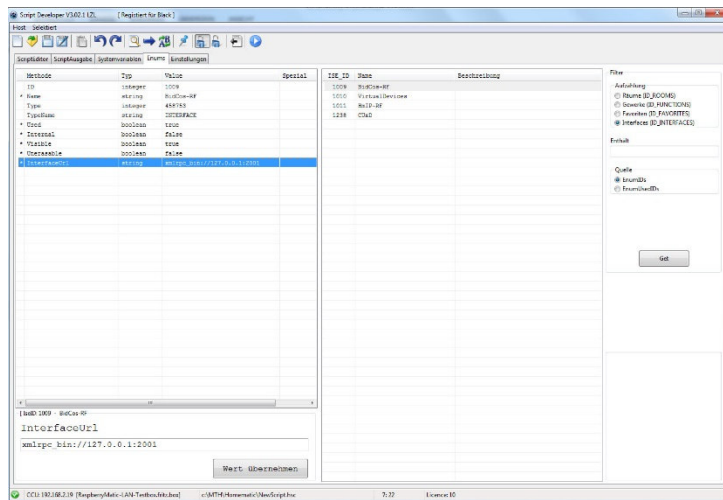
Die Spaltenbreite kann sowohl in Normalansicht als auch in Maximize separat eingestellt werden (Das Programm sollte sich die Breiten merken und je nach Darstellungsart automatisch wieder einstellen, sollte...)

Dargestellt werden die Methode, der Vartype und die Property.

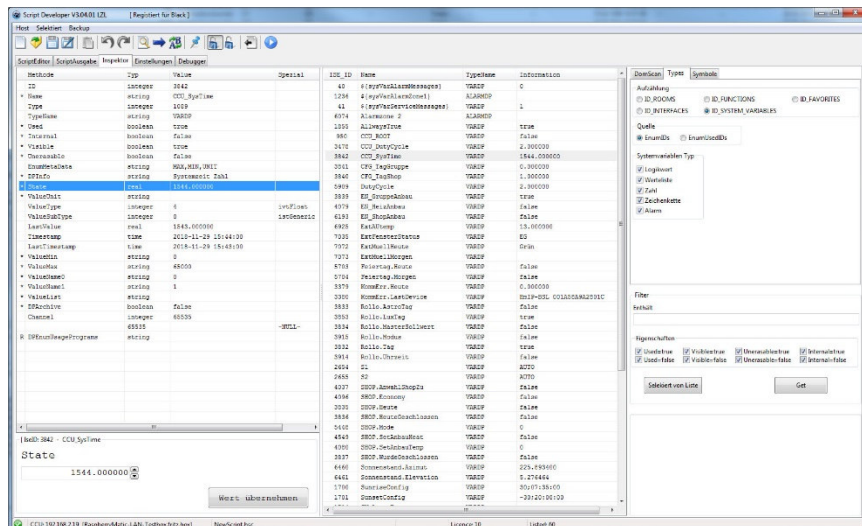
Bei den Aufzählungen wird jeweils eine Rekursionsstufe aufgelöst, um an die Detailinformationen zu kommen. Hier die Liste der Channels, die diesen Raum verwenden, aufgelöst in die ID, der Typ (hier Channels und der Name des Channels)

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.

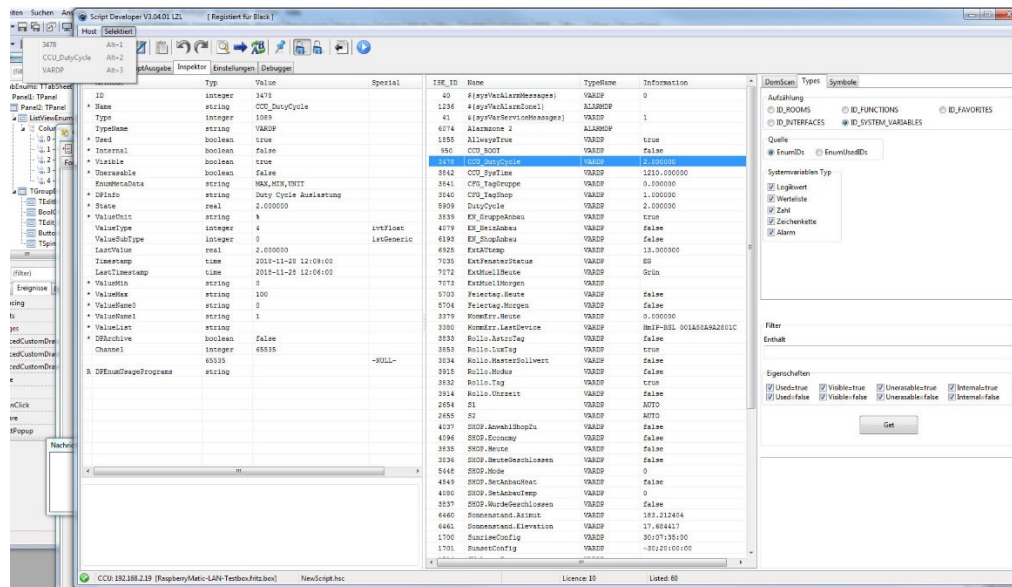


## 4.4 Daten aus Inspektor in Editor übernehmen

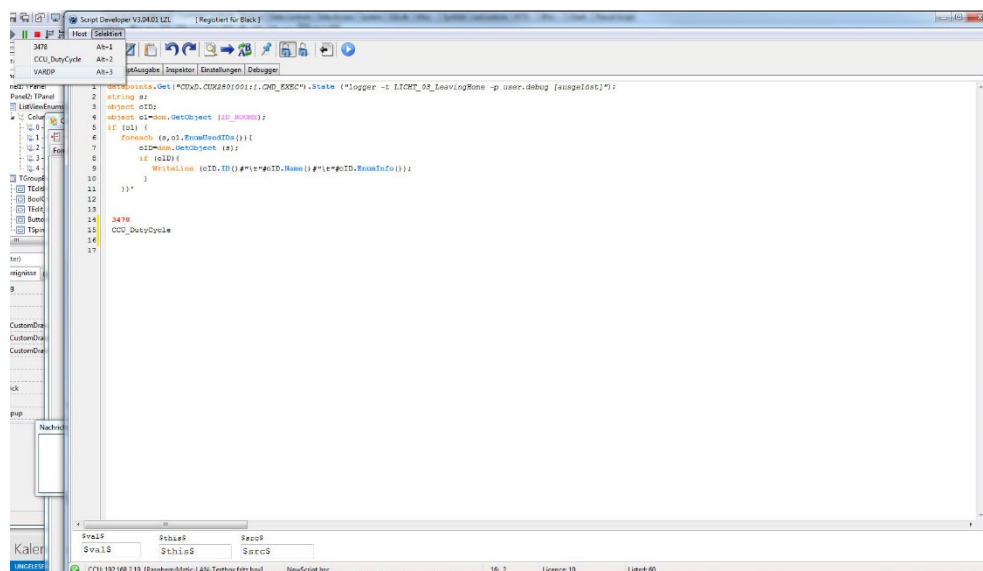
Zur schnelleren und auch möglichst fehlerfreien Bearbeitung besteht die Möglichkeit, Daten aus dem Inspektor direkt in den Editor zu übernehmen.

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

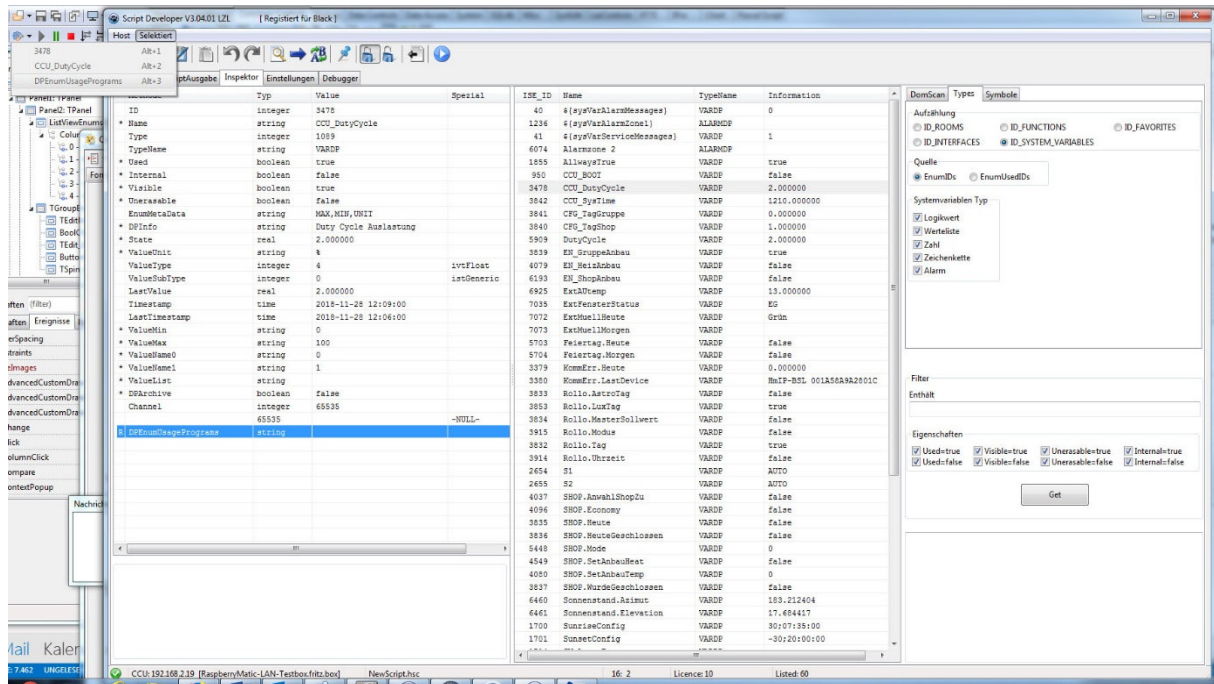


Unter selektiert sind die Eigenschaften herausgefiltert worden und lassen sich im Editor entweder durch das Menü selektiert oder durch die Kurztasten Alt-1: ID, Alt-2: Name und Alt 3: Eigenschaft bzw. Methode einfügen.





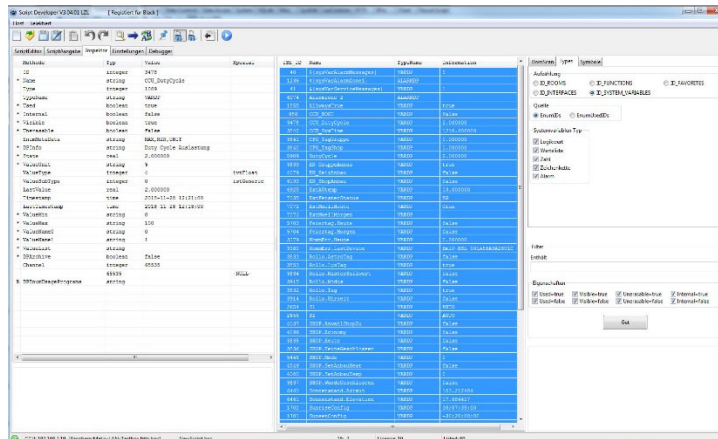
Im Detailauswahlfeld wird bei klicken auf die Methode auch noch der Methodename gespeichert, der sich dann auch durch Alt-2 einfügen lässt



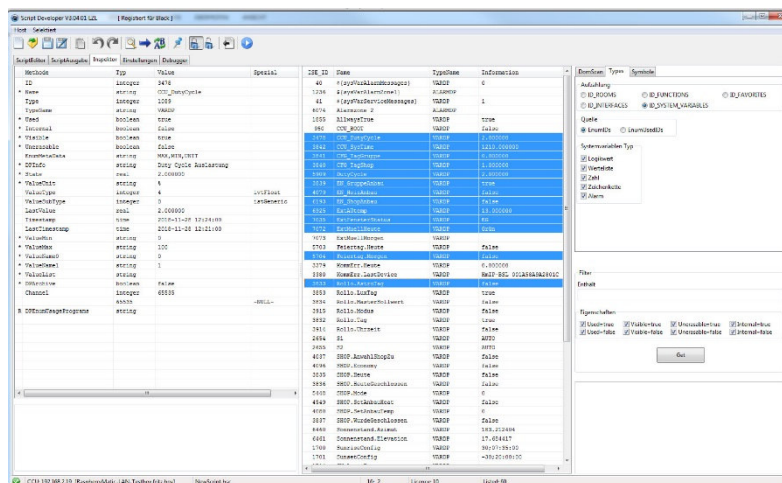
## 4.4.1 Mehrfachauswahl als Enum String



Es lassen sich im Hauptauswahlfeld Mehrfachselektionen vornehmen.

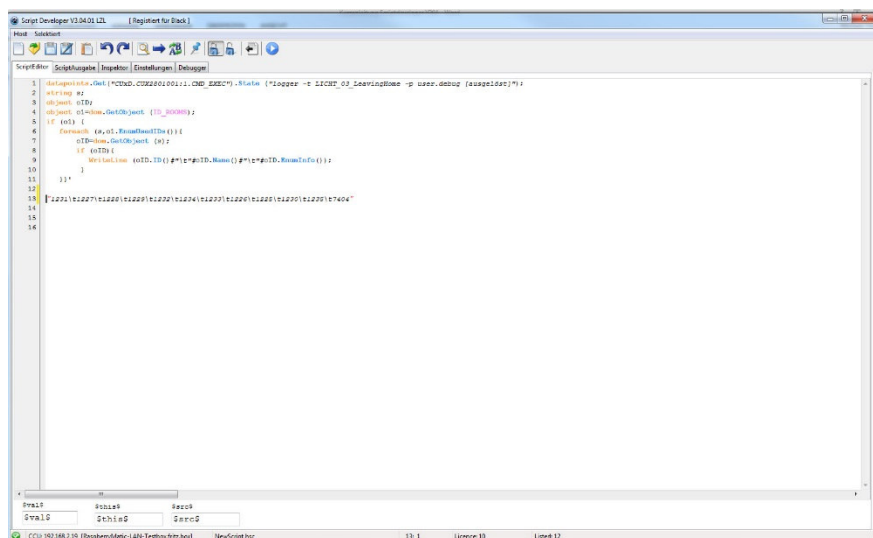
Ctrl-A : alle auswählen.



Oder die Übliche Mausbedienung:



Mit der  Taste merkt 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,

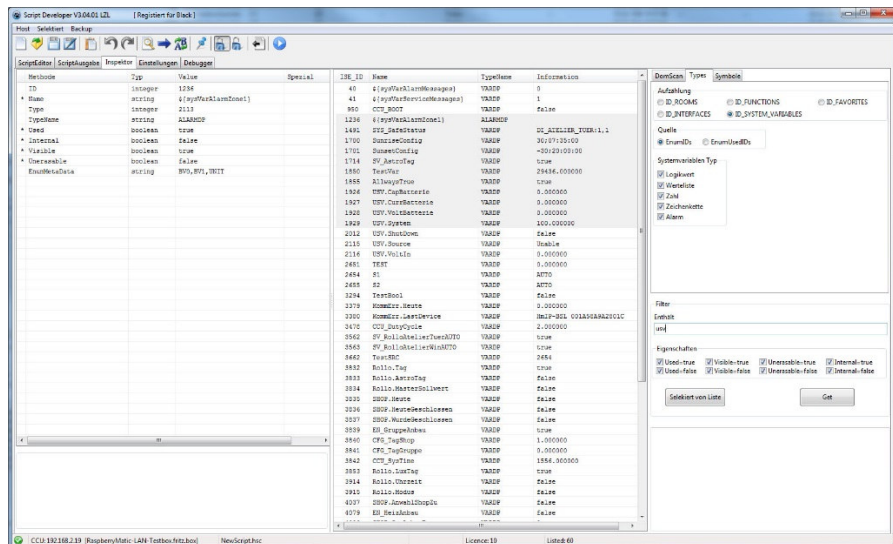


dann befinden sich die Selektierten Elemente wieder im Mittleren Feld.

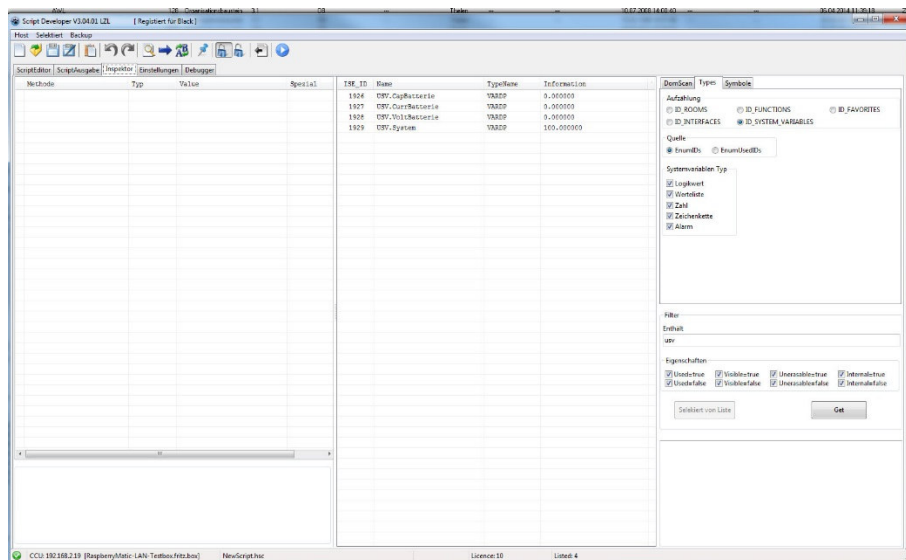
## 4.5 Selektion von Selektion

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

Hier Beispiel



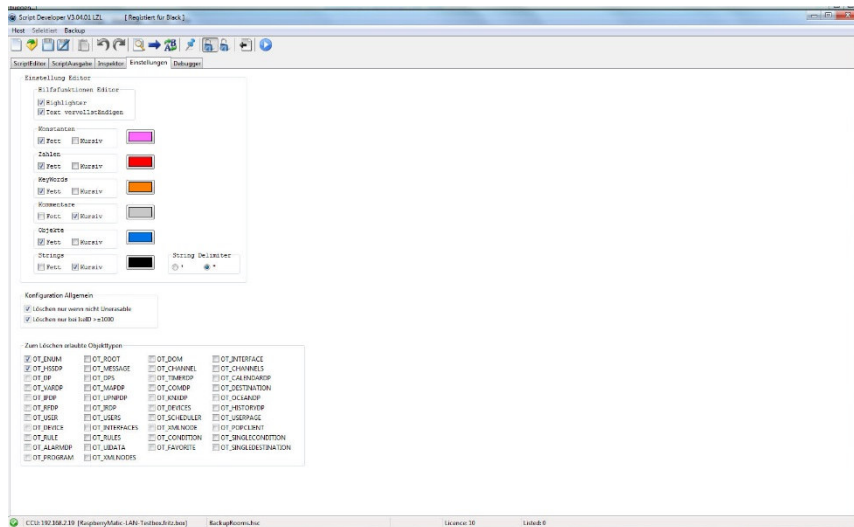
Selektierter Bereich von Systemvariablen, die hier darauf gefiltert werden sollen, dass der Name den String „usv“ enthält. Es müssen 4 Sysvars gefunden werden, die IDS 1012,2115,2116 werden hier nicht berücksichtigt, da diese sind selektiert sind. Bei <Druck auf: Selektiert von Liste: ergibt sich dann



## 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 unerassable. (unlösbar) Ist der Haken gesetzt, geht löschen nur wenn das Objekt nicht auf unerassable = checked steht. Um nicht löschbare Elemente zu löschen entweder:

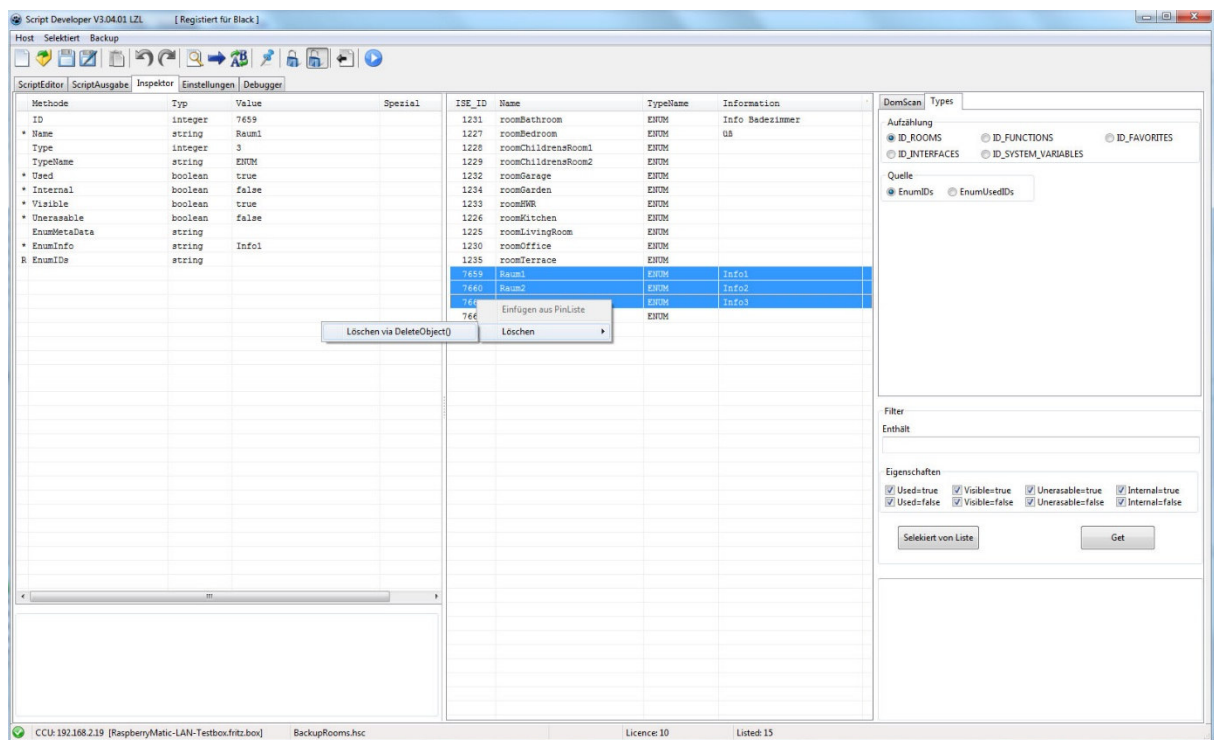
Im Inspektor unter Detailsview die Property entfernen (gilt nur für das Objekt), oder hier den Haken wegmachen (gilt für alle)

Löschen nur wenn ID >= 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.

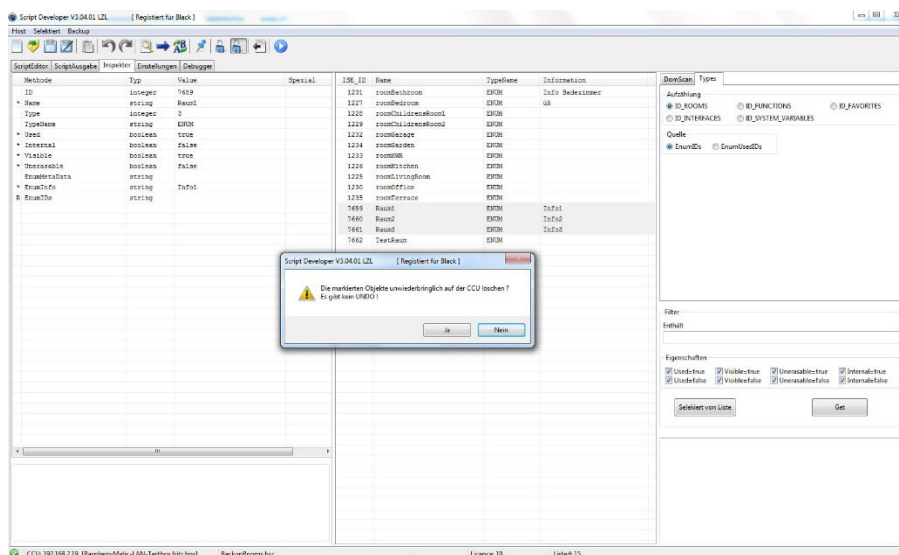
Löschbare Objekttypen. Die Letzte Sicherheit: ein zu löschendes Objekt muss einen hier gechecked Objekttyp haben, sonst wird es nicht gelöscht.

Löschen läuft so ab:



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

Mehrfachselektion ist möglich



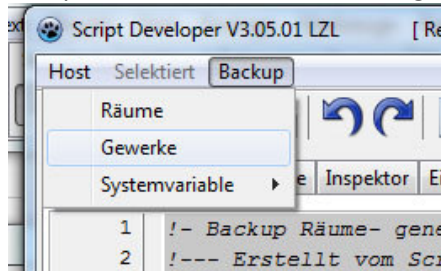
Nach dieser Sicherheitsabfrage sind die Objekte dann weg.. Zurück geht's dann nur mit restore.

## 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 angelegt, 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 angelegt, 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 Alarmer immer scharf, auch wenn dieser Alarm vorher im Alt System über AlArm (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)