Kurzanleitung Script-Developer SDV V5.02.01A (15. September 2023)

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

Die Software läuft auf Windows PC, ist bei nicht kommerzieller Nutzung Freeware.

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)

Inhalt

1 Installation

1.1 Historische Entwicklung

Bekannter weise wurde im Jahre 2017 sämtliche Programmversionen des damals verfügbaren Editors / Analyseprogrammes durch den Entwickler entfernt und waren mit der Argumentation, „(…) das Forum hätte es nicht verdient (…)“, nur noch einen elitären Kreis zugänglich. Einige User und ich hatten allerdings die Möglichkeiten erkannt, die sich durch eine gute Aufbereitung und vor allem Darstellung der Regadom Informationen ergaben. Da ich nicht unbedingt der Anhänger von monopolisiertem Wissen bin, begannen die Arbeiten an Alternativen, Ende des Jahres 2017 war ein ProofOfConcept erfolgreich gelaufen. Dieser SDV V1 bestand nur aus einem Editor, und einem primitiven Inspektor als reinem Textausgabefeld schnell an einem Abend coded in Scite. Die Schwächen dieses Inspektors zeigten sich recht schnell, keine Flexibilität. Es folgte der zweite Ansatz, der SDV V2, erstmals zu Jahresbeginn 2018 auf dem West-Stammtisch. Diese Frühform besaß schon einen Editor, und einen 2 spaltigen Inspektor für die Listen und die Detailansicht, ein Konzept, dass sich bis heute bewährt hat. Auch frühe Versionen dieses V2 konnten schon in der Anfangszeit Standartelemente der Rega analysieren sowie Skripte aus Programmen extrahieren und im Editor bearbeitbar machen. Intern war das Konzept, dass die RegaHss durch Skripte nur Rohdaten lieferte, die von dem SDV dann in Hochsprache aufbereitet und visualisiert wurden. Schnell wuchsen die Möglichkeiten, man lernte aus den Ergebnissen, die man im Inspektor sah und untersuchte. Allerdings wuchsen auch die Wünsche und Anforderungen, zum einem die persönlichen, zum anderen aus dem Tester und Stammtischkreis und aus dem Forum. Zudem hatte die V2 auch einen entscheidenden Haken, eigentlich als Testkonzept unter der Skriptsprache AutoIT in Scite entwickelt, erreichte das Konzept schnell seine Grenzen, die Geschwindigkeit einer reinen nicht objektorientierten Skriptsprache, viele Virenscanner klassifizieren die ExecEngine von AutoIT als potentiell gefährlich (man kann damit auch wirklich Viren schreiben und böse auf dem TCPIP Stack rummachen wenn man’s kann). Jedenfalls stand hier ein Cut an.
Auswahl musste eine Hochsprache sein, mit der sich GUIs programmieren lassen, aber ebenso Server und Clients umsetzen lassen, die Multithreading beherrscht sowie einen extrem guten Inline Debugger mitbringt, die ebenso eine entsprechende fertige Komponentenbibliothek hat, hierbei fiel auch ein Augenmerk auf eine gute Editorkomponente. Und die Programmiersprache sollte ich schon beherrschen, Babylon erfinden für so ein Projekt wollte ich auch nicht. Also fiel VisualBasic raus. Blieben noch C und Delphi. Aufgrund seit 1983 kontinuierlicher Programmiererfahrung unter Borlands TurboPascal Versionen und dem Nachfolger Delphi fiel dann die Wahl auf Delphi, im Hinblick auf die Lizenzproblematik von Embarcadero aber nicht auf Delphi selber, sondern auf den Opensource Ableger Freepascal mit Lazarus als „Rapid Application Development“. Damit stand dann das Konzept fest. An einem Wochenende schon im Sommer 2018 entstand der noch sehr rudimentäre Grundzug des SDV 3 mit Editor, Inspektor, Debugger und Setup. Hier spielte die gewählte RAD ihre Vorteile aus, in der Komponentenbibliothek gab es einen vernünftigen Editor mit Highlighter und Completition Funktionalität und auch die ersten Testversionen fanden im Stammtischkreis und auch im Forum grossen Anklang. Um der Rega ihre Methoden zu entlocken wurde die RegaHss unter dem IDA pro Decompiler von Hexware einmal genauer betrachtet. Daraufhin existierten dann auch bei mir die vollständigen Methoden und Konstantenlisten. Es machte noch etwas Arbeit, diese zu sortieren und mit Versuch mach klug hinsichtlich ihrer Parametrierung zu untersuchen. Ab diesem Zeitpunkt wuchsen dann die Möglichkeiten des SDV exponentiell, neue Erkenntnisse flossen ins Programm ein, deren dargestellte Ergebnisse dann wieder zu neuen Erkenntnissen führen, quasi das Moore´sche Prinzip. Und mit jeder Neuerung wurde letztlich das Arbeiten mit der CCU einfacher.

Im Sommer 2020 kam dann bedingt durch die von mir hochbejubelten Braking Changes der Rega der SDV in der Version 4. Das reine codebasiere Konzept kam da allerdings beim Editor an seine Grenzen, Hilfestellungs-Hints oder gar Pre-Syntaxanalyse waren so nicht allgemein umsetzbar. Dafür bräuchte es bekannterweise eine schnelle Codebasierte Datenbank und so wie es FPC in seinem Editor umsetzt, einen Analyse Backgrund Task, der die syntaktische Vor-Analyse macht. Hinzu kamen noch ein paar Unbekannte aus dem Syneditor selber, welcher ja nicht unbedingt ausführlich dokumentiert ist. Die Hürde mit der Dokumentation löste sich mit der Übersetzung der „Biblia de la Synedit“ aus dem spanischen mit Hilfe eines spanisch sprechenden Arbeitskollegen. Ab da ging es dann drastisch schneller, es folgten Foldings, Sprungmarken etc. Den letzten Ansporn, den Editor und den Highlighter auf Datenbankbasis umzustellen bekam ich durch die vollmundige Behauptung, es gäbe nur einen „vernünftigen“ Editor, und der SDV Editor sei eh Schrott- nun denn, im Spätsommer verbrachte ich berufsbedingt so einige Zeit in den Terminals verschiedener Flughäfen und so entstand die Grundlage der Datenbank basierten Version während der Wartezeit auf den Flughäfen Wien, Danzig und Bukarest. Damit ließen sich dann das Highlighting und die Syntaxprüfung realisieren und quasi als Abfallprodukt, da die Datenbank ja nun eh schon da war, gab es noch die ultrageheime Skriptdoku. So geheim ist die nun wirklich nicht, und die Anzeigeaufbereitung aus der Datenbank brauchte maximal 200 Programmzeilen.

Einen weiteren größeren Schritt im Editor gab es im Februar 2022 mit der Einführung des LL1 Parsers/ Lexers erstmal nur zur genaueren Methodenvorhersage bei der IntelliSense Funktionalität. Diese auch als hoch geheim und unendliche schwierig eingestufte Funktionalität, das übliche Gepolter halt, diese Algorithmen sind so ultrageheim, die wurden schon zu meiner Studienzeit Anfang der 1990er Jahre in den Vorlesungen zum Thema „Compilerbau“ gelehrt. Und die unendliche Schwierigkeit bei der Programmierung war eigentlich nur – meine alten Vorlesungs-Skripte auf dem Dachboden zu finden.

Es folgten bis Juni 2023 nur partielle Verbesserungen / Erweiterungen begründet darin, dass ich eigentlich Möglichkeiten zur Erweiterung der WebUI schon im Jahre 2023 aufgezeigt hatte. Mich störte es immens, dass die WebUI kein Drag& Drop hat, dass sich Regeln mitten in einem Programm weder einfügen, verschieben noch löschen lassen, ohne dass dieses einem Neuschreiben des Programmes gleichkommt. Sicher, es gab Möglichkeiten dieses umständlich über Skripting im Zusammenspiel mit dem Inspektor zu machen, aber Normal-Anwendertauglich ist etwas Anderes. Ansätze dazu hatte ich. Damit es aber auch für die WebUI tauglich würde, hätte es ein paar zusätzliche Methoden bedurft, worauf ich schon 2020 ein Issue mit jährlicher Wiedervorlage im entsprechenden Github eingereicht hatte. Es tat sich in der Richtung allerdings nicht, null, niente.
Also ist mir irgendwann mal der Kragen geplatzt, an einem mal wieder Warte Aufenthalt an einem Flughafenterminal (das scheint sehr produktiv für die Entwicklung zu sein), habe ich angefangen den Programmedit des SDV umzusetzen. Da ja kein Interesse bestand, dieses auch WebUI tauglich auch in der CCU direkt umzusetzen, standen mir ja alle Wege offen und ich konnte Programm und Komponententechnisch alle Register ziehen ohne Rücksicht auf Strukturen und Gegebenheiten der WebUI nehmen zu müssen. Er ist noch nicht ganz fertig, erlaubt aber jetzt schon eine Vielfalt an Funktionen die es so in der WebUI nicht gibt (und höchstwahrscheinlich bei gegebener Struktur auch nicht geben wird)

Welche Schwerpunkte ich beim SDV setze lässt sich grob aus der Anzahl der Programmzeilen für die einzelnen Funktionalitäten ableiten. Die Version 5.02.01 hat etwas über 81000 Programmzeilen (81000 von mir geschriebene Codezeilen, ohne Kommentare und die von Lazarus schon in der RAD implementieren Komponenten (Syneditor, SynEditHighlighterFoldBase als Highlighter, alle visuellen Komponenten, crypto units und Netzwerktools nicht mit eingerechnet) auf die 81000 Zeilen verteilen sich funktional (aufgeteilt, stellenweise überschneiden sich auch cases). Mit der Version 5.02.xx ergibt sich folgende Aufteilung von Programmzeilen zu Funktionen.

Ca. 500 Zeilen - Allgemeines (Load / Save von Einstellungen, Fenster Splits)
ca. 1000 Zeilen - CCU Kommunikation und Auswertung, Thread und Exception handling
ca. 600 Zeilen - Crypto und Licensing
ca. 1400 Zeilen - Child der Synedit Class incl. der Completion (benutzt die Datenbank)
ca. 850 Zeilen - Child der SynEditHighlighterFoldBase
ca. 518 Zeilen - Multi Tab Editor
ca. 4800 Zeilen - Code Datenbank und deren Initialisierung (Methoden, konstanten, Syntax, Beschreibung)
Ca 1800 Zeilen – der Tokenizer und der LL1 Parser
Ca 1000 Zeilen – Die RegaAbstractionUnit
ca. 24000 Zeilen - Inspektor (Childs von TListView) mit allem was da so zugehört, wie Paramsets, Direktverknüpfungen
ca. 400 Zeilen - Favoriten
ca. 200 Zeilen - Systemprotokoll
ca. 400 Zeilen - Setup
ca. 200 Zeilen - Die Skriptdoku (ist ein Abfallprodukt der Code Datenbank)
ca. 6000 Zeilen - CCU Services
ca. 20000 Zeilen - Die Verschiedenen unscheinbaren Backups, die der SDV ermöglicht (Programme, Geräte, WebUi etc.)
ca. 13000 Zeilen – Der SDV Programmeditor (in der Form Stand August 2023)
ca. 5500 Zeilen - Der große Punkt Gerätekopieren (Heizprofile zwischen HM und HmIP, Wochenprogramme, Gerätetausch) nur die separaten Zeilen, setzt auf classes der Backups auf)
ca. 5000 Zeilen (nicht genau zuordbar, allgemeines, grundlegende Classes und Deklarations, JSON Interpreter, die von vielen Punkten gebraucht werden, interner Debugger)

1.1 Installation des SDV

Das *.rar File in ein beliebiges Verzeichnis entpacken. Ein Installer ist nicht notwendig. In diesem Verzeichnis befindet sich auch das Konfigurationsfile SDV4.INI. Bei der erstmaligen Verwendung muss dieses angepasst werden

```
[LAST]
DATEI=c:\MTH\Homematic\NewScript.hsc
; Zusätzliche Filetypen zur Filterung
; *.BMP;*.JPG;*.GIF
; die Aufzählung muss mit Semikolon getrennt sein
FILETYPE=                           Beispiel *.htm;*.scr
; Scriptdir
; ist Scriptdir leer, so wird Standartmäßig das Verzeichnis
; genonnem, in dem der SDV gestartet wird
SCRIPTDIR=                          Beispiel c:\Scriptdir\

[HOST]
NICKNAME=DerNickNameAusDerLizenzanfrage
CUXD=CUxD.CUX2801001:5
SYSTEMEXEC=true                     Aufruf wenn möglich über systemExec statt CuxD
LICENCE1=DerLizenzschlüssel1
LICENCE2=DerLizenzschlüssel2
SSHPLINK=true
THREADKILL=15000
THREADKILLXML=20000


[CCU1]                              Neuer Schlüssel
IP=192.168.2.19                     Ip Von CCU1
USERNAME=ExternAdmin                Nutzername auf der CCU1
PASSWORD=XXXXXXXXXX                 Passwort der Nutzers auf der CCU1
USEHTTPS=true                       Zugriff über HTTPS
SSHUSERNAME=root                    root wenn SSH Zugriff erwünscht
SSHPW=xxxx^                         Das SSH Passwort der CCU eintragen
BACKUPDIR=C:\PfadderCCUWoDerSDVBackupsAblegenwird\


[CCU2]                              Dito wie bei CCU1
IP=192.168.2.6
USERNAME=
PASSWORD=
USEHTTPS=false
SSHUSERNAME=root                    root wenn SSH Zugriff erwünscht
SSHPW=xxxx^                         Das SSH Passwort der CCU eintragen
BACKUPDIR=C:\PfadderCCUWoDerSDVBackupsAblegenwird\



[HOSTCCU]                               Die Werte der aktuellen CCU bei Start
IP=192.168.2.19                         im Idealfall hier die Daten von CCU1 oder 2
                                        eintragen für 1. Start
USERNAME=ExternAdmin
PASSWORD=XXXXXXXXXX
USEHTTPS=True
SSHUSERNAME=root                    root wenn SSH Zugriff erwünscht
SSHPW=xxxx^                         Das SSH Passwort der CCU eintragen
BACKUPDIR=C:\PfadderCCUWoDerSDVBackupsAblegenwird\



[SECURITY]                              Die Ports die der SDV benutzt
HTTPREGASCRIPT=8181
HTTPREGAXMLRPC=1999
HTTPSREGASCRIPT=48181
HTTPSREGAXMLRPC=41999
```

[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

Wie bekommt man das ganze nun in Funktion

Hinweis für CCU2 Nutzer oder Nutzer älterer Firmware: Wenn sich kein Anfrageschlüssel generieren lässt (keine Hashs erkannt), in der INI Datei prüfen, ob dort der entsprechenden CCU der Eintrag USEHTTPS=True steht. Damit klappt es nicht. In diesem Fall muss der Eintrag in USEHTTPS=False geändert werden, dann klappt auch.
Alte Firmwares mögen es auch nicht, wenn über Nutzername und PW ein authentifizierter Zugriff über Name:PW@Host versucht wird. Bei älteren Firmware oder CCU keinen Nutzernamen oder Passwort eintragen bei CCU1 / CCU2 CCUHOST

Bei CCU2:
Auf jeden Fall USEHTTPS=false sowie USERNAME= und PASSWORD= (Username und Password leerlassen, sonst gibt's keine HashAnfrage

Bei CCU3:
Bei Authentifizierung auf CCU3 ein: USERNAME und PASSWORT des Admins müssen eingetragen sein

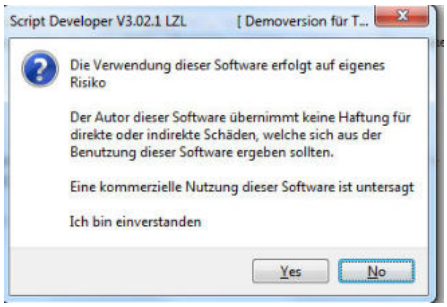
Und auch wichtig, dieser Fehler kommt auch oft vor: Der Nutzername darf nicht Testuser sein !
Bitte dann in der INI ändern, die INI speichern, SDV neu starten und neuen Anfragecode generieren.

1.3 Lizensierung

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.

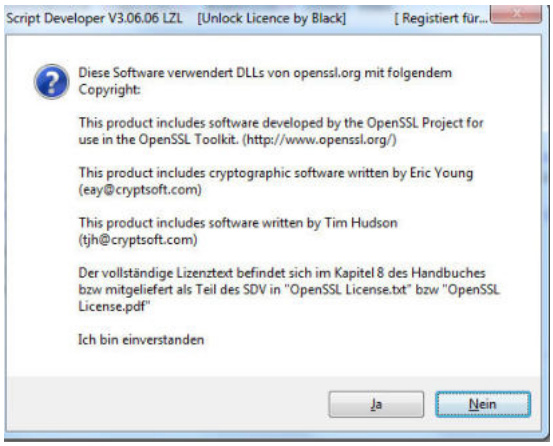
Wie arbeitet das ?
Der SDV telefoniert nicht nach Hause. Um eine Lizenz anzufragen ist folgender Weg einzuschlagen.

1. Die Konfigurationsdatei SDV4.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



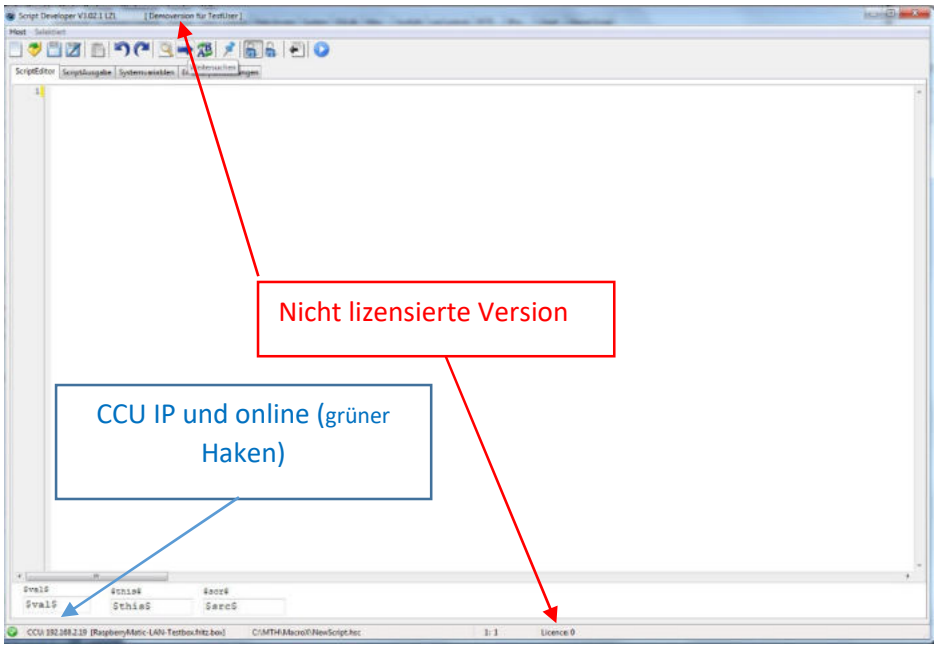
Bei allerersten Start muss dieses Fenster mit yes bestätigt werden. No führt so einem sofortigen Programmabbruch

Ab der Version 3.06.06 befinden sich in dem Package des SDV 2 DLL´s von OpenSSL.org, welche für den HTTPS Zugang zur CCU benötigt werden. Hierbei ist einmalig ebenfalls eine Zustimmung über die Verwendung der Software nötig. Der komplette Lizenztext kann im Kapitel 9 nachgelesen werden bzw. ist als Datei mit in dem Package enthalten.

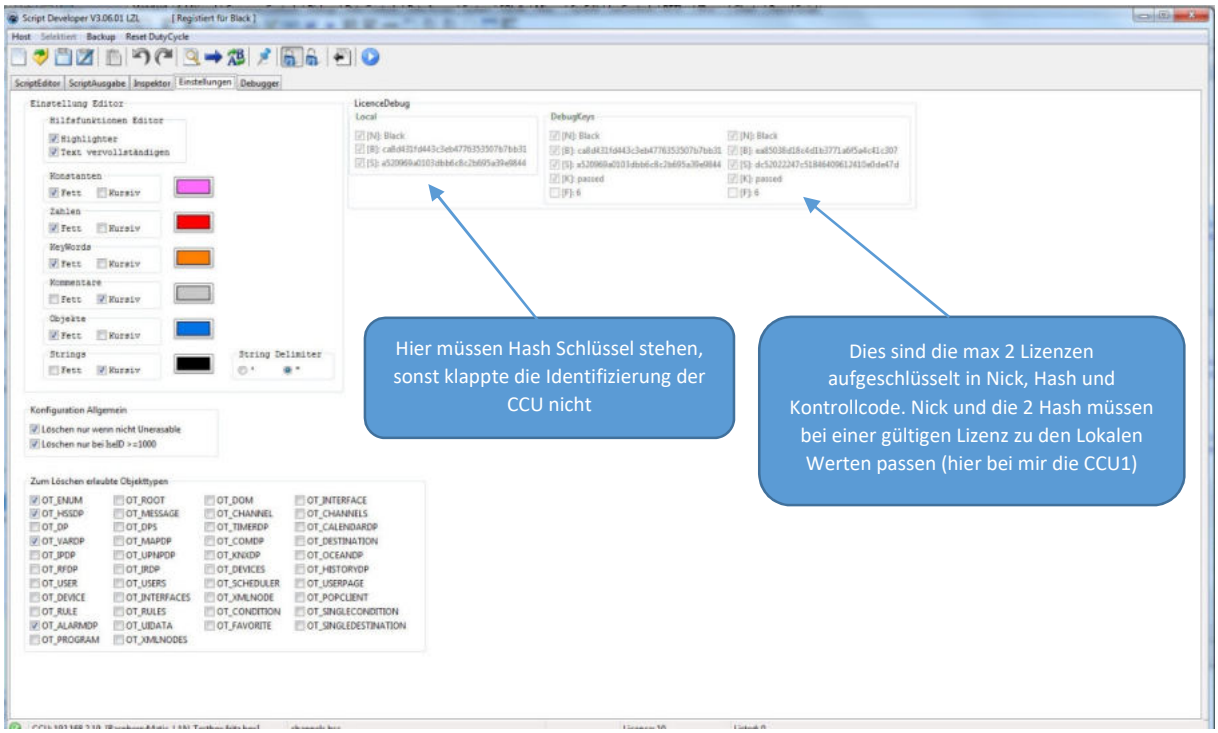


„Nein“ führt auch hier zu einem sofortigen Programmabbruch

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



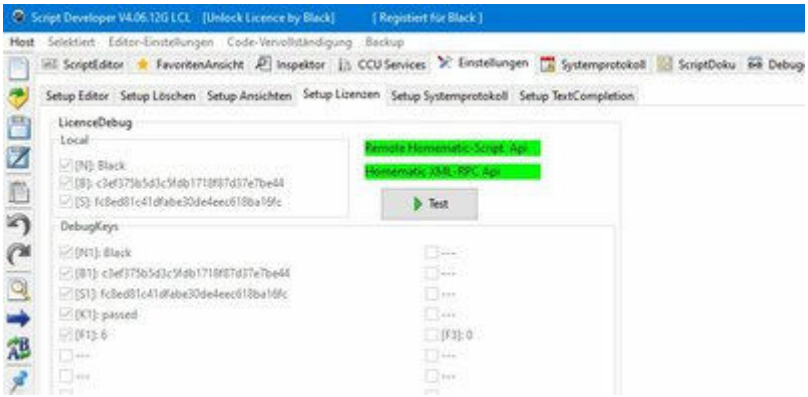
Wenn die CCU, für die der Schlüssel angefragt werden soll, als grün angezeigt wird, bitte vorher einmal unter dem Reiter Einstellungen kontrollieren



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

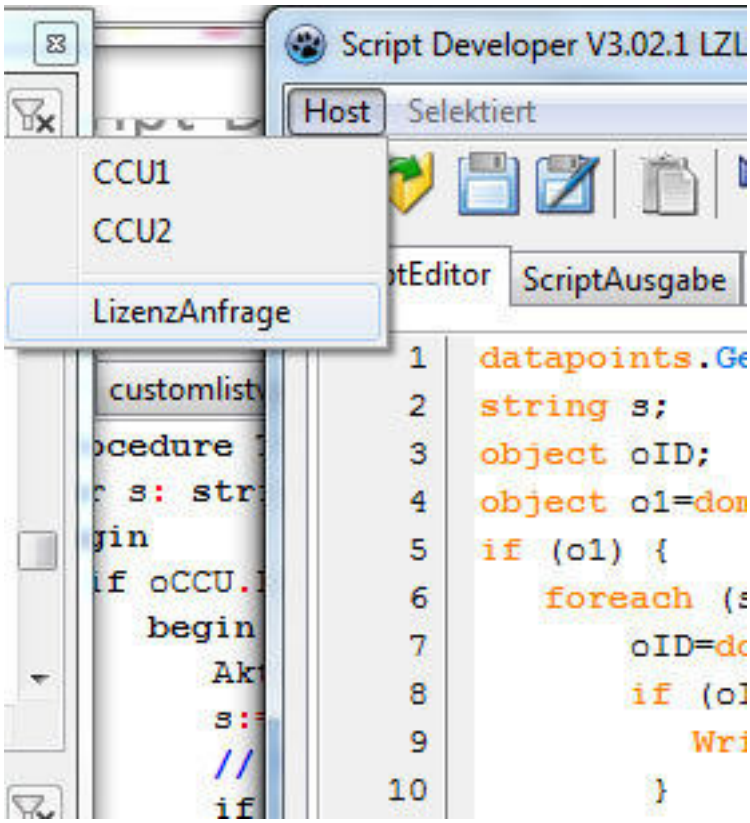
Im Auslieferungszustand ist unter NICKNAME= noch TestUser eingetragen. Mit TestUser lässt sich kein Key generieren. Hierbei dann bitte die INI nochmal anpassen und den SDV neu starten-

Im Service unter Debug Lizenzen Möglichkeit zum Test der Verbindung programmiert. Also bitte erst Anfragecode generieren, wenn der test 2 mal grün ergibt. wenn nicht, stimmt etwas in der Konfiguration der Firewall bzw. wenn Authentifizierung vorgegeben ist, in der Authentifizierung nicht. Das sehe ich dann spätestens im Licencer von mir

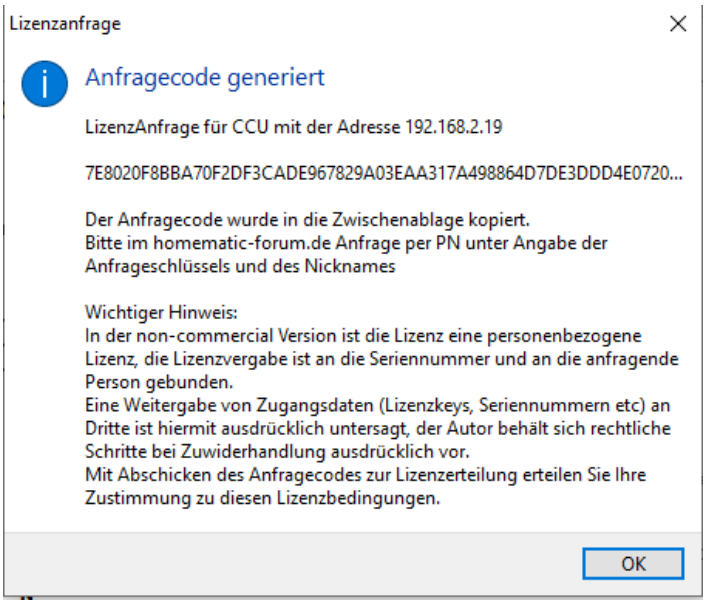


Den Punkt gabs zwar schon seit ein paar Versionen. Gelb bedeutet: Test läuft grade, rot: war ein Griff ins Klo, grün: tuts

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.

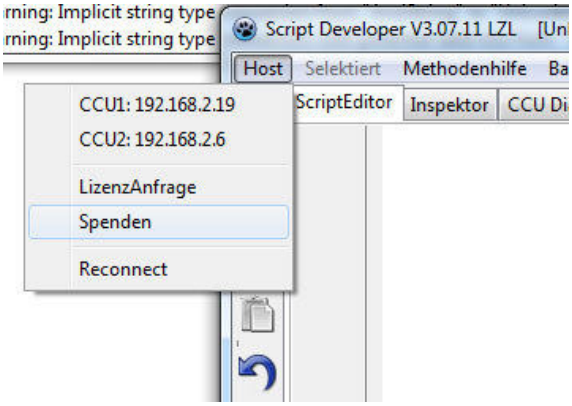
Geplant hab ich folgende Lizenzabstufungen

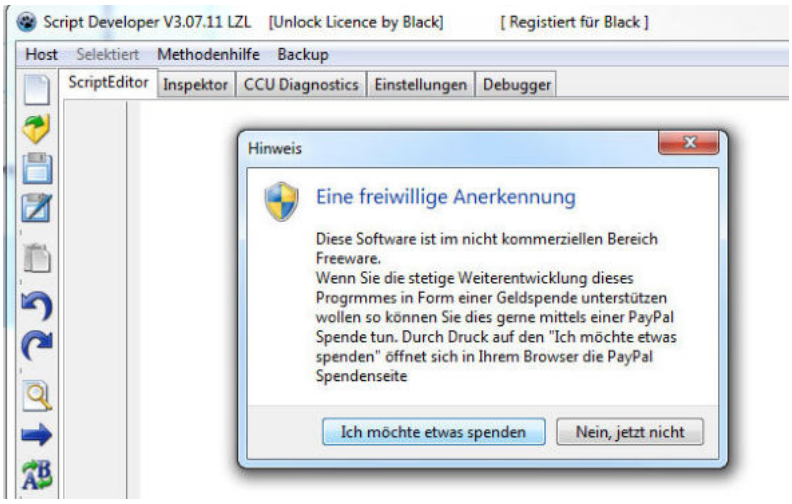
Level	Editor	Skript ausführen	Highlighter Und Vervollständiger	Enums	SysVar	Programs	Backup Restore		Special Funktions (resetDC, restor950 etc
0	X								
1	X	X							
2	X	X	X						
3	X	X	X	X	X				
4	X	X	X	X	X	X			
5	X	X	X	X	X	X	X		
6									
7	X	X	X	X	X	X	X		X

Ach so, noch ein Tipp an denjenigen, der versucht, den Code für meine Unlock-Licence in dem Code zu finden…. Eher findest du den Heiligen Gral, das ist eine Compiler Direktive, die beim Kompilieren an x Stellen anderen Code erzeugt bzw. manche Punkte und Funktionen vom kompilieren ausnimmt und keine einfache Variablenabfrage zur Laufzeit… so als Tipp halt ^^

1.4 Donate

Es besteht die Möglichkeit, dem Autor dieses Programmes für seine Arbeit eine Spende über Paypal zukommen zu lassen. Diese Möglichkeit ist völlig losgelöst von der Vergabe von Lizenzschlüsseln und rein freiwillig.





1.5 Systemvoraussetzungen

Mittlerweile wird der SDV V4 unter einer 64bit Umgebung kompliert. Getest und Lauffähig ist die Anwendung unter Win10/64bit, Win7/64bit sowie unter entsprechenden VMs unter Linus bzw Apple Umgebungen

Auf der Homematic-Seite wurde bei mir auf einer Raspberrymatic 3.65.11 getestet.

Aufgrund der breaking Changes der 3.51.6.20200621 gilt Folgendes:

Ab Rega version der 3.51.20200621 arbeitet der SDV V4 komplett, in älteren Versionen sind die Backup Funktionalitäten, die Gerätekopien und alles, was mit den Schnittstellenprozessen direkt arbeitet, disabled bzw ausgeblendet. Wird dieses benötigt, so muss die Last Legacy Edition der SDV V3 eingesetzt werden.

Eventuell werde ich im Laufe des Jahres 2021 mal schauen, ob ich es nativ unter Debian auch kompiliert und zum Laufen bekomme. Wahrscheinlich wird dieses Projekt aber zurückgestellt, da leider das Threadhandling unter Linux sich völlig anders wie Windows verhält.

Windows11:

Es gibt Stand November 2022 43 User, bei denen der aktuelle SDV problemlos läuft und einer, wo wohl massive Probleme auftraten. Ursachen und Hintergründe kenne ich noch nicht, da ich noch nicht Windows11 einsetze.

1.6 Was tut es bis jetzt

Der Editor funktioniert inkl. Suchen und Suchen / ersetzen. Der Highlighter und der Code Vervollständiger sowie Code folding arbeiten auch.

Undo / Redo arbeiten
Skript 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
Datenpunkte
Kanäle
Programme
Favoriten
User
Paramset Master
Paramset Links
Skript Bearbeitung aus Programmen heraus
Verändert von Verzögert um von Skripten
Editor mit bis zu 32 SkriptTabs (Früher 16, ab 5.02 32 Tabs)
Verändern von Retriggern
Komplettes Aufschlüsseln von Programmen in Rules, Subrules, Conditions, SingleConditions, Destination und SingleDestination
MarkupUp im Editor
Suchen in Skripten nach Namen von Systemvariablen, Devices, Channels, Rooms und Functions
Suchen in Skripten nach Seriennummer von Devices und Channels
VollTextsuche in Skripten auch mit Regex Ausdrücken
Backup von Räumen, Gewerken, Favoriten, Systemvariablen, Devicenamen und Programmen
Backup von Mastersets
Backup von LinkSets
Backup von WebUI Einstellungen eines Gerätes (Zuordnung zu Räumen, Gewerken Favoriten und Programmen)
Diverse Konsistenstests
Auflösen von Direktverbindungen
Backup von WEBUi Programmen und Einstellungen eines Gerätes
Devicekopien auch unter „ähnlichen“ Geräten
Selektives Kopieren von Linkparametern zwischen Direktverknüpfungen

1.7 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.~~ (Edit: mittlerweile geändert)

~~Folding im Editor arbeitet noch nicht. Wenn der Rest läuft gucke ich da mal nach.~~ (Edit: mittlerweile umgesetzt)

~~Kommentare im Skript 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~~ (Edit: mittlerweile umgesetzt)

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. (Edit: History ist mittlerweile programmiert, Diagramm gibt es aber nicht und wird es auch nicht geben)

Linkset Restore von Heizungsgruppen arbeitet noch nicht sauber und ist deshalb erstmal disabled.

~~Completion greift bei der Erkennung schon mal gerne ins Leere~~ Ist Stand Nov. 2022 auch gefixt

Der Programmeditor ist noch nicht abschließend fertig, bei der Entwicklung habe ich darauf Wert gelegt, zuerst die Funktionen zu implementieren, die die WebUI nicht hat. Was jetzt im SDV Editor noch fehlt, kann die WebUI. Bis zur finalen Version müssen da Programm Editor und WebUI zusammenarbeiten

1.7 Geänderte Anforderung an Auflösung

Nach Rückmeldung aus dem Testerkreis bezüglich dem Herabsetzen der nötigen Bildschirmauflösung habe ich dieses etwas neu aufgebaut.

Der SDV startet mit Höhe 769 und Breite 1300 und kann angepasst werden. Die Einstellungen werden beim Verlassen gespeichert.

Mit dem Kommandozeilenparameter SDV_xxxx formreset kann eine Bildschirmauflösung wieder auf den Grundzustand zurückgesetzt werden. Alternativ die Einträge unter dem Schlüssel FORMVIEW in der INI Datei löschen.

1.7.1 Mehrfacher Start

~~Der SDV kann mittlerweile mit mehreren Instanzen Gleichzeitig gestartet werden. Dabei hat allerdings nur die zuerst gestartete Instanz Schreibrechte auf die INI Datei. So lassen sich mehrere Skripte gleichzeitig bearbeiten und testen oder auch gleichzeitig mehrere Inspektoren benutzen~~.
Die frühere Möglichkeit des mehrfachen Starts ist mittlerweile wieder abgeschafft worden, durch die Einführung des Multi Tab Editors sowie der Interprocess-Communikation wird ein durch Doppelklick auf ein Skript 2 fach gestarteter SDV keine zweite Instanz öffnen, sondern seine Startparameter via IPC an den Server des Hauptprozesses übertragen und dann wieder beenden. Die gewünschte Datei wird dann im EditorTab des 1. SDV geöffnet und kann dort bearbeitet werden

Zu empfehlen ist auch, zB. die Dateiendung .hsc mit dem SDV zu verknüpfen, so öffnet ein Doppelklick auf eine Skriptdatei dann automatisch den SDV mit dem geladenen Skript

1.8 Changelog V5.x.x

1.8.1 Changelog V5.01.02C

Lazarus RAD aktualisiert auf V2.2.4
Programmeditor begonnen
Central Links lassen sich löschen
Anzeigen von Ghosts auch in der Listenansicht
Im Editor Möglichkeiten nicht druckbare Zeichen anzuzeigen
Automatischer Vorschlag der Skriptvariablen beim Schreiben nach einstellbarer Zeichenanzahl
Reparaturmöglichkeit für verloren gegangene Channels bei den internen Systemtasten
Einige Verbesserungen und Bugfixes
Änderung des Scan-Algorithmus für DomScan und interne Reparaturläufe
Vergleichslauf Paramset in Paramset Description

1.8.2 Changelog V4.09.05C

Lazarus RAD aktualisiert auf V2.2.0
Markierte Listen in der Detailansicht des Inspektors lassen sich mit CTRL-C kopieren und z.b. in Excel einfügen
Filtermöglichkeiten im Inspektor erweitert (Maincondition etc)#
Zeitmodule lassen sich nun komfortabel in einem eigenen Menü ändern und aktivieren
Systemprotokoll lässt sich im SDV nun ausdocken und z.b. Auf einen zweiten Bildschirm legen
This,src und value nun für jeden Skripttab Separat
Completion von Skriptvariable wesentlich verbessert

1.8.3 Changelog V4.09.01

Verbesserte Algorithmus zur Intellisense Methodenvorhersage.
Verschiedene ZusatzMenüs wie:
Crossreferenzliste Skriptvariablen
Auswahlmenü mit Automatischer Erzeugung von vollqualifizierten Zugriffen
Verbesserungen im Highlighter und der Syntaxerkennung
Verbesserungen im Inspektor (zusätzliche Menüs und Veränderbarkeit von Objekten)

1.8.4 Changelog V4.07.03E-HF1

Verbesserungen am Editor (Block EIn/Ausrücken)
Umfassen Funktionalität im Editor
darstellbarkeit im Editor von {} und () Bereichen
Programme reparieren/Kopieren Funktionalität
Verbessungen im Inspektor zur Werteänderung an normalen Datenpunken / eigenlich nicht beschreibbaren Datenpunkten
Inspektor Möglichkeit Master,LinkSets zu verändern
Kommentarmöglichkeiten für alle Objekte eingefügt
Direktes Editieren von Metadaten
Komfortables Editieren von Metadaten
Verschieben von Link Paramsets in andere Direktverknüpfungen
Verbesserte Löschalgorithmen
Wesentliche Verbessungen von Hintinformationen von Master / Linksets
EInfache Testmöglichkeit, ob die Anfrageeinstellungen in Ordnung sind (2 mal grüne Ausgabe)
Inspektor verbessert, damit dieser auch mit Ghost Objekten klarkommt und diese darstellen kann
idarray: Reihenfolge eingehängter Objekte einfach verschiebbar
Skriptdoku weiter verbessert
Inspektor Hint Informationen wesentlich erweitert
Zyklische Abfrage des Systemprotokolls implementiert
Inspektor wesentlich schneller gemacht: Optimierung Classen und Umstellung auf direkten Schnittstellenprozesszugriff

1.8.5 Changelog V4.05.03

Ab dieser Version Umstellung des Highlighters auf eine Datenbank. Damit bildet der Highlighter nun die vollständige Regasyntax ab

Codebasierte Methodendatenbank
vollständige Regasyntaxunterstützung des Highlighters
Echtzeit Systaxprüfung im Editor (Warn und fehler)
Überarbeites Code Completion
Komfortables Auto-Complete (Einfügen completter Code Schnipsel)

1.8.6 Changelog V4.02.07G

MultiTab Editor
diverse Bug Fixes

1.8.7 Changelog 04.01.10

Bugfixes
Erweiterte Hints
Bookmarks
Einführung des MultiTab Editors
Drag/Drop zum Kopieren von Parameters von Linkssets in einen gleichartigen anderen Linksets
Erweiterung des Highloghters
Eingeschrängt lauffähig auch auf alten Regas, hier aber ohne Backup und Gerätekopie

1.8.8 Changelog 04.01.01

Die Breaking Changes version, als v4 nur noch Lauffähig auf der neuen Rega.

Die alten V3 Versionen betrachte ich als obsolet, diese sollten auch nicht mehr eingesetzt werden.

1.9 Neue Authentifizierungsmechanismen Raspberrymatik

Die neuen Versionen der Raspberrymatik unterstützen SSL und Authentifizierung über Nutzername / Password. Nach Wunsch und Tips aus dem Testerkreis habe ich auch dies versucht umzusetzen.

1.9.1 SSL

SSL wird nun realisiert über die OpenSSL Library, welche auch von der Lazarus Foundation empfohlen wird:
Quelle: https://sourceforge.net/p/lazarus-ccr/s ... en_ssl.pas.
Der SDV liegt als 32bit Compilat vor, wenn man sich die dazu benötigten DLL´s selber herunterladen möchte, die Quellen sind folgende;

http://packages.lazarus-ide.org/openssl-1.0.2j-x64_86-win64.zip für die 64 Bit Version
und
http://packages.lazarus-ide.org/openssl-1.0.2j-i386-win32.zip für die 32 Bit Version

Das Zip File auspacken und die 2 DLL´s in das Verzeichnis kopieren, im dem sich auch die SDV.Exe befindet. In dem SDV rar File befindet sich auch die 32 Bit Version der DLL´s inkl. dem Lizenztext als TXT und PDF. Damit sollte der Zugriff über HTTPS schon mal funktionieren.

1.9.2 Authentifizierung über Nutzername / Passwort.

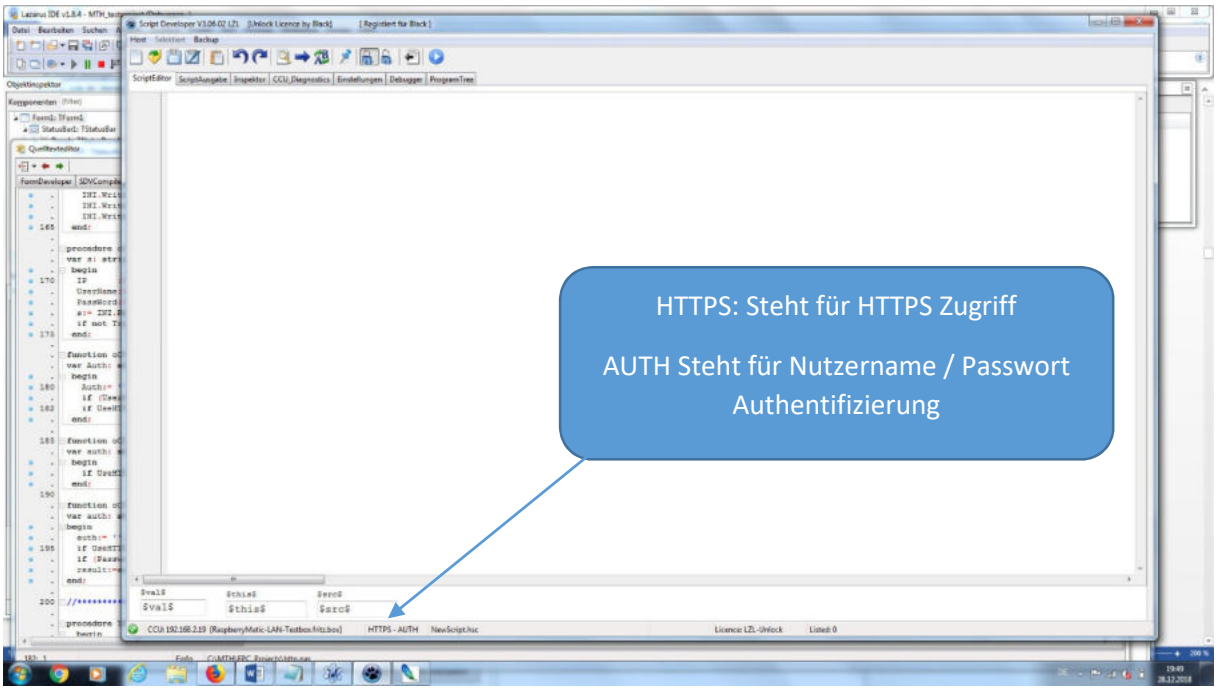
Die Struktur der INI Datei wurde geändert, damit sich pro CCU nun auswählen lässt zwischen HTTP und HTTPS Zugriff und die Authentifizierung Nutzername / Passwort wahlweise genutzt wird.

Wird bei Nutzername oder Passwort nix eingetragen, so generiert der SDV einen Zugriff ohne die Kennung NN:PW@HOST. Sind beide Werte eingetragen, so wird der Zugriff überNN:PW@HOST generiert.

Beispielhafter Aufbau der neuen INI

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

[HOST]
NICKNAME=DerNickNameAusDerLizenzanfrage
CUXD=CUxD.CUX2801001:5
SYSTEMEXEC=true
LICENCE1=DerLizenzschlüssel1
LICENCE2=DerLizenzschlüssel2

[CCU1]                                  Neuer Schlüssel
IP=192.168.2.19                         Ip Von CCU1
USERNAME=Admin                          Nutzername auf der CCU1
PASSWORD=XXXXXXXXXX                     Passwort der Nutzers auf der CCU1
USEHTTPS=true                           Zugriff über HTTPS

[CCU2]                                  Dito wie bei CCU1
IP=192.168.2.6
USERNAME=
PASSWORD=
USEHTTPS=false

[HOSTCCU]                               Die Werte der aktuellen CCU bei Start
IP=192.168.2.19                         im Idealfall hier die Daten von CCU1 oder 2
                                        eintragen für 1. Start
USERNAME=Admin
PASSWORD=XXXXXXXXXX
USEHTTPS=True

[SECURITY]                              Die Ports die der SDV benutzt
HTTPREGASCRIPT=8181
HTTPREGAXMLRPC=1999
HTTPSREGASCRIPT=48181
HTTPSREGAXMLRPC=41999
```

Die gelben Texte müssen auf jeden in einer Bestehenden INI geändert werden. Ist das alles richtig gemacht, so zeigt der SDV im Statusfeld auch die Art des Zugriffs an:



Ein Debugger Breakpoint im SDV zeigt hier den zusammengebauten HostString, der bei HTTPS und Authentifizierung benutzt wird

1.10 Ein Wort des Authors zum Thema „gewerbliche Nutzung“

Aufgrund der Tatsache, dass ich von einem Supplier versehentlich hörte, man freue sich schon auf die Backup Funktionalitäten der Direktverknüpfungen, Blöd nur, dass ich die kommerzielle Nutzung explizit untersagt habe. Immerhin zwingt mich diese kommerzielle Unverfrorenheit ein wenig an dem Ausgabeformat zu feilen. Das derartige Vervielfältigungsmechanismen einen immensen zeitlichen Vorteil bei der Implementierung von baugleichen Anlagen bieten leuchtet allen ein. Möglichkeit wäre diese Funktion dann ganz zu sperren oder per CompilerDirective erst gar nicht einzucompilieren. Hilft aber allen anderen privaten Anwendern nicht.
Also bleibt mir nur um das zu unterbinden bzw. möglichst zu erschweren:

Als Ausgabeformat für das Backup von Geräten wird es nur noch JSON geben.

Damit ist direktes 1 zu 1 Kopien ziehen aus einer Masteranlage zwar möglich, das JSON kann auch manuell angepasst werden, allerdings lassen sich diese JSON Files aber aufgrund des Pairings nicht mehr in die Child Anlagen einspielen. Als privater Anwender sollte man davon nichts merken, der SDV verbindet sich ja mir der CCU und kann das JSON in ein Programm umrechnen und damit das Restore ausführen.

Ich hoffe der Schritt von mir ist nachvollziehbar

2 Oberfläche

Die Bedienoberfläches des SDV besteht aus den folgenden Elementen: Zu fast allen Elementen bzw die wo es möglich ist, sind Hints programmiert

2.1 Hauptmenu (Mainmenü)

Host: Wechsel zwischen den verschienen CCUs, Möglichkeit zur Lizentanfrage und Donate

Selektiert: Markierte und Selektierte Objekte aus dem Inspektor lassen sich im Editor abrufen.

Syntaxcheck: Noch in Entwicklung

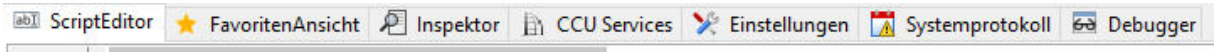
Methodenhilfe: Tief geschachteltes Menü mit typischen Hilfen für den Skriptprogrammieralltag

Backup: Menü mit den Möglichkeiten gezielte Backups vom System zu ziehen.

2.2 Toolbar

Die links seitliche Toolbar bietet Schnellzugriff auf die üblichen Editorfunktionen (undo, redo, Copy, paste, Suchen und Ersetzen etc. Die Buttons sind mit Hilfe-Hints ausgestattet

2.3 Arbeitsflächen (Pages)



SkriptEditor: S. Kapitel 3

Favoritenansicht s. Kapitel 5

Inspektor : s. Kapitel 4

CCU Services: Hilfe und Reparaturfunktionen sowie Gerätekopien

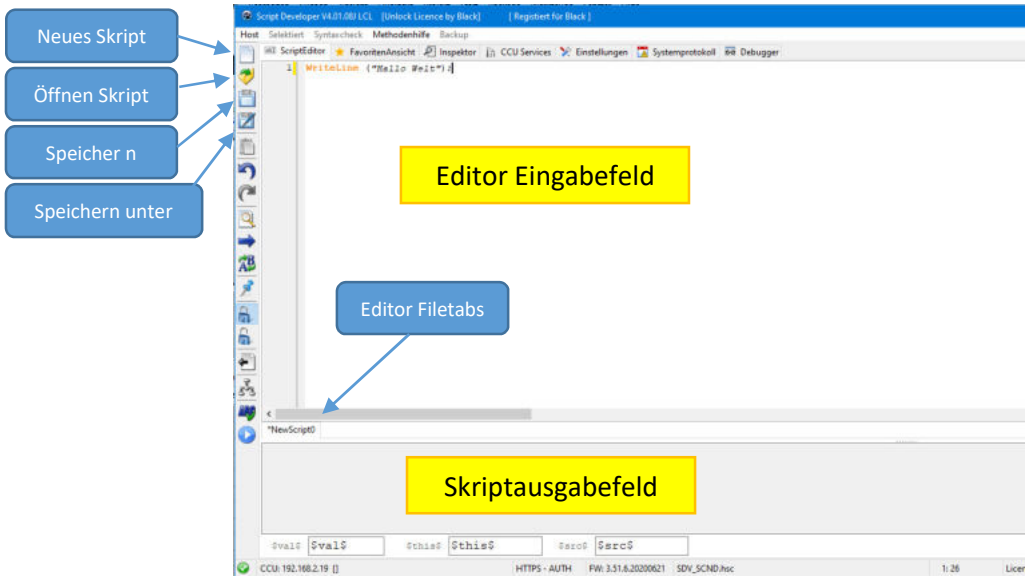
Einstellungen: Setup Möglichkeiten

Systemprotokoll: s. Kapitel 6 Ausgelesenes Systemprotokoll mit vielen Filtermöglichkeiten

Debugger: Testbereich für mich für die Entwicklung

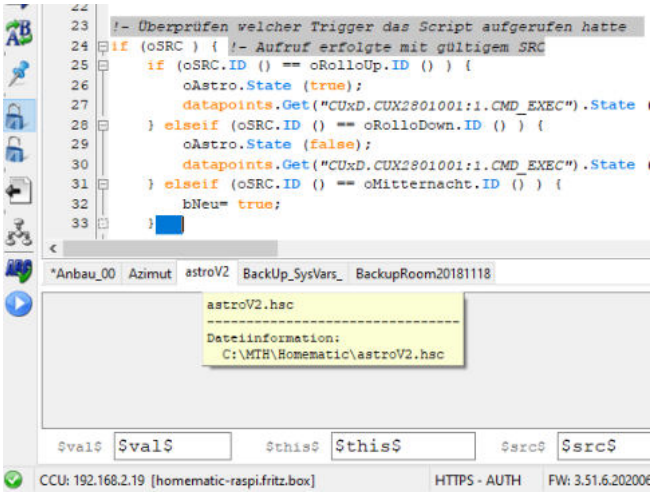
3 Skripteditor

Im Skripteditor werden die Skripte geschrieben oder geladen, die mittels Run Skript oder an die CCU zum Ausführen mit mit zum Testen an die CCU gesendet werden. Das Skriptergebnis wird dann unter Skriptausgabe dargestellt.

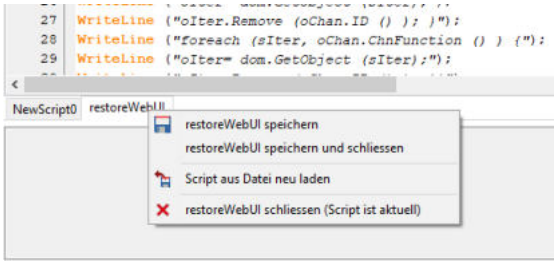


3.1 EditorTabs

Bei Mouse über einen Tab öffnet sich der Hint mit Informationen über die Datei.

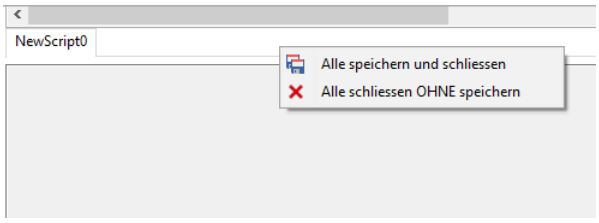


Rechte Mousetaste auf einen EditorTab öffnet PopupMenü:



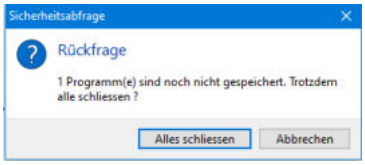
Speichern: schreibt den Inhalt des Editors bzw des Puffers in die Datei
Speichern und schliessen: schreibt den Inhalt des Editors bzw des Puffers in die Datei und schliesst den Tag
Script aus Datei neu laden: Wenn die Datei schon existiert, dann der Inhalt neu geladen werden und damit der Ext im Editor überschrieben werden.
Schliessen (mit Hinweis ist aktuell oder wurde geändert) schliesst den aktiven Editor Tab.

Klick mit rechte Mousetaste in freies Feld ausserhalb der Tabs öffnet Menü:



Alle Speichern und schliessen : alle geänderten Dateien werden gespeichern und die Tabs geschlossen

Alle Schliessen ohne Speichern: Alle Tabs werden ohne zu schliessen geschlossen. Es erfolgt noch eine Sicherheitsrückfrage:



1.1.1 Öffnen eines Scriptes

Es können bis zu 16 Scripte gleichzeitig geöffnet werden. Beim Versuch das 17. Script zu öffnen kommt eine Meldung, dass dieses nicht geht. Als Neues Script wird ein Script immer mit dem Namen NewScript mit fortlaufender Nummer angelegt.

Der Öffnen Mechanismus ist immer der folgende:
Existiert schon ein Tab mit dem Scriptnamen ?
Nein: Weniger als 16 Scripte offen ?
    Ja : Fehlermeldung und Abbruch
    Nein: Script öffnen
Ja: Steht das Script als Modified eingetragen ?
    Ja: Rückfrage ob Script überschrieben werden soll ?
    Nein: Script öffnen

3.1.2 Öffnen eines Scripts aus dem Inspektor (Extrakt aus der CCU)

Durch Klicken auf ** (Sternspecial) im Inspektor kann bekannter Weise direkt ein Script aus dem Inspektor exportiert werden und im Editor geöffnet werden.

Hier gilt folgender Mechanismus:

1. Der Scriptname ist Programmname_IDdesSingleDestinationObjektes
1.1 Es existiert schon ein Tab mit diesem Namen ?
2. Nein: Weniger als 16 Scripte offen ?
3. Ja : Fehlermeldung und Abbruch
4. Nein: Script öffnen
5. Ja: Steht das Script als Modified eingetragen ?
6. Ja: Rückfrage ob Script überschrieben werden soll ?
7. Nein: Script öffnen

In dem Tab wird die Herkunft des Scriptes vermerkt. Wird ein Tab geöffnet mit Kennung eines exportierten Scriptes, so wird der Rückladebutton farbig dargestellt und das Script bei Push auf den Rückladebutton wieder in das Singledestinationobject in der CCU wieder hochgeladen.

3.1.3 Öffnen eines Scriptes durch das Betriebssystem.

Ein oder auch mehrere Scripte können durch das Betriebssystem (Doppelklick auf eine Datei bei gesetzter Bedingung Öffnen mit) oder Drag und Drop abgelegt werden. Sollten dabei mehrere Prozesse durch das OS geöffnet werden, sprechen dies die SDV Instanzen über Inter-process-Communication ab, so dass nur die Hauptinstanz dann die Dateien öffnen wird und die weiteren Instanzen sich selber beenden.

Der Öffnen Mechanismus geht dann folgendermaßen:

1. Ist die Datei ein gültiges Script ?
2. Ja: Mechanismus gemäß 3.1.1

3.1.4 Öffnen der Scripte bei SDV-Start

Der SDV merkt sich die Namen der geöffneten Scripte beim SDV Programmende. Bei Start des SDV wird diese Liste wieder aus der INI geladen und nach folgendem Schema abgearbeitet:

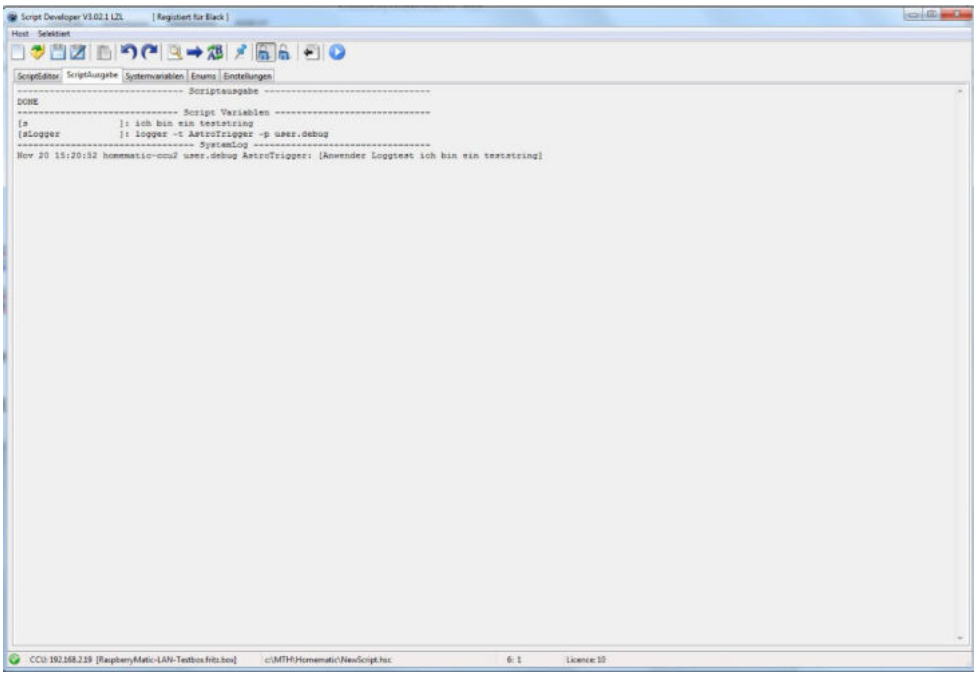
1. Datei existiert ?
2. Ja: Inhalt laden, Datei öffnen
3. Nein: Die Datei existiert nicht, also nix machen

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]
```

Ab 3.07.04 : Script testen eingefügt, hierbei wird das Skript nur nach Fehlern geparst aber nicht ausgeführt (Syntax Check). Funktionsweise äquivalent zu der KlickiBunti Funktion: Script Testen.

Dazu im Scripteditor auf  drücken, es wird dann der Syntaxcheck der CCU aufgerufen. Bei erfolgreichem Syntaxcheck erfolgt für 2 Sekunden die Meldung

```
Test Script erfolgreich
```

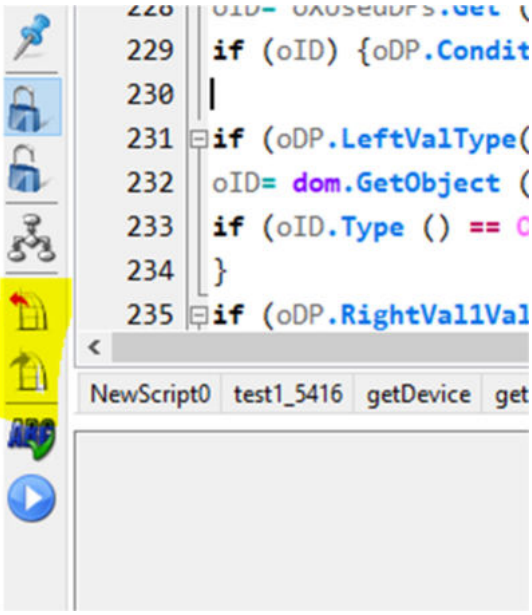
Trat ein Fehler auf, so erfolgt Sprung auf den Reiter Scriptausgabe mit Anzeige der Fehlermeldung

Bestandsnutzer können den folgenden Schlüssel in die SDV.INI schreiben:

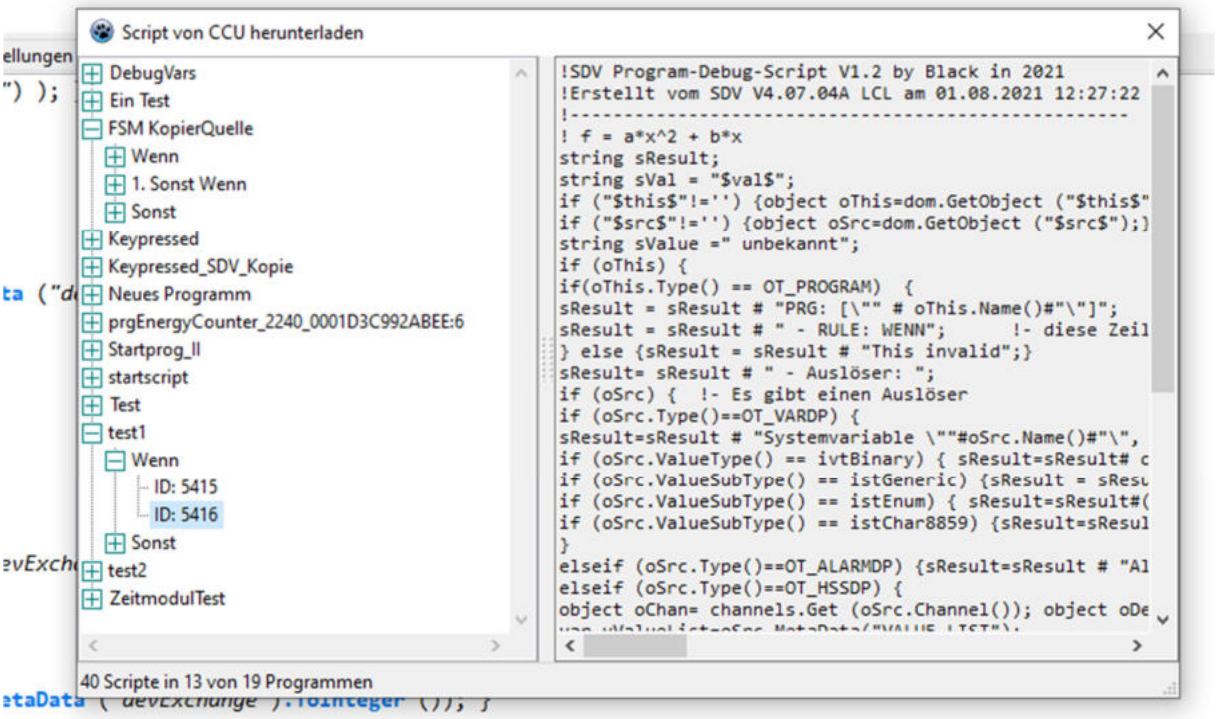
```
[SCRIPTRUN]
RUN=F1
TEST=F12

Die legt Script RUN auf F1 und Script testen auf F12. Kann selber angepasst
werden. F3 ist reserviert !!! für suche erneut
```

3.1.5 Öffnen eines Skriptes aus der CCU und Zurückspielen

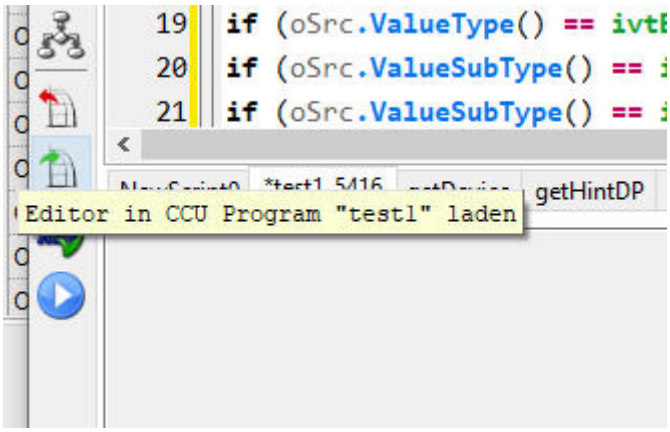
Mit der neuen Version ab 4.7.5 ist es komfortabler Möglich, direkt ein Script über die Menüleisten aus der CCU herunterzuladen bzw. auf diese zurückzuspielen



Roter Pfeil öffnet ein Auswahlmenü, mit dem sich unter allen Scripten aus der CCU eins auswählen lässt:

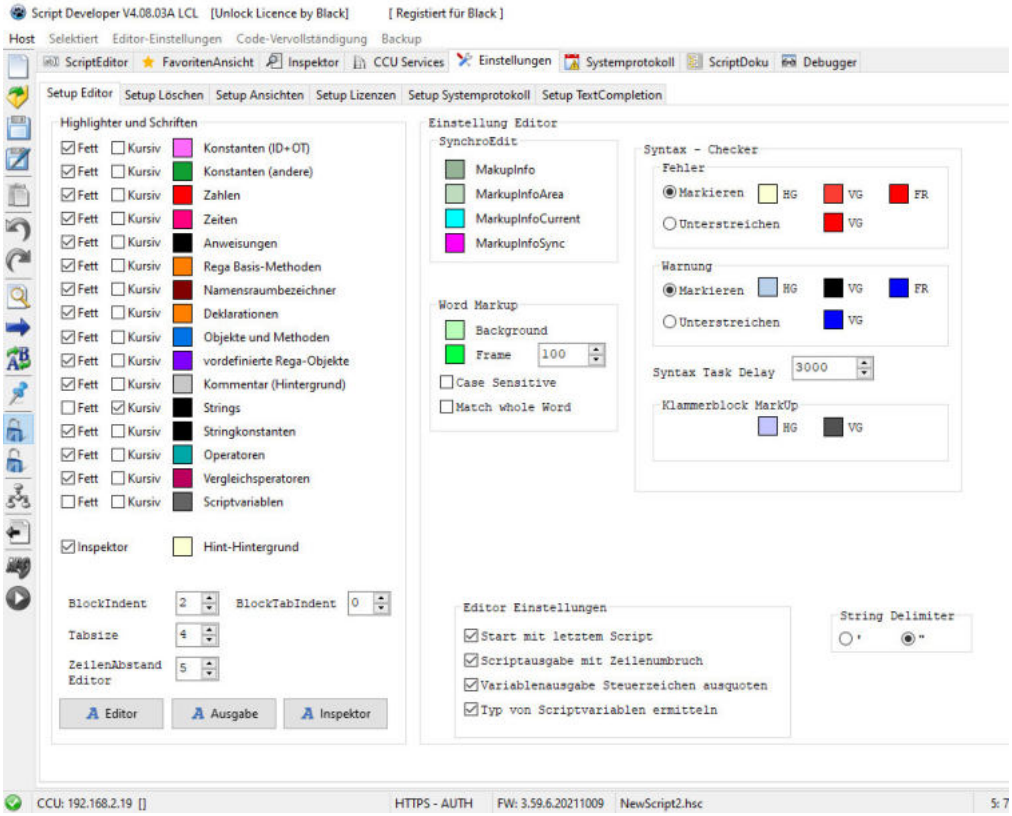


Links im Treeview sind die Programme und die darin enthalteten Scripte inkl des Verweises, in welchem Teil des Programmes das entsprechende Script steht. Eine Schnellansicht des Scripte wird im rechten Teil des Fensters dargestellt. Doppelklick oder Enter übernimmt das Skript gemäß der unter 3.1.2 beschriebenen Bedingungen im Editor.



Mit der grünen Taste wird das Programm aus dem Editor dann wieder in die richtige Stelle in der CCU zurückgespielt.

3.2 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.

Editor Einstellungen:
Start mit letztem Script:
Enabled: Beim Starten des SDV werden die beim Schliessen gemerkten Tabs wieder versucht zu öffnen.
Disabled: Der SDV startet mit einem leeren Neuen NewScript0

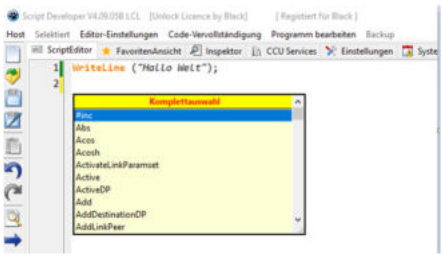
Scriptausgabe mit Zeilenumbruch:
Enabled: bei der Darstellung der Scriptausgabe wird bei langen Zeilen umgebrochen
Disabled: lange zeilen werden auch nur als eine Zeile dargestellt mit Scrollbar

Variablenausgabe Steuerzeichen ausquoten
Enabled: XML Codierte Ausgabe des Variableninhaltes wird menschenlesbar wieder nochmal umgequotet
Disabled: Ausgabe bleibt in Originalzustand

Typ von Scriptvariablen ermitteln.
Hier wird vom SDV automatisch ZusatzCode an das zu testende Script angefügt und anschliessend das Ergebnis ausgewertet. So sind Aussagen über den Variablentyp möglich und ob sich der Typ der Variable geändert hatte und wenn ja, in welchen Typ.

3.3 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.

Diese recht grobe Gesamtvorgabe gibt es noch etwas feinmodularer:

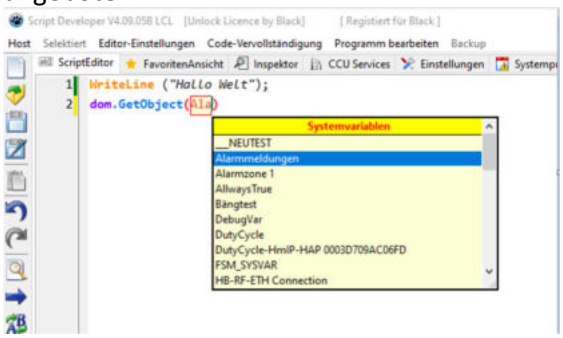
3.3.1 Verfeinerte Vervollständiger Funktionen

CTRL SHIFT M : Alle Methoden
CTRL SHIFT S : Alle String Methoden
CTRL SHIFT C : Alle Mathematic Methoden
CTRL SHIFT N : Alle ID_ Konstanten
CRTL SHIFT O : Alle OT_ Konstanten

3.3.2 Spezielle Vervollständiger Funktionen

CTRL SHIFT D : Echtzeit Skriptanalyse: Alle im Skript deklarierten Variablen werden vorgeschlagen

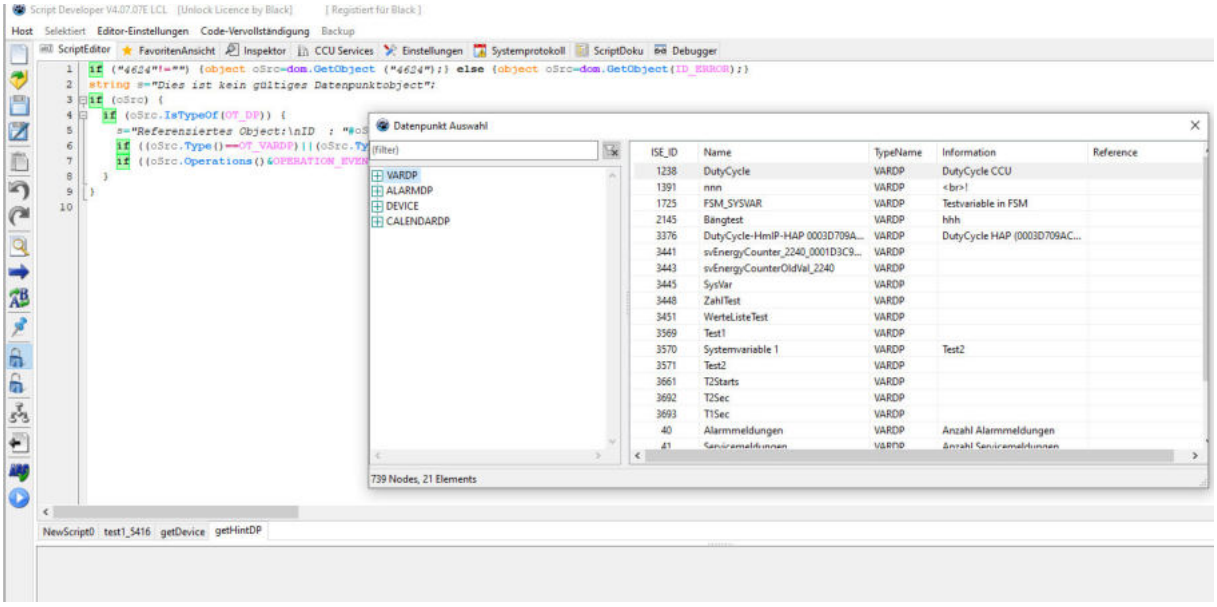
CTRL ALT S : Echtzeitanalyse auf der CCU: Im Vervollständiger werden alle Systemvariablen angeboten



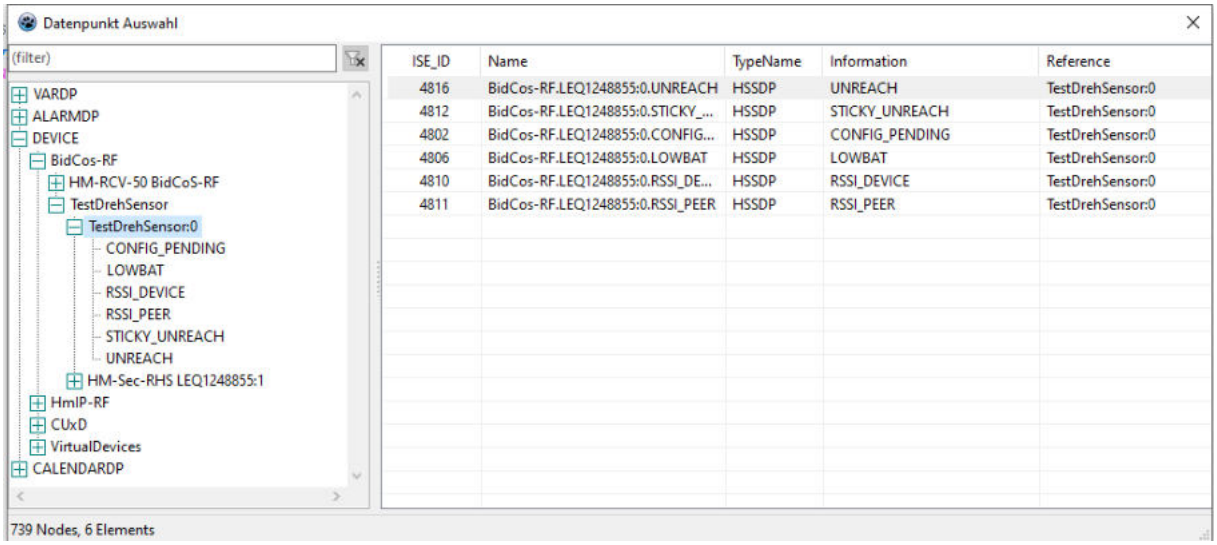
CTRL ALT G : Echtzeitanalyse auf der CCU: Im Vervollständiger werden alle Gewerke angeboten
CTRL ALT R : Echtzeitanalyse auf der CCU: Im Vervollständiger werden alle Raum angeboten
CTRL ALT D : Echtzeitanalyse auf der CCU: Im Vervollständiger werden alle Devices angebote
CTRL ALT P : Echtzeitanalyse auf der CCU: Im Vervollständiger werden alle Programme angeboten

3.3.3 Qualifizierter Objektzugriff

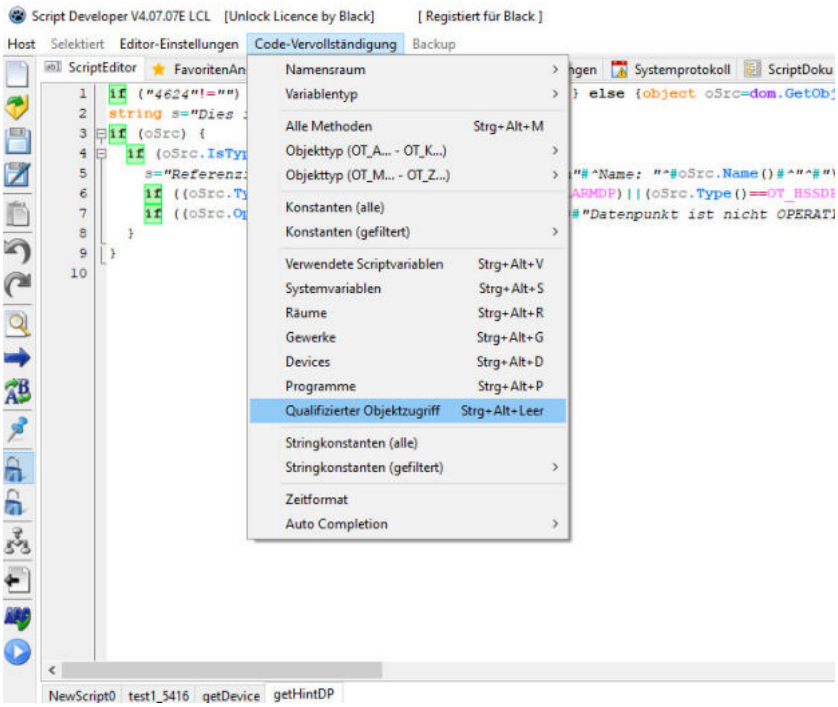
Anwendung findet dieses erstmal an zwei Stellen im Editor. Erstmal über src und Doppelclick öffnet sich der Auswahl Dialog:
An der Stelle sind zum Programmtriggern: Systemvariablen, Datenpunkte und Zeitmodule erlaubt. Grösse des Dialogs, der Anteil zwischen dem linken VirtualTreeview und dem rechten Listview sowie die Spaltenbreiten des Listviews lassen sich verändern und werden beim Programmende persistiert und beim Programmstart geladen.



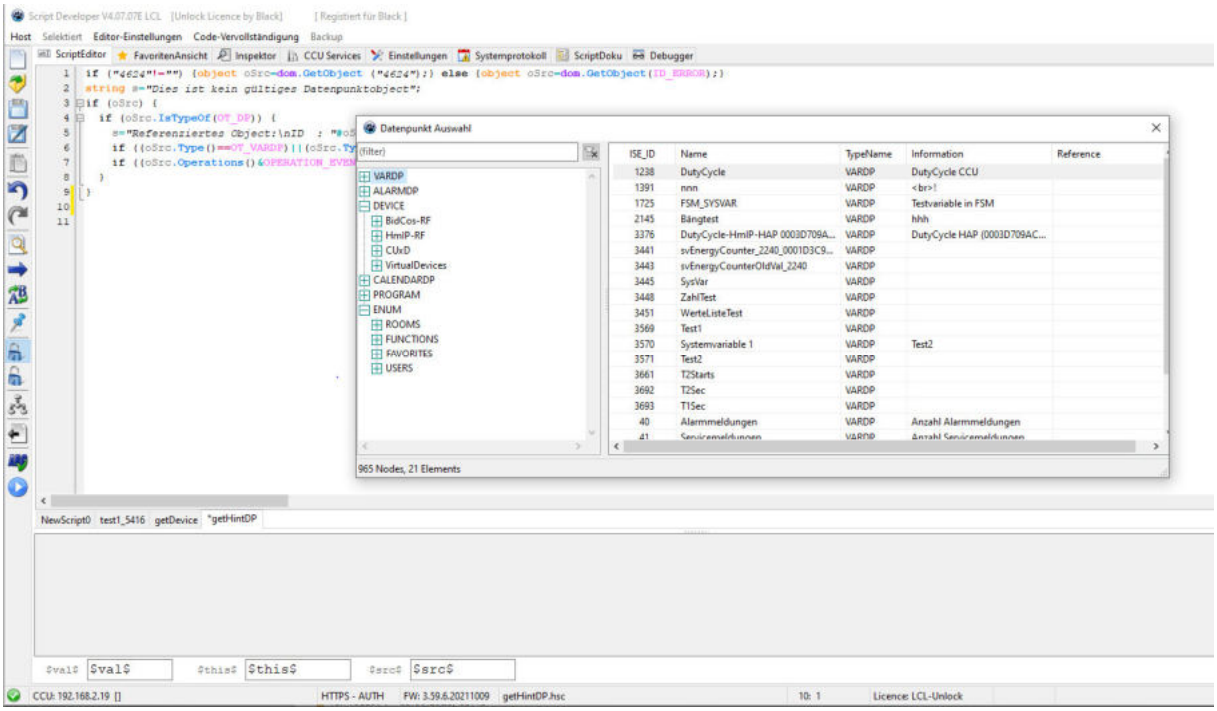
Über den Treeview wird die Grobauswahl gemacht (Endauswahl mit Doppelklick oder auch Enter geht auch) in der Listview die Endauswahl gemacht (Auch hier Doppelklick oder Enter). Zwischen dem Baum und dem Listenfenster lässt sich mit ctrl rechts bzw ctrl links öffnen. im Baum klappt Pfeil rechts einen Node auf, Pfeil links klappt den Node zu. Die Filtereingabe funktioniert so wie auch in der Skriptdoku, damit wir der gesamte Baum nach dem eingegebenen Suchbegriff gefiltert und die passenden Nodes aufgeklappt.



Die Funktion wurde nun auch voll in den Editor integriert. Es gab im SDV Editor noch keine vernünftige Auswahl direkt im Editor von Kanälen und Datenpunkten, die verwendete Completion Componente wäre für eine sinnige Baumstruktur darstellen schlicht ungeeignet. Der Umweg ging bisher über den Inspektor, auswählen des Punktes und dann mit Alt-O den vollqualifizierten Zugriff im Editor generieren lassen. Dies geht nun komfortabler:



Die alten Auswahlmethoden über die Completion wurden beibehalten, hinzu kam nun der Menupunkt "Qualifizierter Objektzugriff". Anwahl entweder über Menüpunkt oder über Ctrl-Alt-Space. Die gleiche Class wie eben auch, aber mit mehr Auswahlmöglichkeiten:



Hier lässt sich wieder in der Baumstruktur das Gewünschte Object grob (auch fein) auswählen, in der Listview dann Feinauswählen. WIe eben auch: Doppelklick oder Return. Je nachdem, welches Objekt man ausgewählt hat, wird daraufhin im Editor der Vollqualifizierte Zugriff eingetragen. z.b. Bei einem Enum Raum wird ein dom.GetObject (ID_ROOMS).Get("DerGewählte Raumname") erzeugt, für alle anderen entsprechenden Objekte auch der dementsprechende qualifizierte Zugriff. Interfaces, Devices und Channels wählt man in der Baumstruktur aus (Doppelklick oder Enter) Damit lassen sich nun komfortabel direkt im Editor Datenpunkte auf der CCU auswählen und sich auch automatisch der korrekte qualifizierte Zugriff im Editor generieren, ohne dass man über den Inspektor gehen muss.

Wenn das letzte Zeichen vor dem Cursor ein . ist und der Auswahldialog aufgerufen wird, dann werden nur die Devices eingeblendet werden und bei Auswahl nur noch der Part DPByHssDP übernommen. Grund wäre, wenn z.B. ein ChannelObject die Basis bildet und man schnell auf mehrere verschiedene Datenpunkte zugreifen möchte in der Art oChannel. <- dann der Aufruf, da braucht man dann nicht mehr den kompletten vollqualifizieren Zugriff, sondern nur noch den Part DPByHssDP ("SelektierterDatenpunkt")

3.3.4 AutoComplete

Unter Autocomplete lassen sich häufig verwendete Skriptsegmente ablegen und unter Stichworten abrufen.

Beispielhafter Aufbau:

SystemZeit_allgemein
= system.Date("|");

Beschreibung des Skriptsegmentes
= beschreibt den dann einzufügenden Text
Leerzeile trennt nächstes Skriptsegment

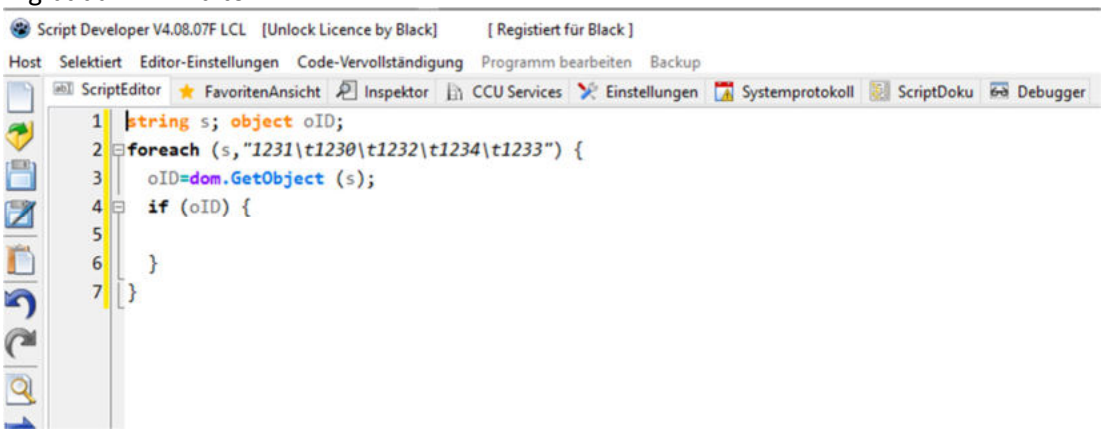
SystemZeit_Stunde_Minute
= system.Date("%H:%M");

foreach
=foreach (s,o)
={
=
=}

Es gibt noch folgende Sonderform:
Der Placeholder $$$ID$$$ kann durch gespeicherte ID-Listen ersetzt werden kann. Konkretes Anwendungsgebiet: Bestimmte Objekte, die im Inspektor markierte wurden, sollen im Skripteditor iteretiert werden um dort Aktionen durchzuführen

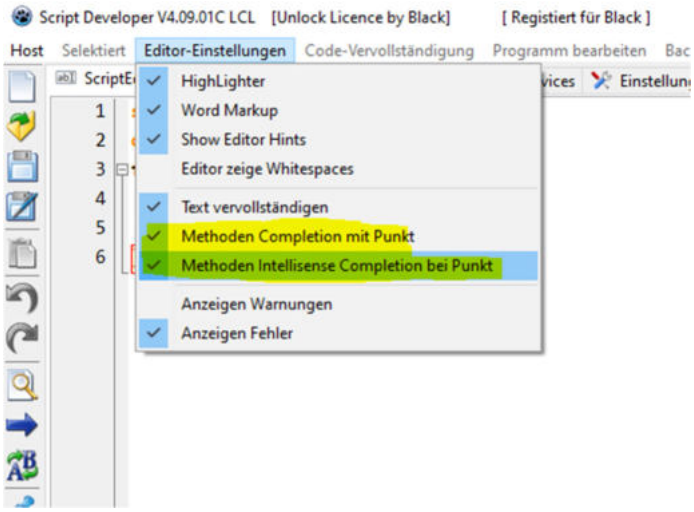
foreach_selected
=string s; object oID;
=foreach (s,$$$ID$$$) {
= oID=dom.GetObject (s);
= if (oID) {
=XXX
= }
=}

Ergibt dann im Editor:



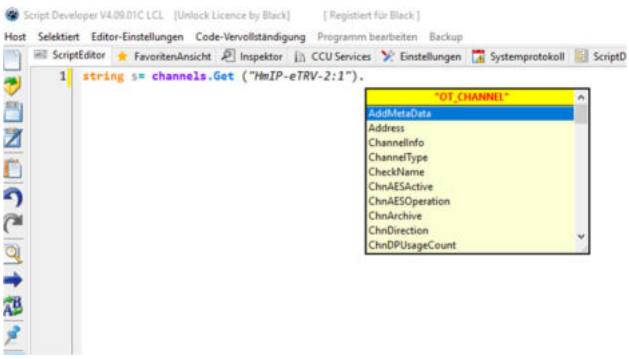
3.3.5 Vervollständigen über „Intellisense“

Der Editor ist ab der Version 4.09.xx mit einer verbesserten IntelliSense Funktionalität ausgestattet. Zum Aktivieren müssen beide Haken gesetzt werden



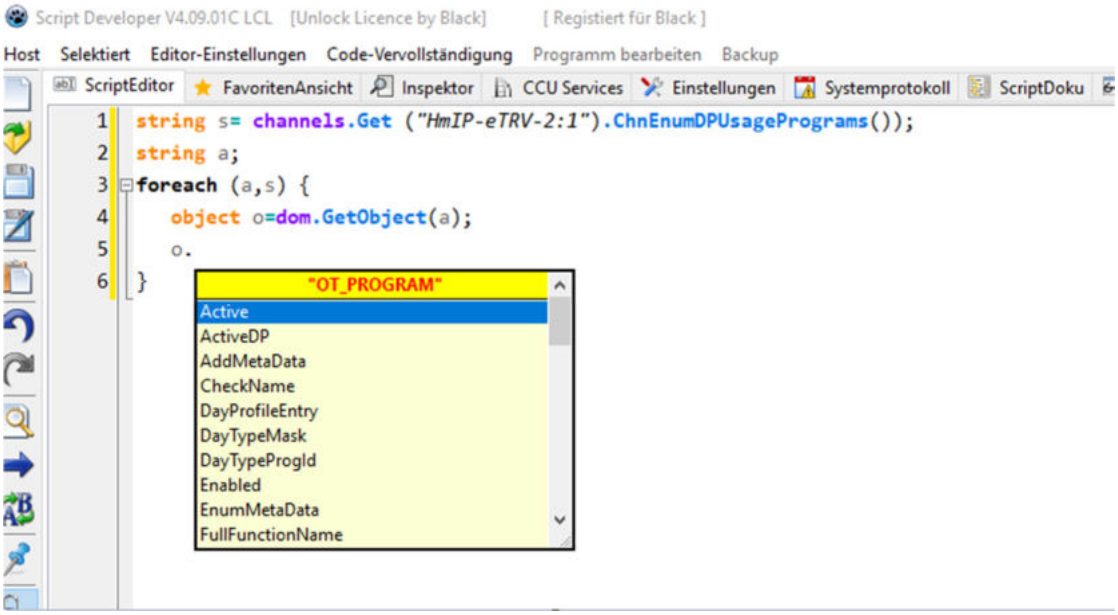
Wird nun beim Schreiben im Editor der . gedrückt und an dieser Stelle eine Methode möglich ist, so bestimmt der SDV nun mithilfe seines Echtzeit LL1 Parsers und unter Zuhilfenahme der RegaAbstraction Unit, welche Methoden an dieser Stelle wahrscheinlich möglich sind und schlägt diese nun vor.

An dieser Stelle hier sind die Methoden eines Objektes von Type OT_CHANNEL möglich



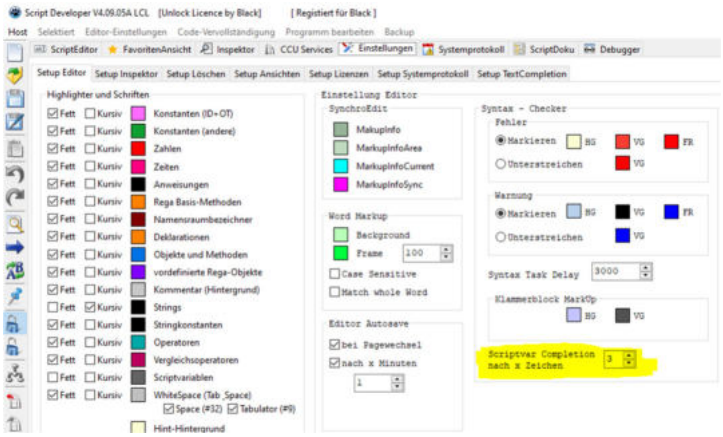
Es wird auch zur Info der Objekttyp angezeigt (oder Variablentyp bzw. Namensraum).

Durch die Rega Abstraction Unit gehen auch Analytische Vorhersagen, wie in dem Beispiel hier wird auf ein OT_PROGRAM gerechnet

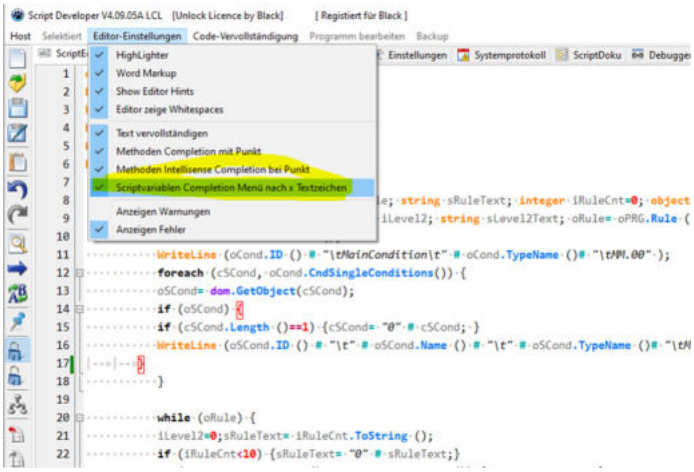


3.3.6 Autopopup Skriptvariablen

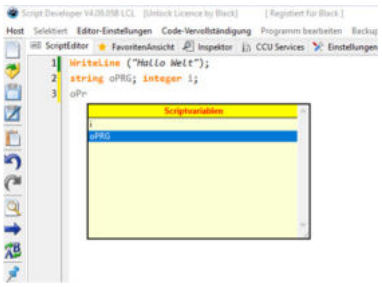
Der SDV Editor kann so eingestellt werden, dass während des Schreibens die Wahrscheinlichkeit der Verwendung einer Skriptvariablen erkannt wird und bei entsprechender Einstellung sich automatisch das Completion Fenster für die Skriptvariablen öffnet.
Die Prüfung findet nach einer einstellbaren Mindestwortlänge statt



In die diesem Beispiel müssen mindestens 3 Zeichen geschrieben worden sein.
Zusätzlich muss die Funktion generell eingeschaltet worden sein



Das Wort muss beginnen mit Zeichen für einen gültigen Variablennamen, es darf kein String oder kein Kommentar sein, die Mindestlänge muss erreicht sein, die Funktion muss angewählt sein und die getippte Buchstabenfolge muss schon dem Anfang mindestens einer vorhanden Skriptvariablen entsprechen, dann öffnet sich automatisch das Completion PopUp.



3.3.7 Komfortables Beenden und Weiterschreiben

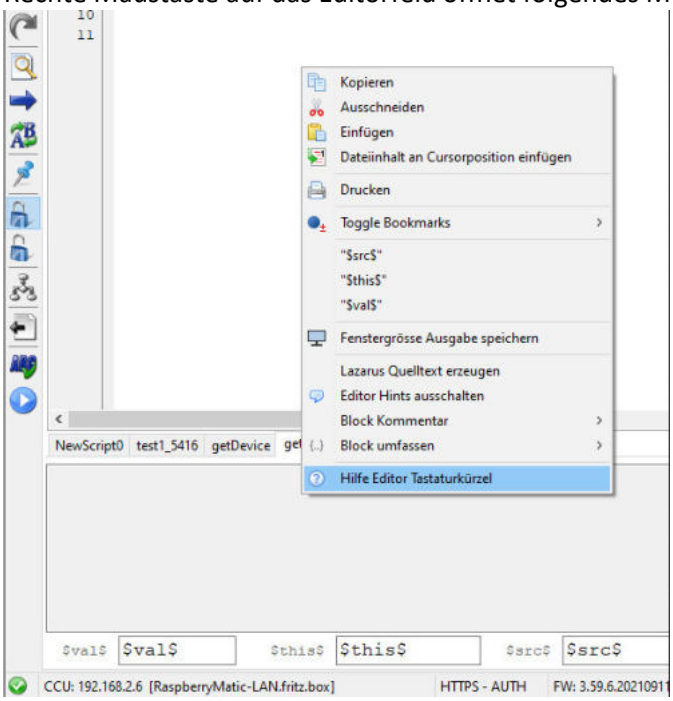
Die Completion Menüs lassen sich alle auch mit „nicht Identifier keys“ beenden, heißt z.B. .,+-*/ etc. Wenn die Auswahl zB. mit dem Punkt beendet wird, so wird z.B. die selektierte Skriptvariable übernommen und es öffnet sich Automatisch das Completion Menü, wo bei aktiviertem IntelliSense die zu der vorherigen Auswahl möglichen Methoden ausgewählt werden können.

Bei Methoden bewirkt das Beenden mit dem „.“ folgendes:
Es werden automatisch Klammen gesetzt
Ist ein leerer Parameter „()“ möglich, so wird der Punkt nach den Klammern gesetzt und automatisch wieder das IntelliSense Menü für die nächste Methodenauswahl aufgerufen
Ist kein leere Parameter möglich, so wird der Cursor innerhalb der Klammern positioniert

Auswahl mit Doppelklick oder Return
Hat die Methode nur leeren Parameter „()“ so wird der Cursor nach der „)“ positioniert, existiert mindestens ein Parameterwert, wird der Cursor in die Klammer gesetzt.

3.4 Editormenü (rechte Maustaste)

Rechte Maustaste auf das Editorfeld öffnet folgendes Menü:



Kopieren , Ausschneiden und Einfügen sind die Standartfunktionen unter CTRL C, CTRL X und CTRL V

3.4.1 Dateiinhalt an Cursorposition einfügen

Das Dateiauswahlmenü öffnet sich. Der Inhalt der ausgewählten Datei wird an der Cursorposition eingefügt. Dabei wird Der Undo Buffer berücksichtigt, nach dem Laden kann der Vorgang mit Undo rückgängig gemacht werden

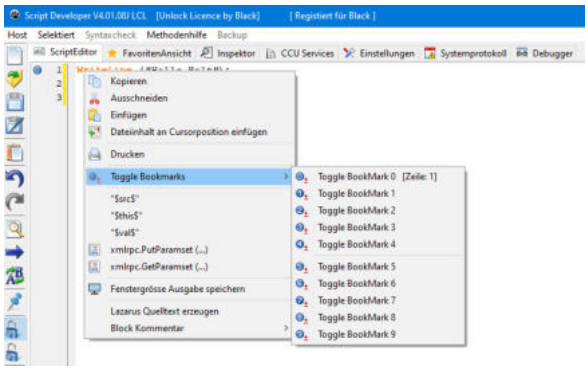
3.4.2 Drucken

Der Text des Editors samt Highlighting wird an den Standartbrowser geschickt zum Drucken.

3.4.3 Toggle Bookmarks

Pro Skript können bis zu 10 Bookmarks gesetzt werden (Möglichkeiten des Schnellsprunges)

Diese werden auch mit gesichert, wenn der SDV verlassen wird.



Natürlich sind die Bookmarks dynamisch, wenn die Zeilen verschoben werden.

3.4.4 src, $this$ und val

fügen den Text an der Cursorposition ein. Diese Felder können verschiedene Werte in den unterschiedlichen Skriptabs haben

3.4.5 Xmlrpc.PutParamset:

Wurde im Inspektor ein oder mehrere Master oder Linkparameter (bis zu 5 Stück) selektiert, so erzeugt dieser Menüpunkt automatisch den richtigen Code, um diese Parameter zu beschreiben. S.a. Kapitel 3.8

3.4.6 Xmlrpc.getParamset:

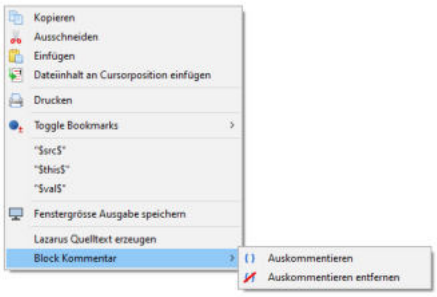
Wurde im Inspektor ein oder mehrere Master oder Linkparameter (bis zu 5 Stück) selektiert, so erzeugt dieser Menüpunkt automatisch den richtigen Code, um diese Parameter auszulesen. S.a Kapitel 3.8

3.4.7 Fenstergrösse Ausgabe speichern:

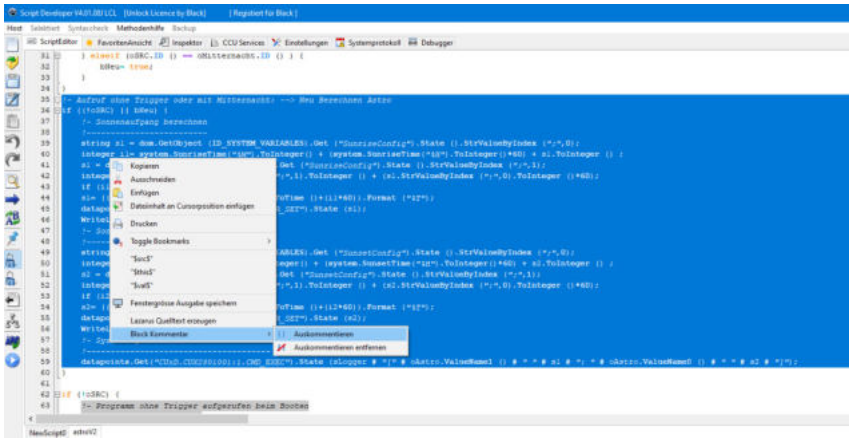
Die Grösse des Ausgabefensters kann abgespeichert werden.
Rechte Maustaste in den Editor und Fenstergrösse Ausgabe speichern

3.4.8 Block Kommentar

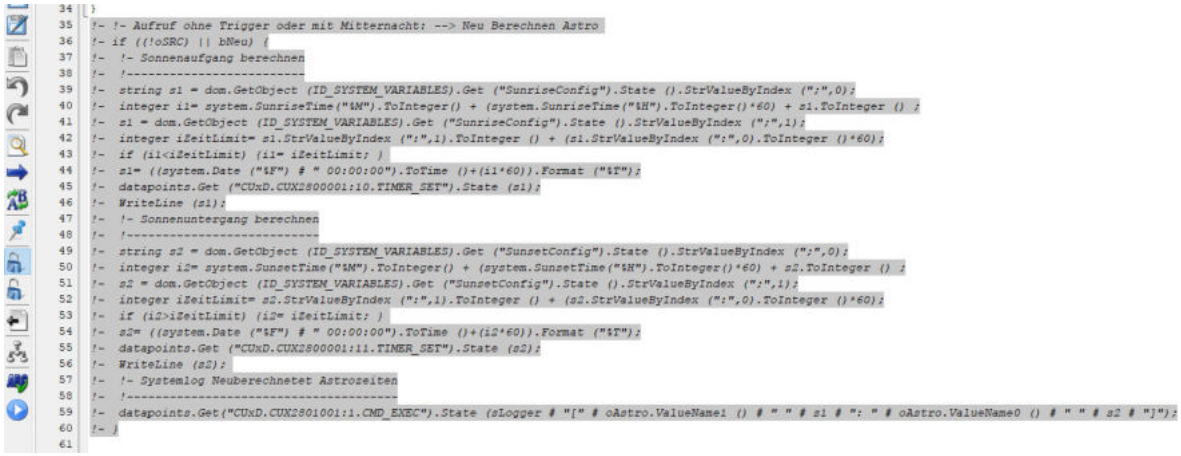
Mit dieser Funktion können im Editor selektierte Zeilen aus bzw auskommentiert werden



Auskommentieren:



Wird zu:



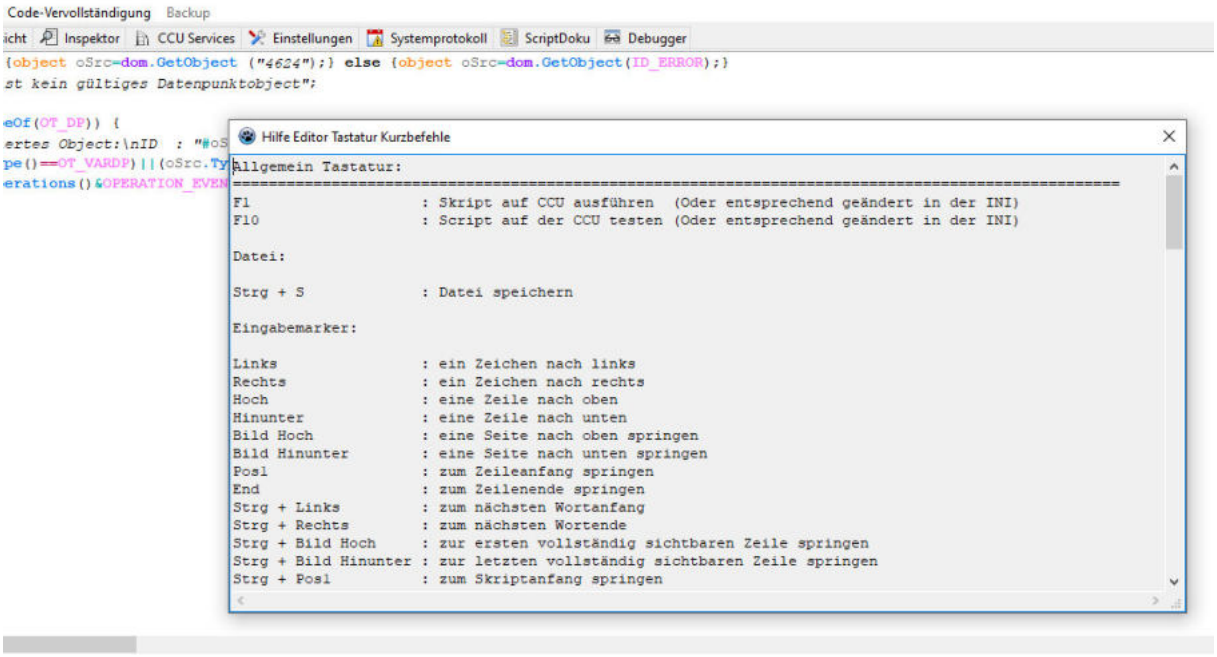
```
}
!- !- Aufruf ohne Trigger oder mit Mitternacht: --> Neu Berechnen Astro
!- if ((!oSRC) || bNeu) {
!-   !- Sonnenaufgang berechnen
!-   !-------------------------
!-   string s1 = dom.GetObject (ID_SYSTEM_VARIABLES).Get ("SunriseConfig").State ().StrValueByIndex (";",0);
!-   integer i1= system.SunriseTime("%M").ToInteger() + (system.SunriseTime("%H").ToInteger()*60) + s1.ToInteger () ;
!-   s1 = dom.GetObject (ID_SYSTEM_VARIABLES).Get ("SunriseConfig").State ().StrValueByIndex (";",1);
!-   integer iZeitLimit= s1.StrValueByIndex (":",1).ToInteger () + (s1.StrValueByIndex (":",0).ToInteger ()*60);
!-   if (i1<iZeitLimit) {i1= iZeitLimit; }
!-   s1= ((system.Date ("%F") # " 00:00:00").ToTime ()+(i1*60)).Format ("%T");
!-   datapoints.Get ("CUxD.CUX2800001:10.TIMER_SET").State (s1);
!-   WriteLine (s1);
!-   !- Sonnenuntergang berechnen
!-   !-------------------------
!-   string s2 = dom.GetObject (ID_SYSTEM_VARIABLES).Get ("SunsetConfig").State ().StrValueByIndex (";",0);
!-   integer i2= system.SunsetTime("%M").ToInteger() + (system.SunsetTime("%H").ToInteger()*60) + s2.ToInteger () ;
!-   s2 = dom.GetObject (ID_SYSTEM_VARIABLES).Get ("SunsetConfig").State ().StrValueByIndex (";",1);
!-   integer iZeitLimit= s2.StrValueByIndex (":",1).ToInteger () + (s2.StrValueByIndex (":",0).ToInteger ()*60);
!-   if (i2>iZeitLimit) {i2= iZeitLimit; }
!-   s2= ((system.Date ("%F") # " 00:00:00").ToTime ()+(i2*60)).Format ("%T");
!-   datapoints.Get ("CUxD.CUX2800001:11.TIMER_SET").State (s2);
!-   WriteLine (s2);
!-   !- Systemlog Neuberechnetet Astrozeiten
!-   !--------------------------------------
!-   datapoints.Get("CUxD.CUX2801001:1.CMD_EXEC").State (sLogger # "[" # oAstro.ValueName1 () # " " # s1 # ": " # oAstro.ValueName0 () # " " # s2 # "]");
!- }
```

Eine nützliche Funktion, um beim Skripttesten mal eben schnell einen Block totzulegen ohne ihn löschen zu müssen oder mühsam von Hand auszukommentieren.

Auskommentieren entfernen entfernt von dem Selektierten Block die beginnende !- Kommentierung

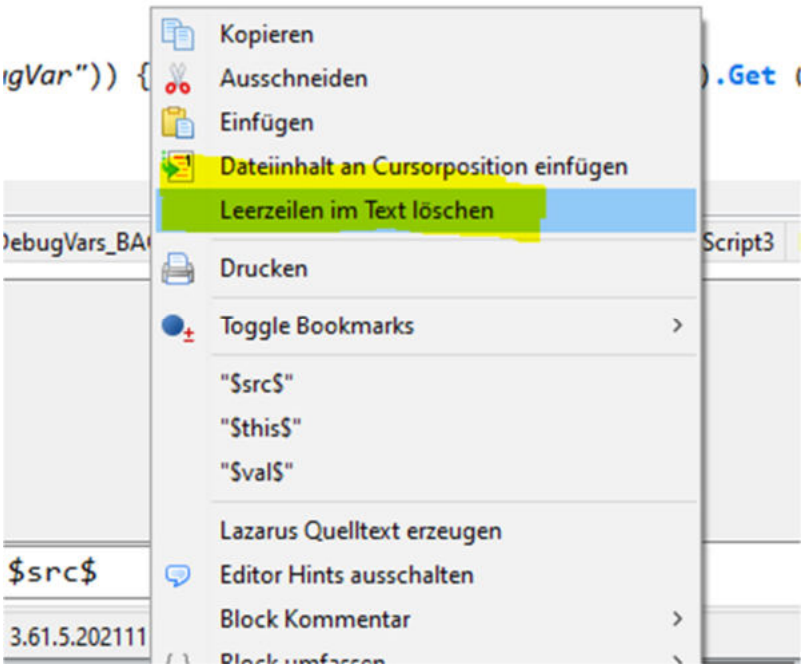
3.4.9 Editor Tastaturkürzel Hilfe

Kurzdarstellung der vom Editor verwendeten Tastaturkürzel



3.4.10 Überflüssige Leerzeilen entfernen

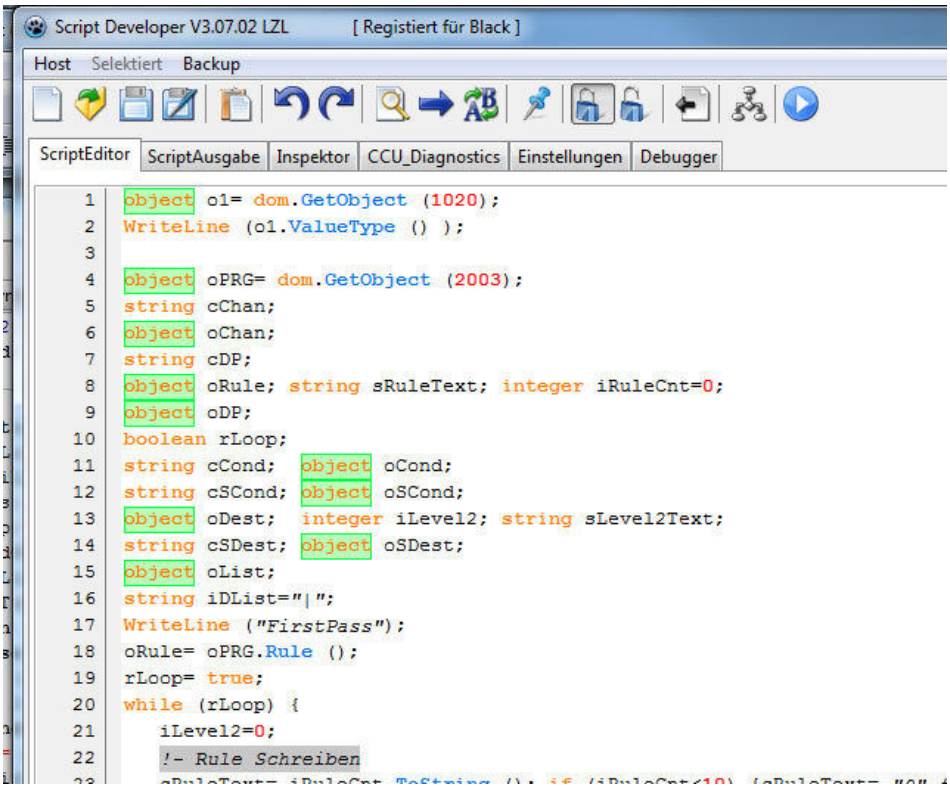
Sollte eine Editordatei aus irgendwelchen Gründen mal kaputt sein (sinnlos viele Zeilenvorschübe etc) kann dieses Menü helfen



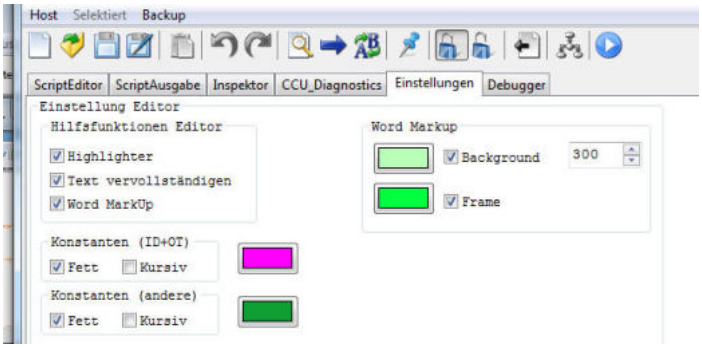
Leerzeilen innerhalb Strings erkennt der SDV und löscht diese nicht

3.5 Word Markup (seit 03.07.02)

Der Editor beherrscht nun auch Wordmarkup. Bei einem Word, welches angeklickt wurde, oder geschrieben wurde (auf dem sich der Focus befindet ^^), werden sämtliche Vorkommen im Text farblich hervorgehoben



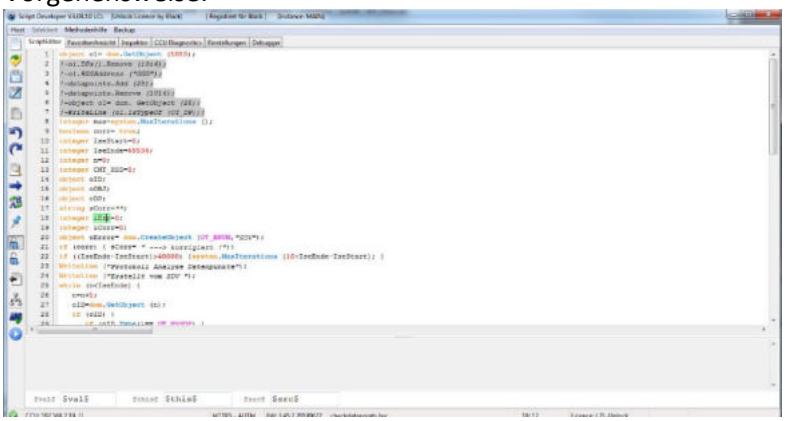
Die Funktion ist konfigurierbar unter Einstellungen



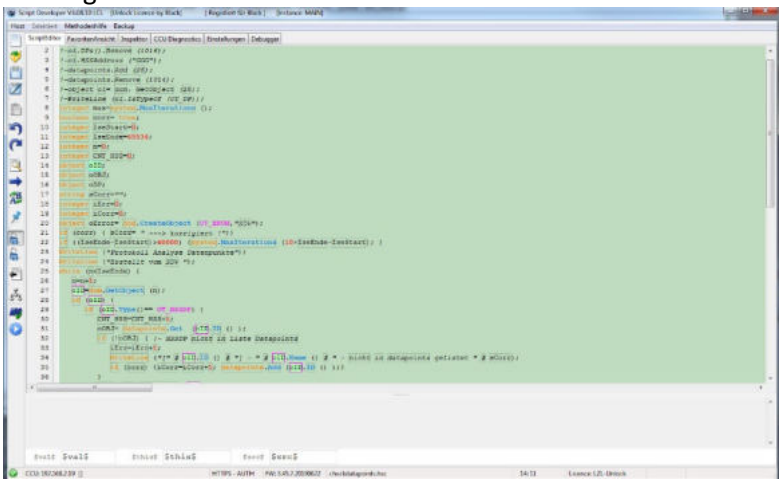
Die Einstellungen sollten selbsterklärend sein, die Zahlenangabe bezieht sich auf die Zeit in ms, ab wann seit dem letzten Tastenanschlag die Hervorhebung beginnt.

3.6 SyncroEdit (ab 3.08.10)

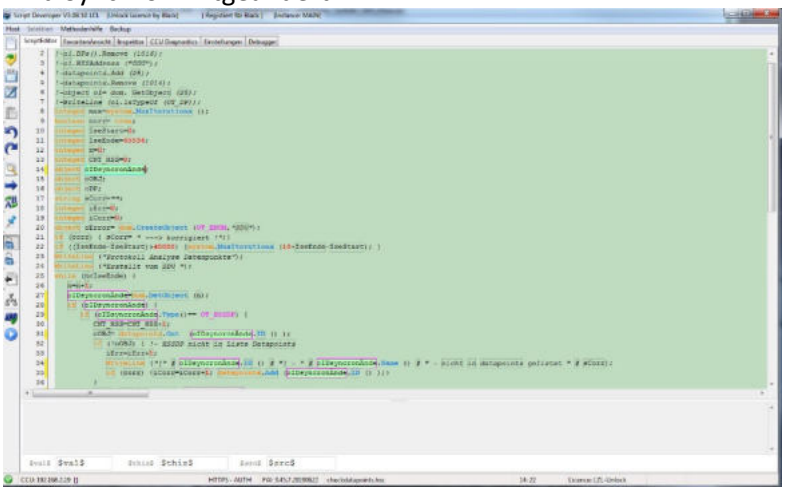
Mit SyncroEdit lässt sich in einem Bereich synchron an mehreren gleichen Stellen ändern.
Vorgehensweise:



Den Textbereich in dem geändert werden soll, markieren, anschliessend mit Strg-J zum Synchroneditieren vorbereiten. Der Bereich wird dann mintgrün, die Worte werden hervorgehoben und ich klick nun dahin wo ich ändern will. Hier Hinter das oID (ist durch den markup auch hellgrün hervorgehoben

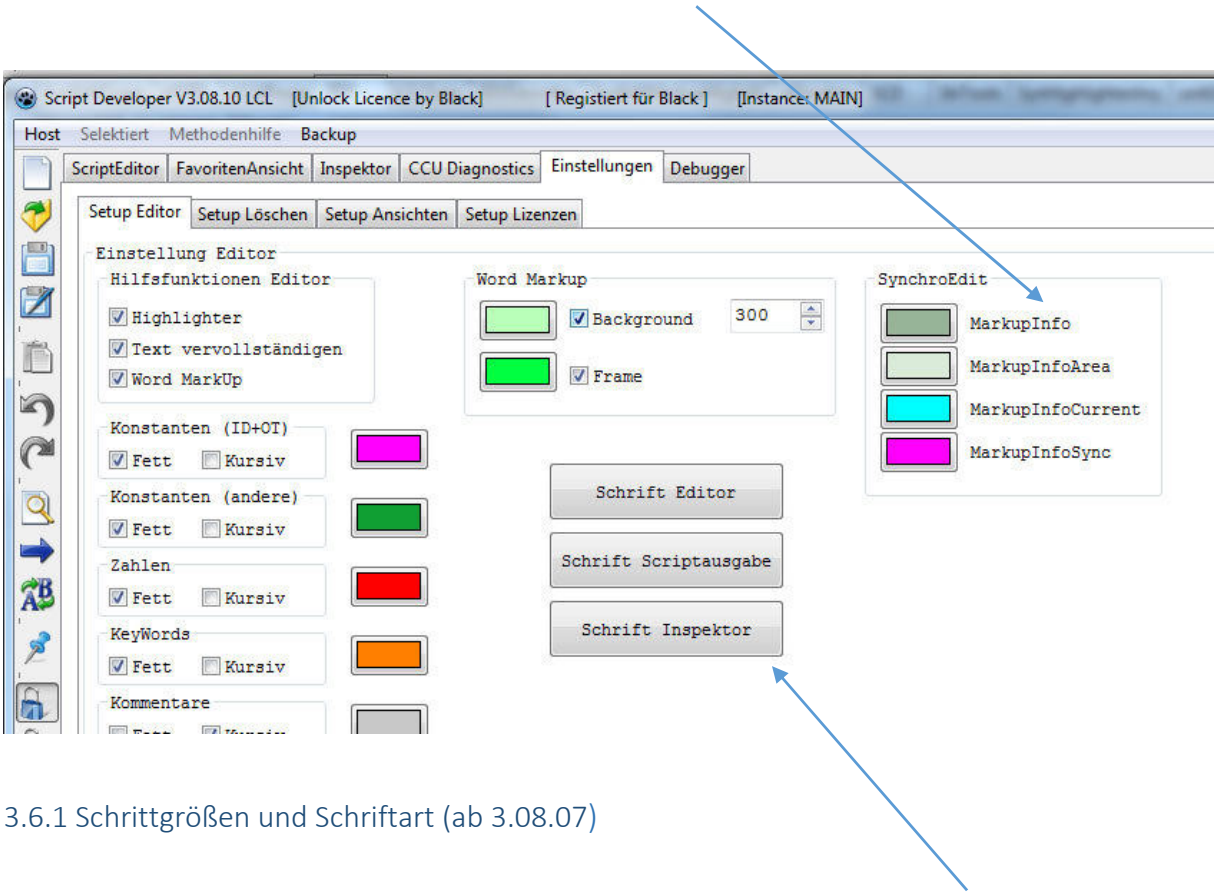


Jetzt kann man schreiben und an allen hellgrün hervorgehobenen Stellen im Mindgrünen bereich wird Synchron mitgeändert.



Die Markierung aufheben geht dann mit Escape.

Die Farben lassen sich natürlich in Setup Menü einstellen

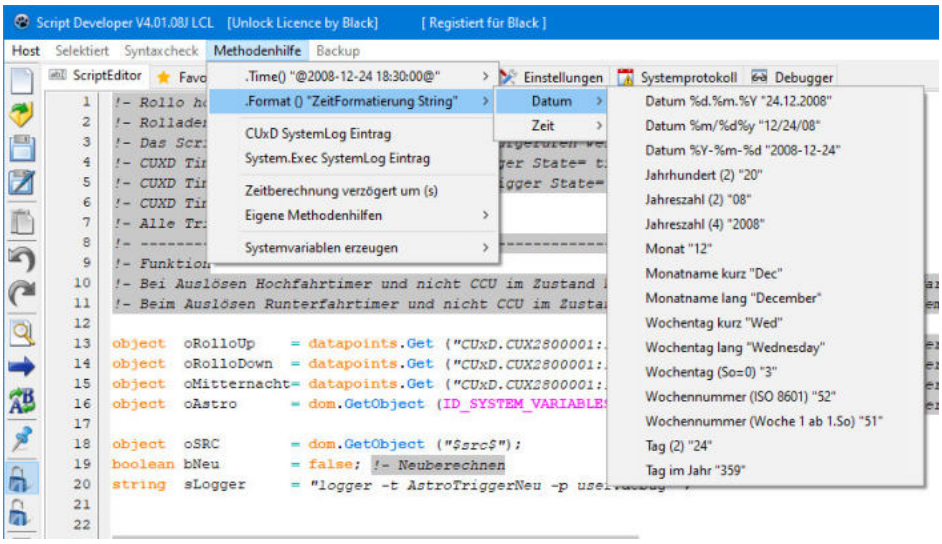


3.6.1 Schrittgrößen und Schriftart (ab 3.08.07)

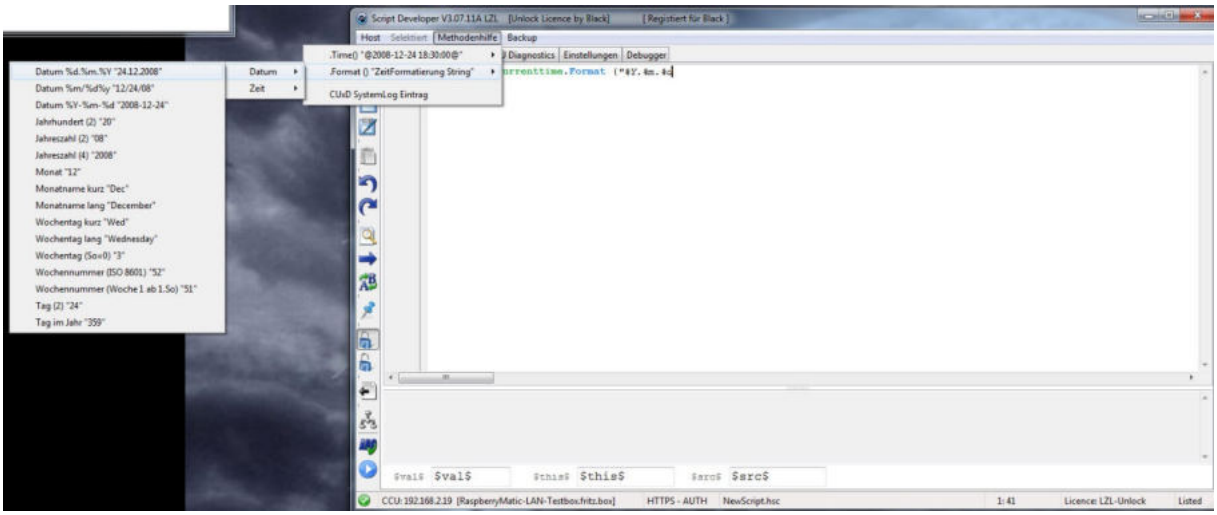
Einstellungen für Schriftwart Editor, dem Editor Ausgabefenster und im Inspektor

3.7 Methodenhilfe

Da es mich immer nervte, bei bestimmten Funktionen doch in Handbuch gucken zu müssen, gibt es nun den Menüpunkt Methodenhilfe (Aktiv im Editor)
Damit lassen sich zu Themen Time, Zeit, und Zeitformatierung Funktionen und die Formatstring schnell finden



Durch Druck wird die Passende Funktion ausgewählt. Dito bei Zeitformatierung



Ab dem Version 3.08.12D verfügt der SDV über die Möglichkeit, bis zu zehn eigene Methodenhilfen zu definieren. Diese werden in der INI angelegt.

```
; Common escape sequences
;\\       \ (a single backslash, escaping the escape character)
;\'       Apostrophe
;\"       Double quotes
;\t       Tab character
;\r       Carriage return
;\n       Line feed
;\;       Semicolon
;\#       Number sign
;\=       Equals sign
;\:       Colon

[METHODHELP1]
NAME=dom.GetObject
TEXT=dom.GetObject (

[METHODHELP2]
NAME=
TEXT=

[METHODHELP3]
NAME=
TEXT=

[METHODHELP4]
NAME=
TEXT=

[METHODHELP5]
NAME=
TEXT=

[METHODHELP6]
NAME=
TEXT=

[METHODHELP7]
NAME=
TEXT=

[METHODHELP8]
NAME=
TEXT=

[METHODHELP9]
NAME=
```

```
TEXT=

[METHODHELP10]
NAME=
TEXT=
```

Sollte fast selbsterklärend sein, mit Name wird defniert, was in der Menüzeile dargestellt wird, mit text wird das definietr, was dann im Editor eingefügt wird. Es muss mindestens ein Eigener MenüHelp definiert sein, damit der menüpunkt unter Methodenhilfe dargestellt wird.

Ebenso ist direkte Verwendung con Kurztasten möglich Shift+Ctrl + 1,2,3...9,0

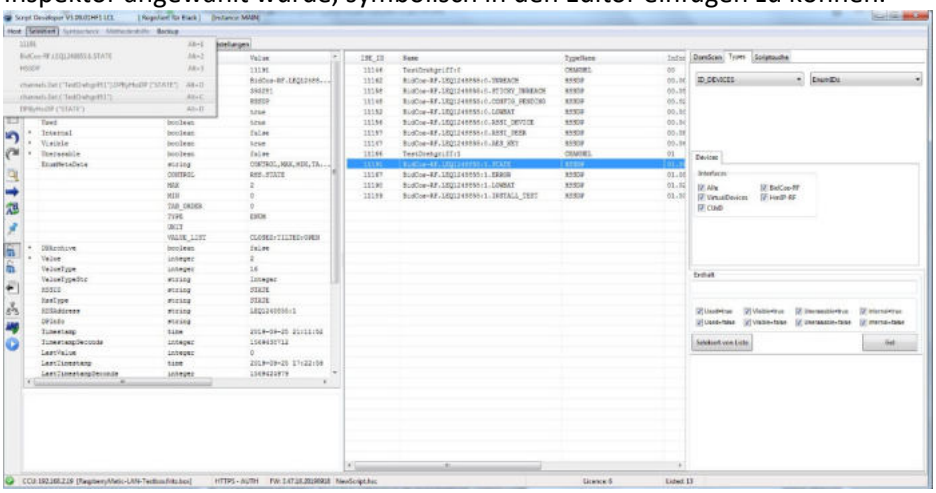
dabei bitte auf die Falle aufpassen !
ich hab extra die Kommentare da mit reingesetzt, sie dort aufgeführten Sonderzeichen müssen gequotet sein rechts des Gleichheitszeichens.

heißt will ich dom.GetObject (" "); als Text schreiben so muss ich eintragen
TEXT=dom.GetObject (\" \")\;

Zweizeilig geht auch, dann muss \n benutz werden.

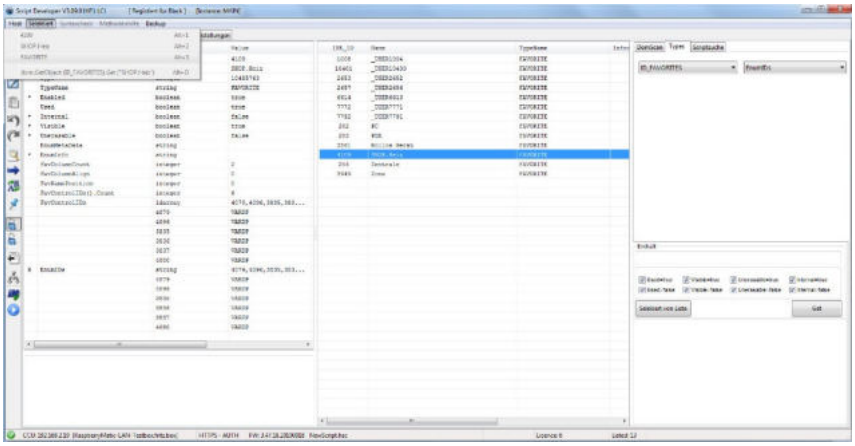
3.8 Symbolischer Zugriff auf selektierte Datenpunkte des Inspektors

Zur Vereinfachung beim Programmieren besteht die Möglichkeit, einen Datenpunkt, welcher im Inspektor angewählt wurde, symbolisch in den Editor einfügen zu können.



Bei einem HSSDP werden 3 Möglichkeiten angeboten (Menüpunkt selektiert)
Vollzugriff über Kanal.HSSDP
Nur Kanal
Nur HSSDP

Fast alle Typen (Favoriten, User, VarDP, AlarmDP) werden automatisch richtig erkannt und dafür der richtige Symbolische Zugriff vorgeschlagen.

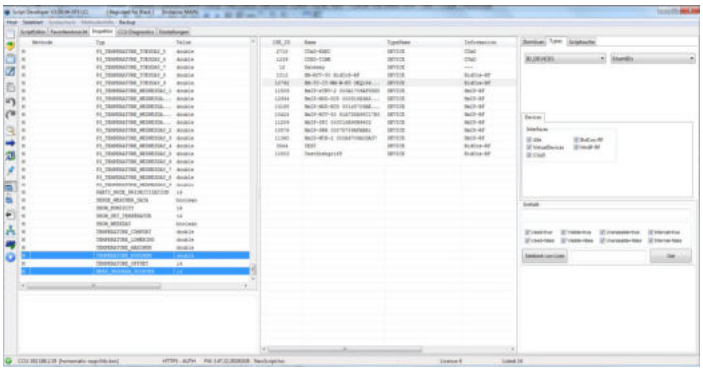


Eine einfache, schnelle und effektive Methode um unnötige Schreibfehler zu vermeiden.

3.9 Automatische Codeerzeugung für Read/Write von Master/Linksets

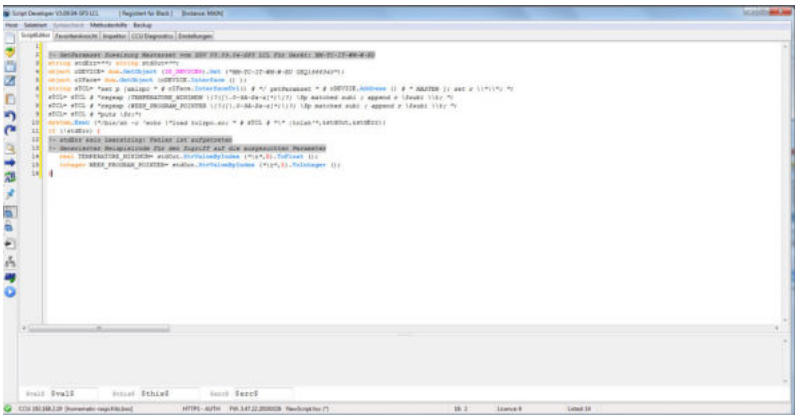
Im Editor kann automatisch Code erzeugt werden für das Schreiben von Master/Linksets (putParamset) oder für das Lesen von Master/Linksets (getParamset).

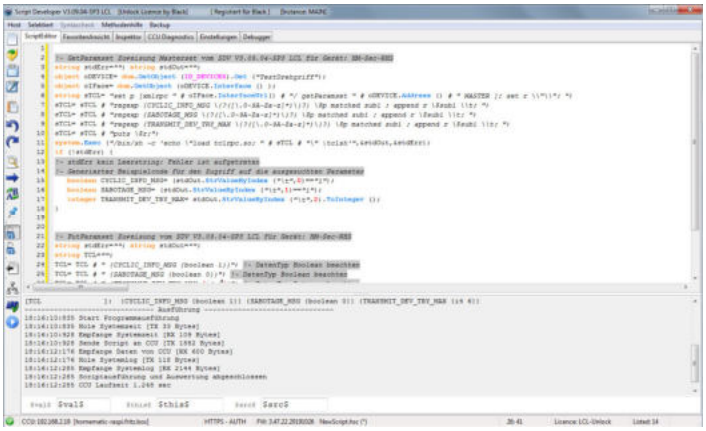
Dazu müssen im Vorfeld im Inspektor die gewünschten Masterparameter (M) oder Linkparameter (L) selektiert werden. Dann hat man im Editor auf der rechten Maustaste im Menü die Auswahl Xmlrpc.PutParamset (Code zum Beschreiben dieser Parameter generieren) bzw xmlrpc.GetParamSet (Code zum Auslesen dieser Parameter generieren)



Dann wechseln in den Editor. Dort wo der Abfragetext in das Skript eingefügt werden soll:
rechte Maustaste - xmlrpc.getParamset

dann erzeugt der SDV den richtigen Code für die Abfrage (Device, Channel wird richtig ausgewählt, die Regex wird richtig definiert und
aus dem Ergebnis werden 2 Skriptvariablen mit dem Namen des Mastersetparameters mit dem richtigen VariablenTyp angelegt.





Natürlich funktioniert die automatische Codeerzeugung für Paramset Master, also die Geräteparameter als auch für die Daten von Direktverknüpfungen

3.10 Nützliche Tastenkombinationen des Editors

Alt – Pfeil rechts: es wird zu dem nächsten folgenden, gleichen Wort gesprungen, welches sich grade unter dem Cursor befindet
Alt – Pfeil Links: es wird zu dem vorherigen gleichen Wort gesprungen, welches sich grade unter dem Cursor befindet

```
Allgemein Tastatur:
=======================================================================
F1                    : Skript auf CCU ausführen  (Oder entsprechend geändert in der INI)
F10                   : Skript auf der CCU testen (Oder entsprechend geändert in der INI)

Datei:
------
Strg + S              : Datei speichern

Eingabemarker:
--------------
Links                :ein Zeichen nach links
Rechts               : ein Zeichen nach rechts
Hoch                 : eine Zeile nach oben
Runter               : eine Zeile nach unten
Bild Hoch            : eine Seite nach oben springen
Bild Runter          : eine Seite nach unten springen
Pos1                 : zum Zeilenanfang springen
End                  : zum Zeilenende springen
Strg + Links         : zum nächsten Wortanfang
Strg + Rechts        : zum nächsten Wortende
Strg + Bild Hoch     : zur ersten vollständig sichtbaren Zeile springen
Strg + Bild Runter   : zur letzten vollständig sichtbaren Zeile springen
Strg + Pos1          : zum Skriptanfang springen
Strg + End           : zum Skriptende springen

Scollen:  (Cursor verbleibt immer im sichtbaren Bereich)
-------------------------------------------------------
Strg + Hoch             : Ansicht nach oben scrollen
Strg + Hinunter         : Ansicht nach unten scrollen

Bearbeiten:
-----------
Strg + A           : alles auswählen
Strg + C           : kopieren                    (Strg + Einfügen)
Strg + V           : einfügen                    (Shift + Einfügen)
Strg + X           : ausschneiden                (Shift + Entfernen)
Strg + J           : synchrones mehrzeiliges Editieren aktivieren (ESC zum Beenden des Modus)
Strg + Backspace   : bis zum Wortanfang löschen
Enter              : Zeile umbrechen
Strg + T           : bis zum Wortende löschen
Strg + Y           : Zeile löschen
Shift + Strg + Y   : bis zum Zeilenende löschen
Strg + Z           : Rückgängig (Undo)
Shift + Strg + Z   : Rückgängig zurücknehmen (Redo)
```

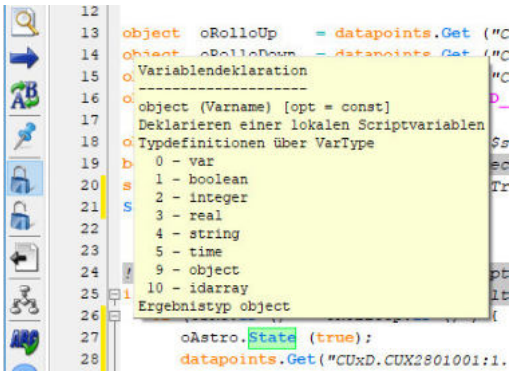
```
Shift + Strg + I    : Block- / Einrücken
Shift + Strg + U    : Block- / Ausrücken

Suchen und Ersetzen
-------------------
Strg + F            : Suchen
Strg + R            : Ersetzen
F3                  : weitersuchen
Shift + F3          : entgegengesetzt weitersuchen
Alt + links         : zum vorherigen Vorkommen des Wortes unterhalb des Cursors springen
Alt + rechts        : zum nächsten Vorkommen des Wortes unterhalb des Cursors springen

Lesezeichen
-----------
Strg + 0            : gehe zum Lesezeichen 0
Strg + 1            : gehe zum Lesezeichen 1
Strg + 2            : gehe zum Lesezeichen 2
Strg + 3            : gehe zum Lesezeichen 3
Strg + 4            : gehe zum Lesezeichen 4
Strg + 5            : gehe zum Lesezeichen 5
Strg + 6            : gehe zum Lesezeichen 6
Strg + 7            : gehe zum Lesezeichen 7
Strg + 8            : gehe zum Lesezeichen 8
Strg + 9            : gehe zum Lesezeichen 9

Spezielle Kombinationen des SDV
===============================

Spezielle Anzeigefunktionen
---------------------------
Strg + H            : Hint Funktionalität im Editor ein bzw ausschalten

Codevervollständigung
---------------------
Strg + Shift + S    : Completion Auswahl Methoden für Strings
Strl + Alt + I      : Completion Auswahl alle ID_xxx Konstanten
Strg + Alt + O      : Completion Auswahl alle OT_xxx Konstanten
Strg + Alt + M      : Completion Auswahl alle Methoden
Strg + Alt + V      : Completion Auswahl: verwendete Skriptvariablen
Strg + Alt + S      : Completion Auswahl: verwendete Systemvariablen
Strg + Alt + R      : Completion Auswahl: verwendete Räume
Strg + Alt + G      : Completion Auswahl: verwendete Gewerke
Strg + Alt + D      : Completion Auswahl: verwendete Devices
Strg + Alt + P      : Completion Auswahl: verwendete Programme
Strg + Alt + Space  : Auswahldialog für das Generieren des qualifizierten Zugriff
Strg + Shift + Space: Autocompletion Auswahldialog
Strg + Shift + F1   : Autocompletion 1. frei definierter, einzufügender Textbaustein
Strg + Shift + F2   : Autocompletion 2. frei definierter, einzufügender Textbaustein
Strg + Shift + F3   : Autocompletion 3. frei definierter, einzufügender Textbaustein
Strg + Shift + F4   : Autocompletion 4. frei definierter, einzufügender Textbaustein
Strg + Shift + F5   : Autocompletion 5. frei definierter, einzufügender Textbaustein
Strg + Shift + F6   : Autocompletion 6. frei definierter, einzufügender Textbaustein
Strg + Shift + F7   : Autocompletion 7. frei definierter, einzufügender Textbaustein
Strg + Shift + F8   : Autocompletion 8. frei definierter, einzufügender Textbaustein
Strg + Shift + F9   : Autocompletion 9. frei definierter, einzufügender Textbaustein
Strg + Shift + F10  : Autocompletion 10. frei definierter, einzufügender Textbaustein

Einfügeoptionen aus dem Inspektor (bei angewähltem Objekt im Inspektor verfügbar)
---------------------------------------------------------------------------------
Alt + 0             : Einfügen der ID aus der Listenauswahl
Alt + 1             : Einfügen des Namens aus der Listenauswahl
Alt + 3             : Einfügen des Objekttypes aus der Listenauswahl
Alt + O             : Einfügen des qualifizierten Zugriffs des Objekttypes aus der
Listenauswahl
Alt + M             : Einfügen des Methodennamens aus der selektierten Zeile des Detailview
Alt + T             : Einfügen des Types aus der selektierten Zeile des Detailview
Alt + V             : Einfügen des Values aus der selektierten Zeile des Detailview
Alt + S             : Einfügen des Specials aus der selektierten Zeile des Detailview
```

```
================================================================================================

Mausbedienung
-------------
Links Einfachclick  : Cursor positionieren
Links Doppelclick   : Wort markieren
Links Dreifachklick : Zeile markieren
Links Vierfachklick : Block auswählen, der zwischen Leerzeilen eingeschlossen ist

SDV Maus Sonderfunktionen
-------------------------
Alt + rechte Maustaste       : der zwischen { } liegende Block wird farblich markiert
Ctrl + Alt + rechte Maustaste: der zwischen ( ) liegende Block wird farblich markiert
```

3.11 Editor Hint-Funktionalitäten

Durch die Hint Funktionalität werden Kontextbezogene Hilfestellungen zu dem Wort unterhalb der Maus dargestellt:

Bzw

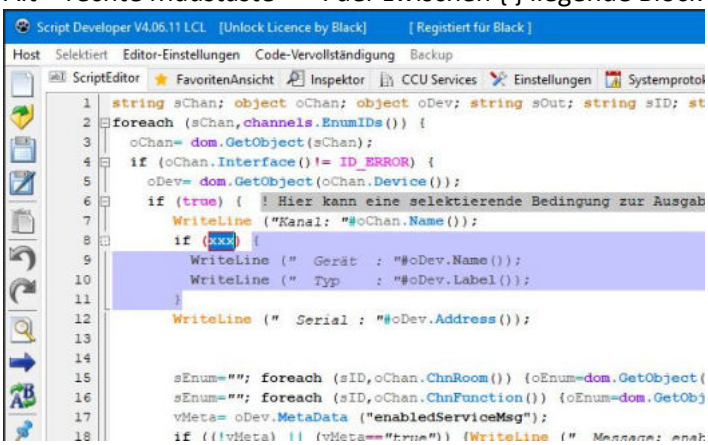


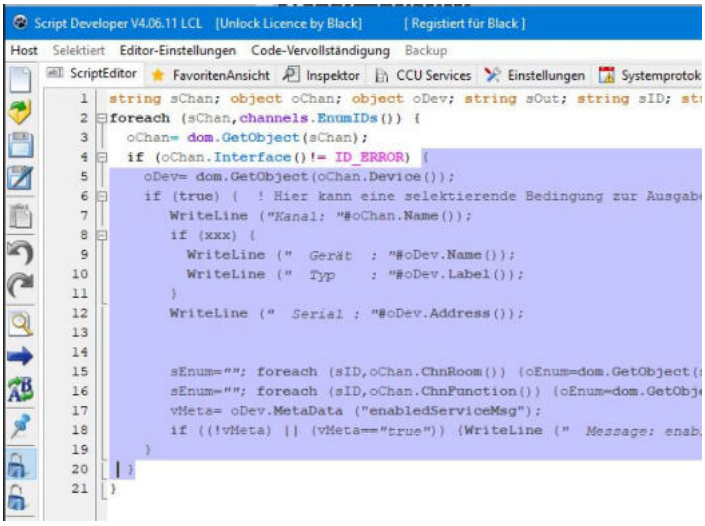
Hints werden kontinuierlich erweitert

3.12 Blockdarstellung

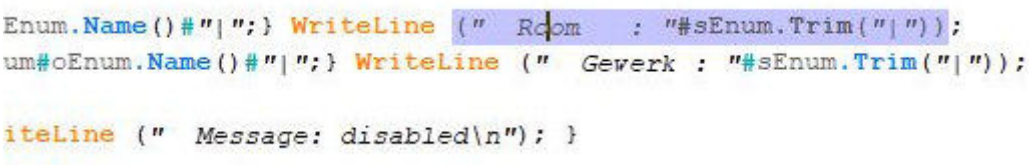
Mit den folgenden Tasten/Mauskombinationen kann an der aktuellen Cursorposition der aktuell wirksame {..} Block (mit alt+recht Maustaste bzw der aktuelle wirksame runde Klammenblock eingefärbt werden.

Alt + rechte Maustaste : der zwischen { } liegende Block wird farblich markiert

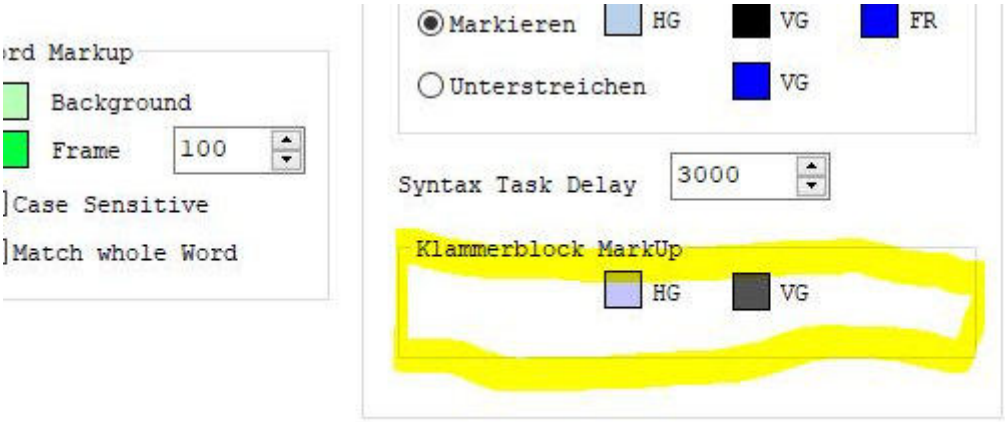




Ctrl + Alt + rechte Maustaste: der zwischen ( ) liegende Block wird farblich markiert

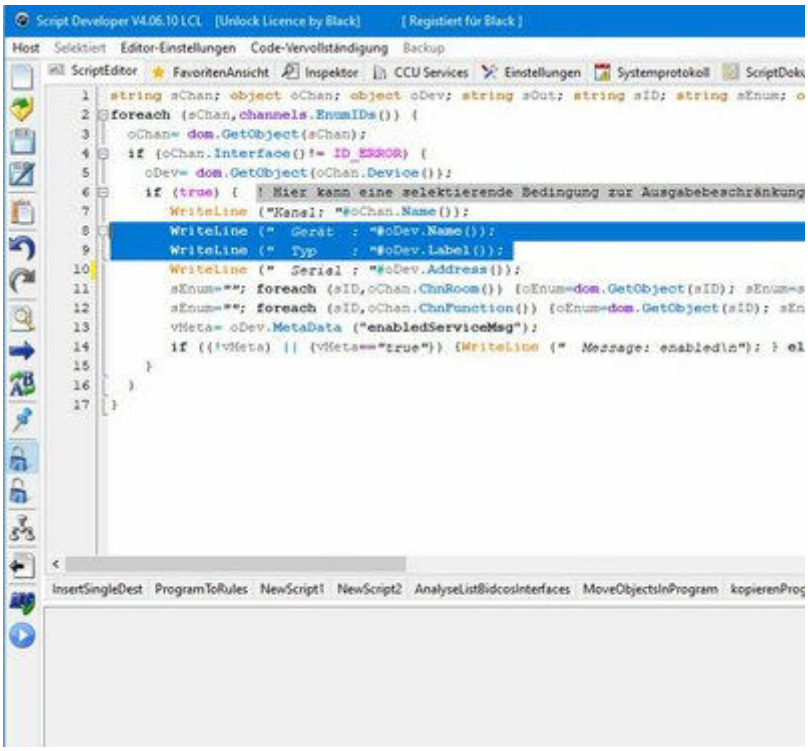


Die Art der Einfärbung kann im Setup Menü Editor eingestellt werden.

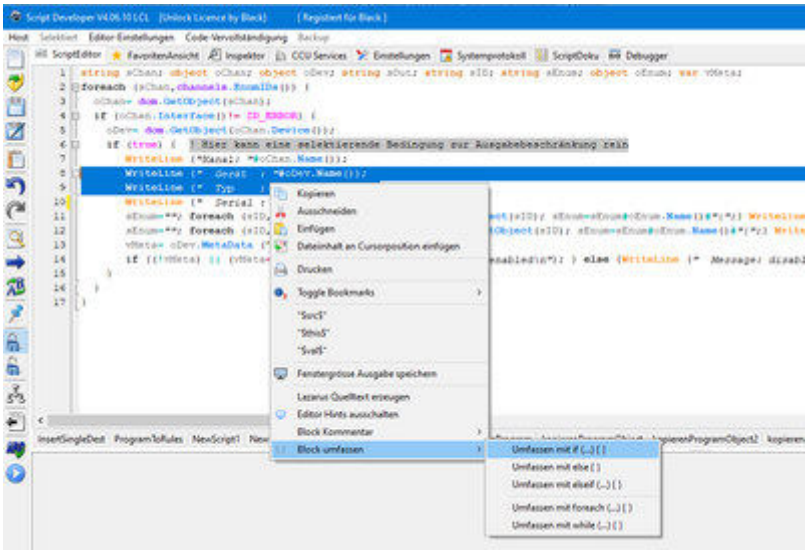


3.13 Umfassen eines Textblockes

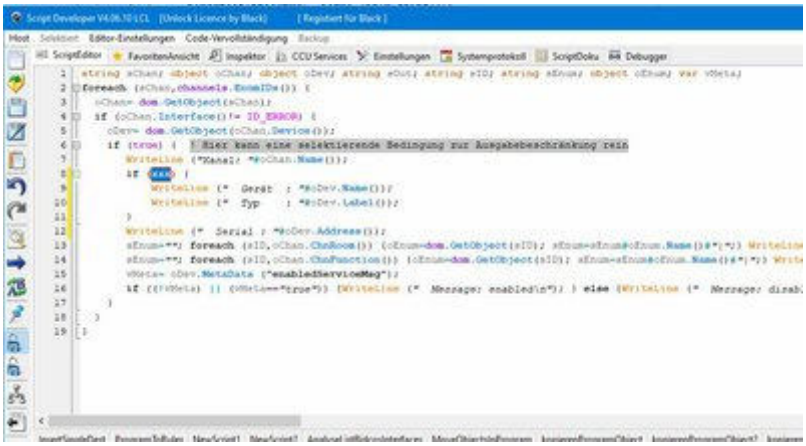
Praktisches Beispiel: gegeben sei dieses beispielhafte Programm:



Die Blau markieren Blöcke möchte ich beispielsweise in eine neu zu erstellende if Bedingung einkapseln. das geht nun schnell und komfortabel mit der Umfassen Funktionalität. Dazu rechte Maustaste, Editorpopup Menü öffnet sich und unten auf Umfassen gehen und auswählen "Umfassen mit if"



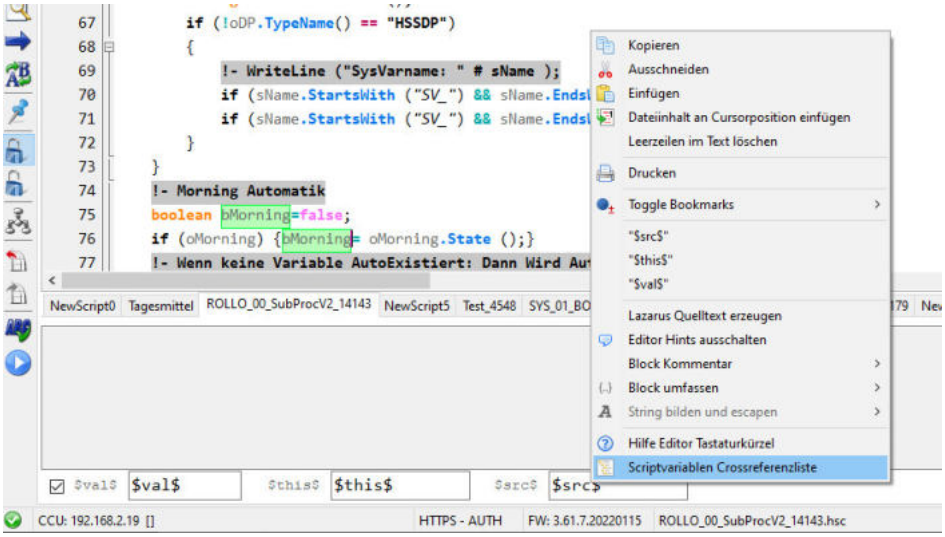
vor den Block wird nun eine if Zeile mit { eingefügt, der Block selber wird nach den eingestellten Werten recht eingerückt und am Ende des Blockes wird formatiert auch wieder eine } hinzugefügt. Der Curor befindet sich dann direkt in der if Bedingung, so dass diese direkt angepasst werden kann.



Eine schnelle Methode, um nachträgliche Änderungen und Erweiterungen in einem bestehenden Skript hinzuzufügen.

3.14 Crossreferenzliste Skriptvariablen

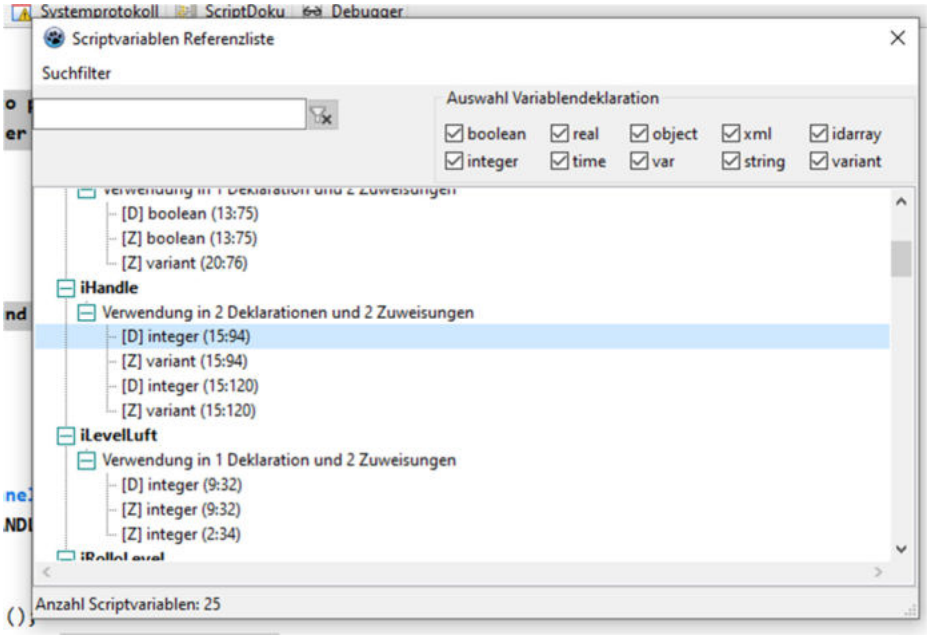
Im Editormenü erzeugt der Menüpunkt „Skriptvariablen Crossreferenzliste“ die Verwendungsliste der Skripvariablen.



Es wird dabei eine Liste angelegt von allen Stellen, wo Deklarierend oder Schreibend auf die entsprechende Variable zugegriffen wird.

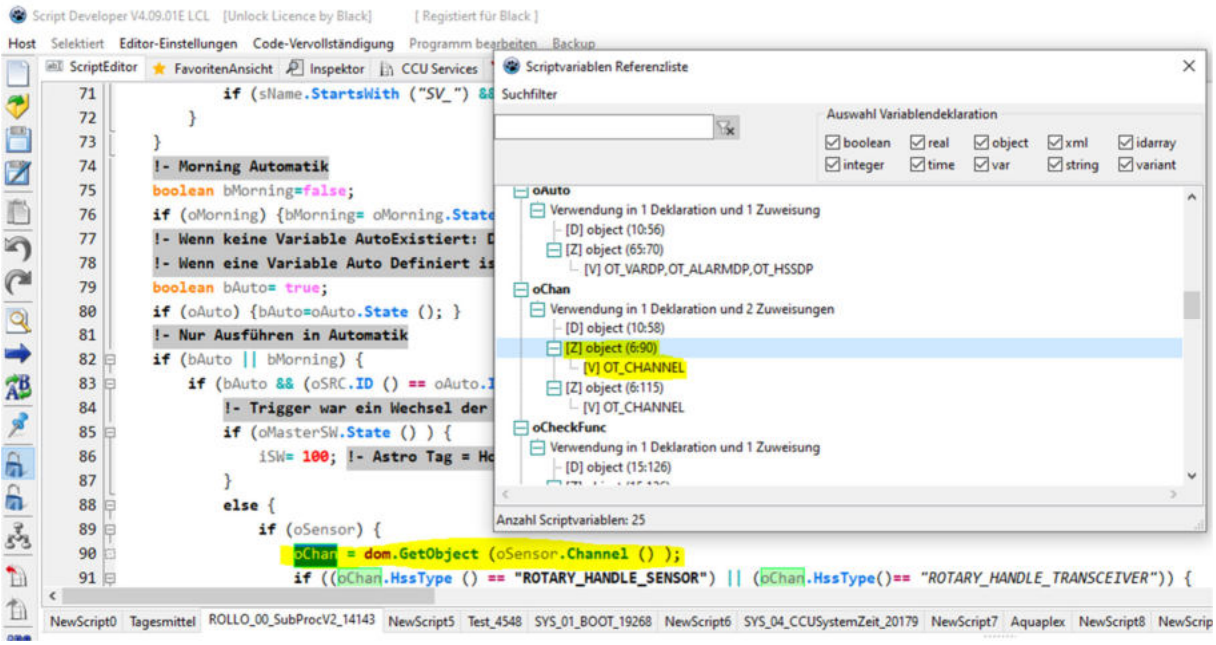
Gelistet werden alle vorkommenden Stellen als
Deklaration [D]
Zuweisung [Z]
mit der Angabe der Textstelle. Click auf die Position markiert die Stelle im Editor

Angezeigt wird auch, als was der SDV den Variablentyp an der Stelle kalkuliert.



Der SDV braucht für seine Methodenvorhersage schon ein paar mehr Informationen als nur die 9 Variablentypen, wenn ein Ergebnis ein bestimmtes Objekt beinhaltet, so wird dieses auch in den Variablen berücksichtigt, siehe in dem Beispiel.

Hier wird die Zuweisung als eine Variable von Typ Object erkannt, weitergehend aber ergibt sich durch die AST Auswertung und der Abstraction Unit, das hierbei dies nicht nur ein object ist, sondern dies noch feiner gradual betrachtet werden kann, es wird ein OT_CHANNEL sein.



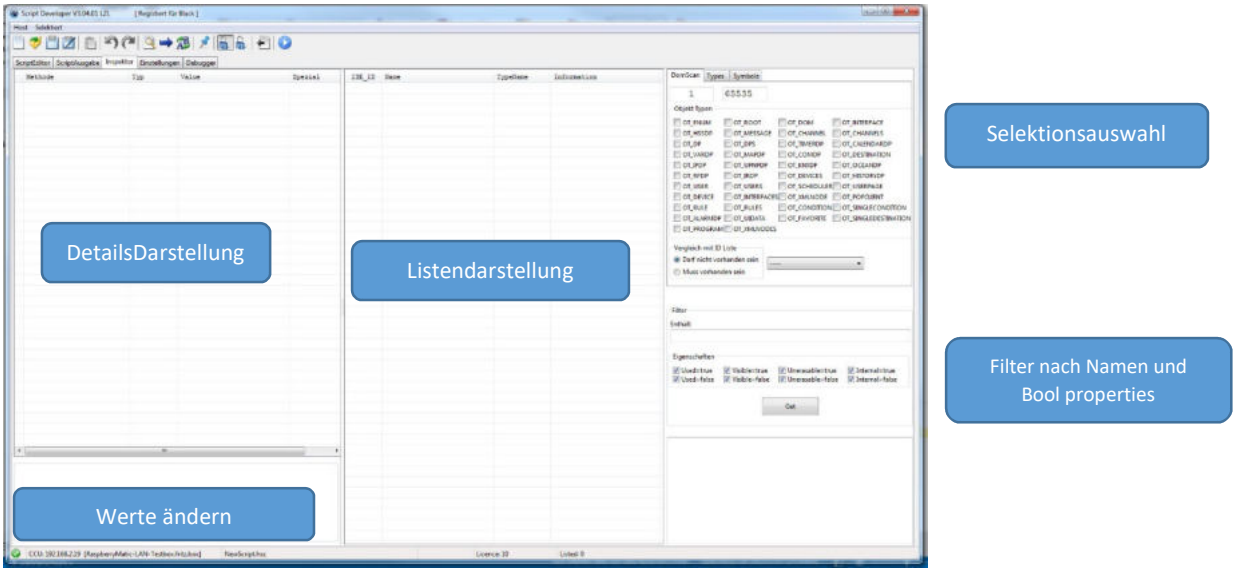
Die Auswahl lässt sich filtern nach Variablenname (anwählbar ob "Name muss so Anfangen" oder "Suchbegriff irgendwo im Namen" und Variablentypen:
Eine Variable wird angezeigt, wenn eine !! ihrer Zuweisungen/Deklarationen in der dem Checkboxfeld vorkommt.

Die Größe des Fensters sollte persistiert werden beim Beenden.

4 Inspektor

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

Es existieren verschiedene Selektionskriterien.



Filteroptionen:

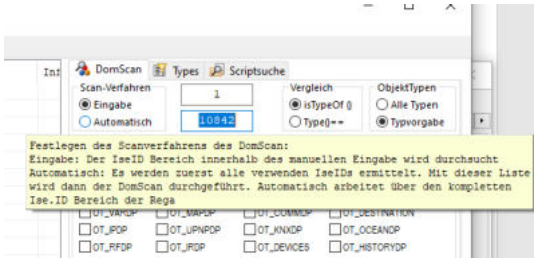
Auswahl der Aufzählungen (Räume, Gewerke, Favoriten, Interfaces , Systemvariablen sind bisher implementiert)

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

4.1 Selektionswahl: DomScan



Eingabe des Scan Bereiches der IseID´s (hier von 1-65535)
Doppelklick auf das Eingabefeld der oberen Grenze ermittelt automatisch die höchste verwendete Ise-ID. Beschreibung der Strategien unter 4.1.1 bzw. unter Setup Inspektor

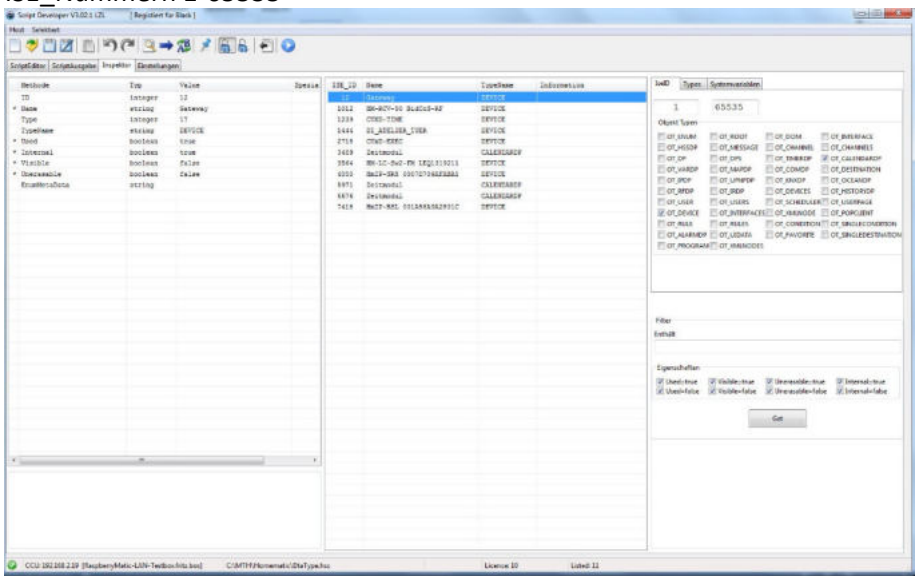
Achtung
Schrott Eingabe von Millionenwerten werden die CCU lahmlegen. Der SDV ist schließlich kein Spielzeug, sondern ein Werkzeug, man sollte schon wissen, was man tut.
Damit ein Objekt selektiert wird, muss es die angeklickte Objekteigenschaft haben.
Mehrfachangaben sind möglich

ObjectType Auswahl:

- Alle Typen: Es wird im Zahlenbereich gescannt ohne Objecttypen zu berücksichtigen
- Nach Objecttypen: Das Objekt muss dem oder den angewählten Typen entsprechen

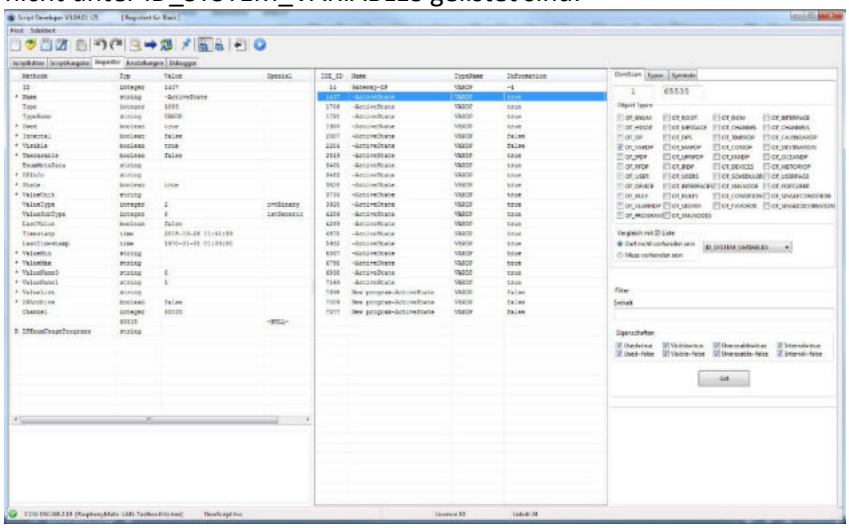
Vergleich
- IsTypeOf () : Der Vergleich findet mit der Methode TypeOf statt. OT_Object findet dann z.B. ALLE Objecte, weil alle Objecte vom Type OT_OBJECTS abgeleitet sind
- Type()== : Das Object muss genau dieser Typ sein. Beispielsweise findet die Suche nach OT_OBJECTS auch nur genau die Objecte, die vom Type OT_OBJECTS sind.

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



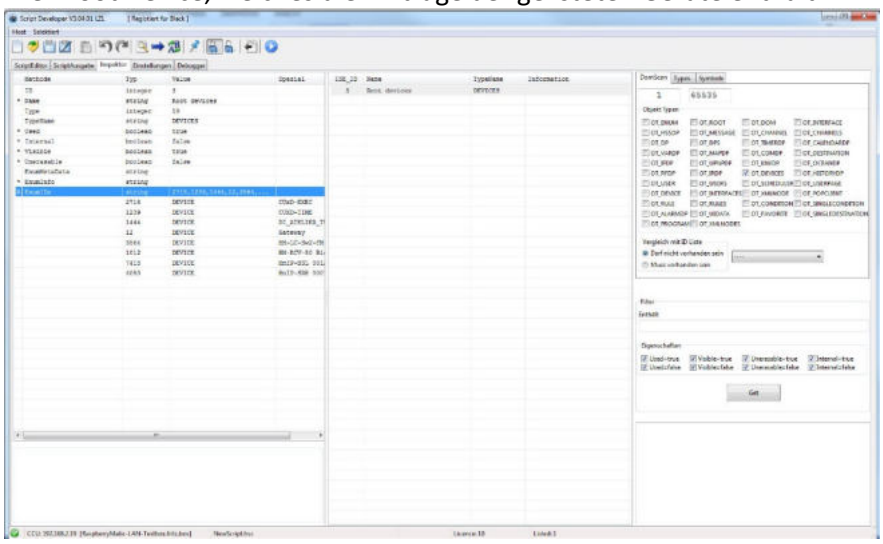
Anklicken eines Wertes in der Listdarstellung öffnet die Detaildarstellung des Objektes.

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:



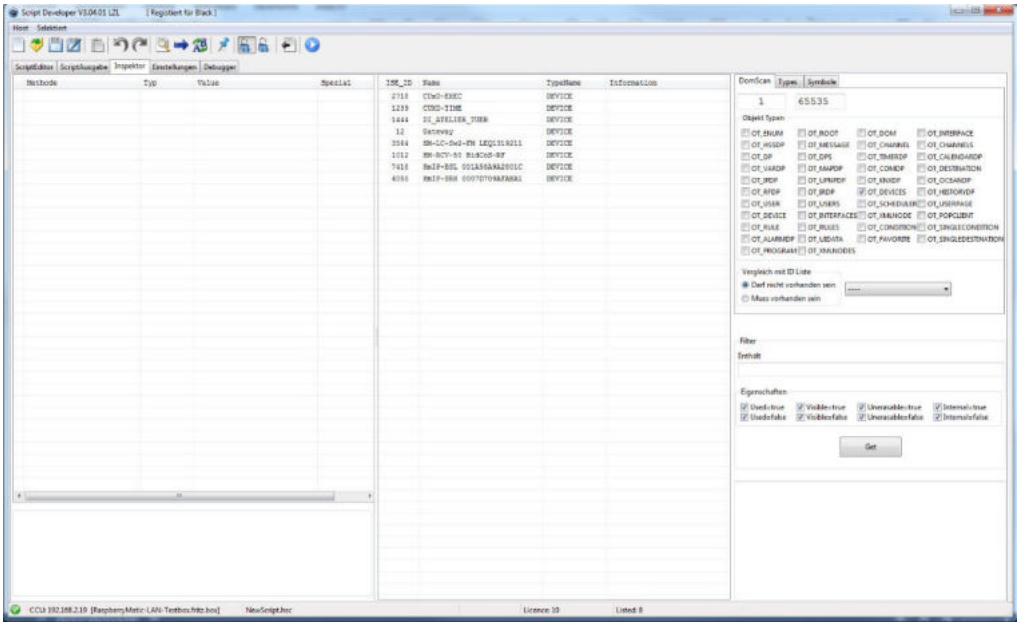
Hier tauchen dann einige interne Datenpunkte auf, im dem Falle sind die –ActiveState keine Leichen, sondern der Anwahl Punkt Programm aktiv unter Programme.

Weiterhin ist auch rekursives Arbeiten nun möglich
Hier Root Device, welches die Einträge der gelisteten Geräte enthält.



Das „R“ in der ersten Spalte zeigt an, dass ein Rekursiver Aufruf möglich ist. Doppelklicken löst und die Liste auf und lädt diese in die Listenauswahl. Ist es nur ein Wert, so wird dieser Wert auch direkt in der Detailansicht geöffnet.

Die neue Auswahllist steht nun zur weiteren Bearbeitung bereit.



4.1.1 Strategien für den Scan der Rega

Ermittlungsverfahren Iteration über die IseIDs
nötig geworden durch die Änderungen in der Strategie über die Vergabe neuer IseIDs
Die Einstellung hier mit Bedacht vornehmen, sie gelten SDV Weit.

RegaDom - MaxIse
Hier wird über eine Linux Befehlsfolge die Maximale IseID in der Datei Regadom gesucht. Nachteil: es muss persistiert werden, das dom.Save()dauert aber. Es wird in Testscheiben über alle Nummern iteriert, auch Lücken müssen untersucht werden

Regadom - Used Blocks
Hier wird durch eine Linux Befehlsfolge (Danke an Jerome für die Hilfe seinerzeit) die Blockbelegung der Rega analysiert. Nachteil: es muss persistiert werden, das dom.Save() dauert aber. Es wird dann nur über die gefundenen IDs iteriert.

Dom.CreateObject - MaxIse
Achtung- Nur mit aktuellen Regas zu verwenden, wo der Allready in MAP Error durch die neue Strategie gefixt wurde. Es wird dann in Testscheiben über alle Nummern
iteriert, auch Lücken müssen untersucht werden.

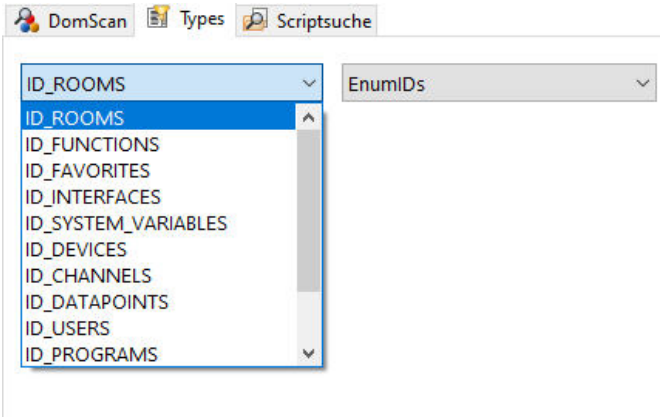
Empfehlung:
==========
Neue Rega: CreateObject
Alte Rega: Auf keinen Fall CreateObject

Neue Rega und riesig aufgebläht so dass die Analyse der Lücken zeit kostet:
Used Blocks

Alte Rega oder CCU2:
Used Blocks

RegaDom MaxIse ist aus Kompatibilitätsgründen noch drin, wurde früher vom SDV verwendet.

4.2 Selektionskriterium Types

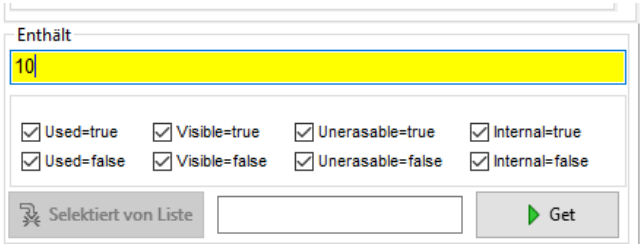


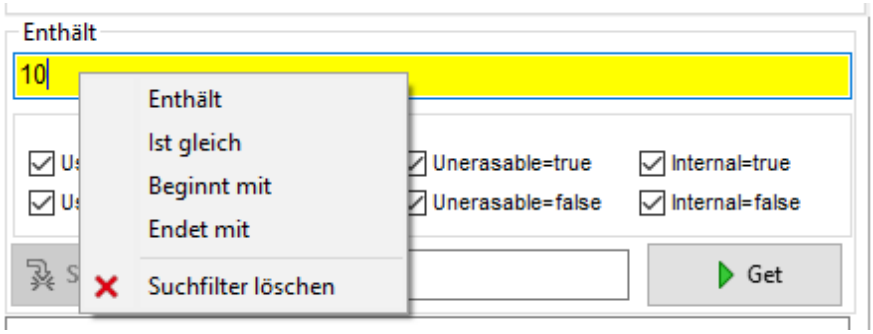
Hierbei wird wie schon in der Version 3.2 in festen Bereichen gesucht und selektiert. Schneller und einfacher zu handeln als die Objekt Selektion, dafür nicht so umfangreich.

4.3 Zusätzliche Selektionsbedingungen

4.3.1 Enthält Filter

Über den Enthält-Filter lässt sich bei der Auswahl über die Property NAME der gefundenen Objekte selektieren. Wenn ein Enthält Kriterium Selektiert ist wird diese Zeile gelb dargestellt, damit dieses auch sichtbar ins Auge fällt.

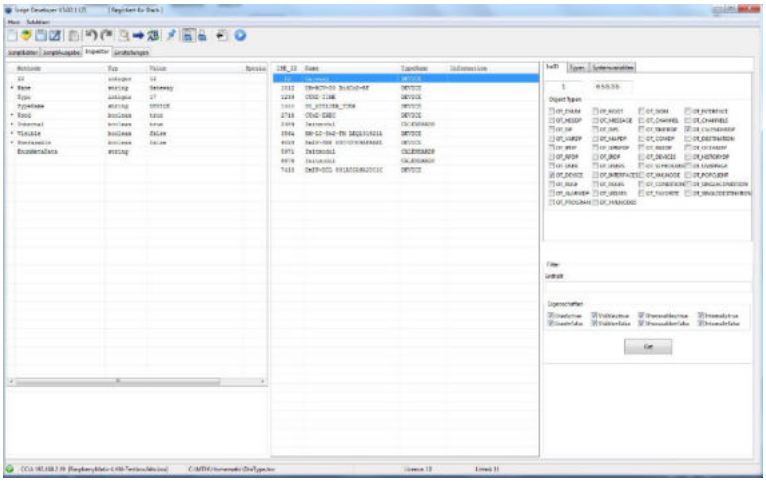


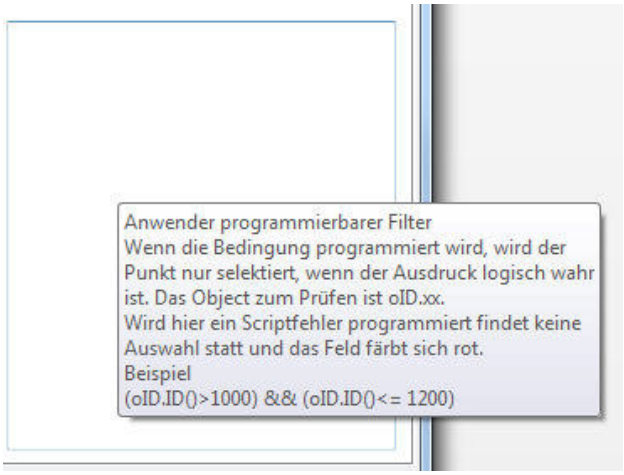


Enthält: Die Zeichenfolge muss irgendwo in NAME enthalten sein
Ist gleich: NAME muss der Zeichenfolge entsprechen
Beginnt mit: NAME muss mit der Zeichenfolge beginnen
Endet mit: NAME muss mit der Zeichenfolge enden

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

4.3.2 Anwenderdefinierte Filter





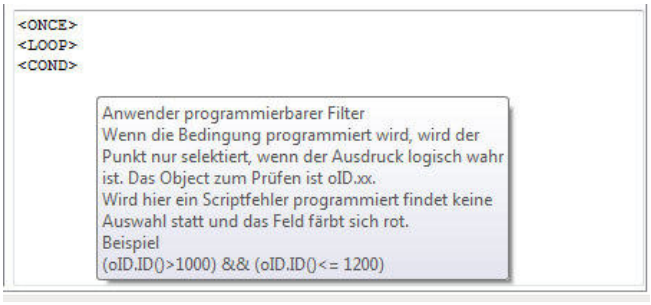
Zusätzliches Feld für Selektionsbedingen

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 Abfrageskriptes 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 () );
}
}
```

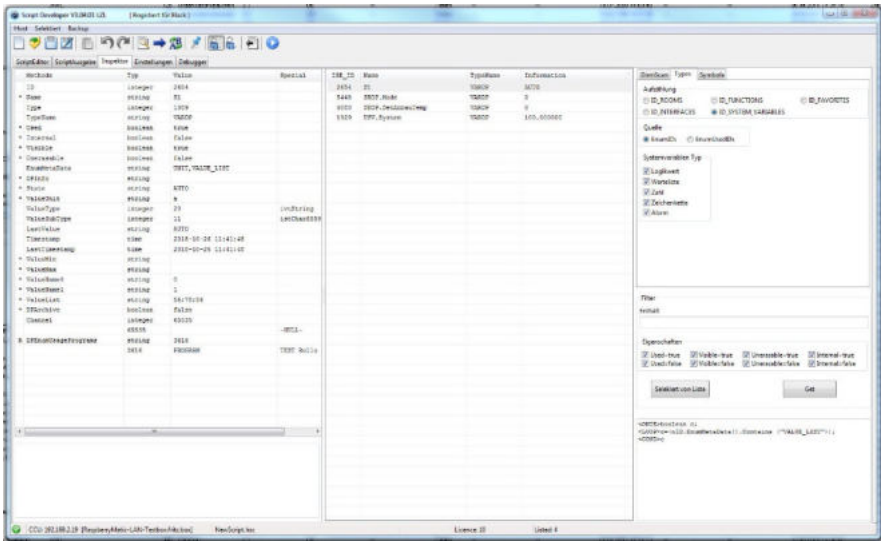
An diesem Kleinen Filter mal verdeutlicht:

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

Daraus generiert der Scriptdeveloper folgende interne Filterabfrage:

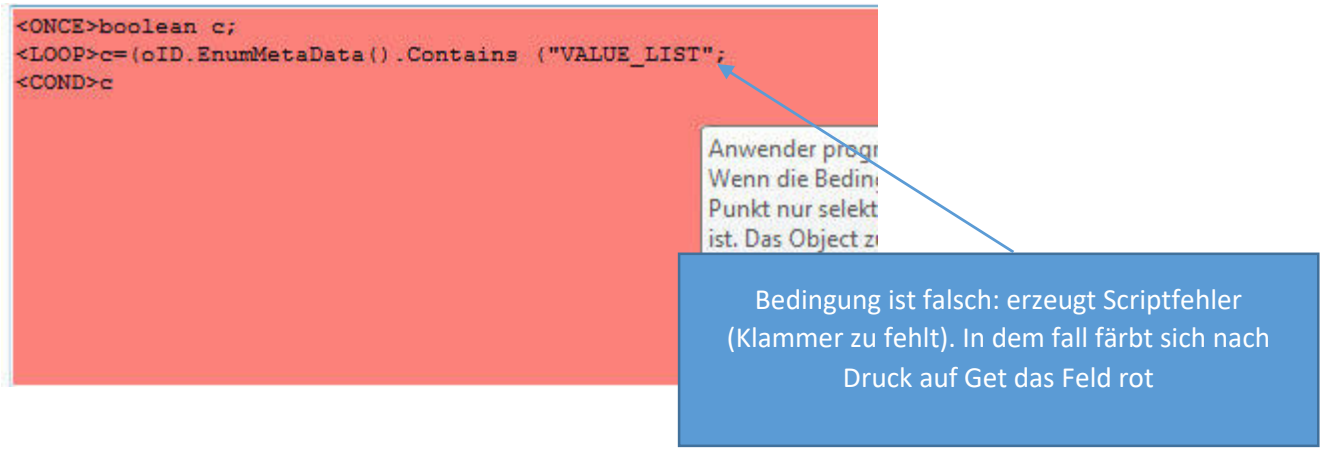
```
object oID;
string s;
boolean c;

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 Filder, auch Finger weg
string done: auch interne Benutzung, auch Finger weg

Die Filterbedingung wird in HM Skript 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:



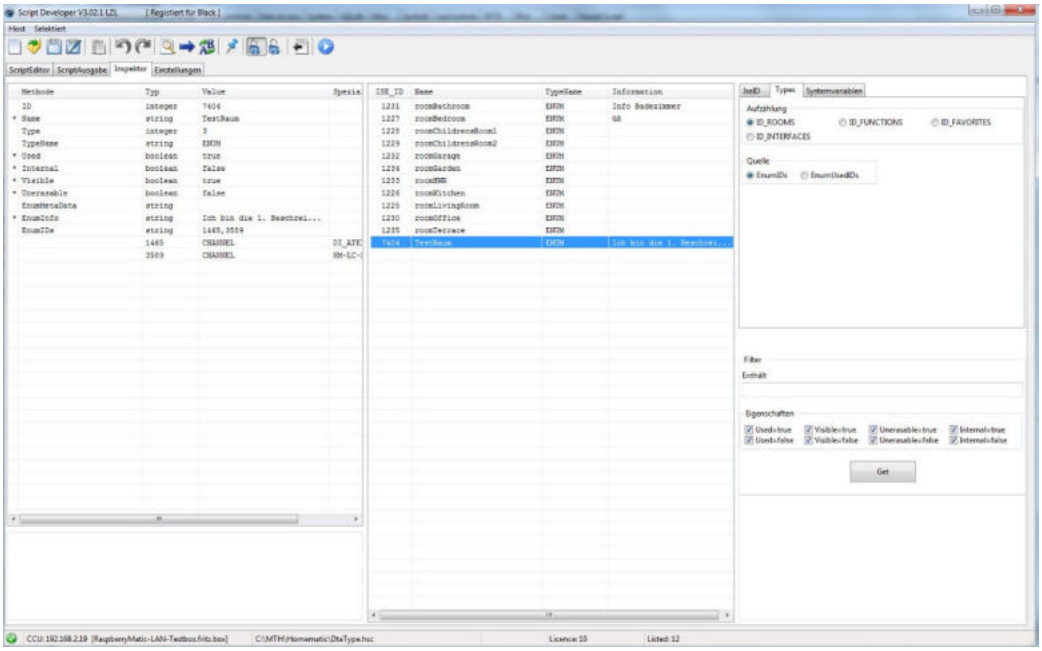
Der rar Datei liegen Standardmäßig nun schon mal 2 Filter bei:
PROGRAM_GeisterProg_CopyID - Filter um Geisterprogramme mit gesetzter CopyID zu finden
SYSVAR_VerwaisterChannel – Filter um Systemvariablen zu finden, deren Channel verweis in Nirvana zeigt

Durch Click auf die Beschreibungszeile 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 Sortieralgorithmen suboptimal arbeiteten, hat das ListView Object neue selektive Sortieralgorithmen 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. (Ab Version 3.08.08 verbesserter Sortieralgorithmus, hier wird durch Klick auf die Spalte zwischen Sortieren aufsteigend und Sortieren absteigend getoggelt.)



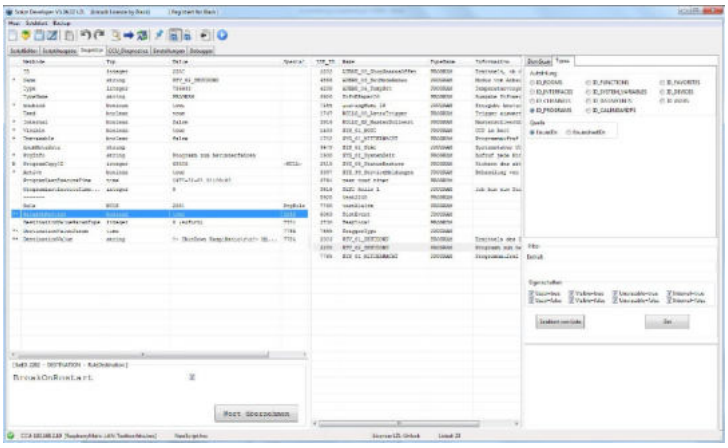
Die Spaltenbreite kann sowohl in Normalansicht als auch in Maximize separat eingestellt werden (Das Programm sollte sich die Breiten merken und je nach Darstellungsart automatisch wieder einstellen, sollte…)

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)

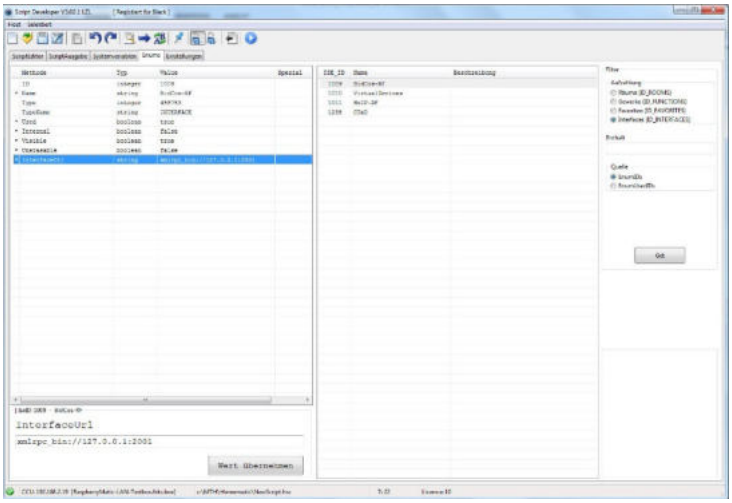
Properties, die in der ersten Zeile mit einem Stern (*) gekennzeichnet sind, können in ihrem Wert geändert werden.

Ab version 3.06.04: Properties, die in der ersten Zeile mit einem Doppelstern (**) gekennzeichnet sind, können ebenfalls verändert werden. Der Doppelstern bedeutet dabei, die geänderte Property nicht element der Haupt ID ist, sondern sich rekursiv in einer untergeordneten Rekursionsebene befand.

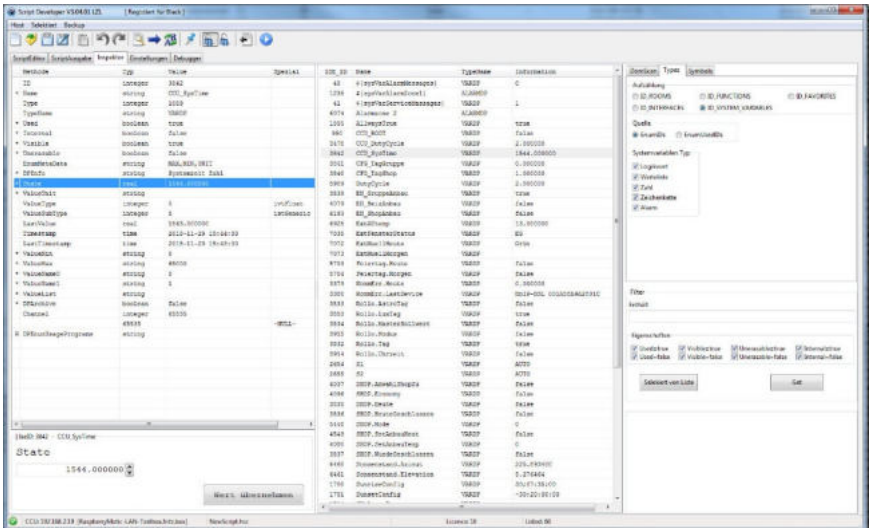


BreakOnRestart ist hier nicht Member von OT_PROGRAM sondern rekursiv von der entsprechenden Rule bzw SubRule.

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.



4.3.3 Schnellausführung von Get

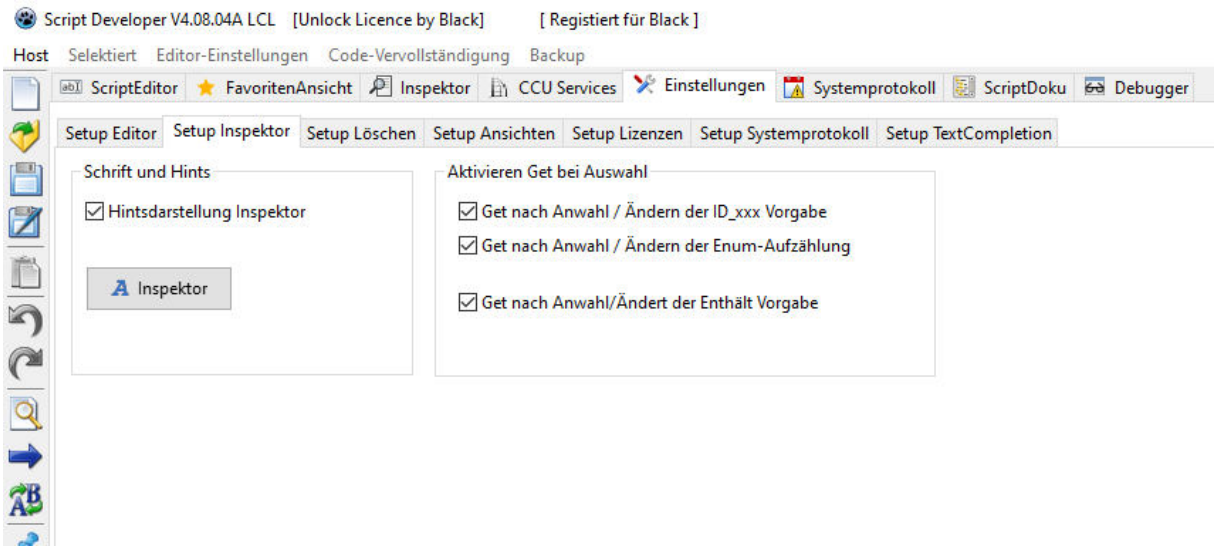
Get kann auf folgenden Wegen ausgeführt werden:

Druck auf den Button
Druck auf F1 im Inspektor (bzw der Taste, die für Skript ausführen definiert wurde)

Ändern der IDxx Eigenschaft (Wenn im Setup eingestellt)

Ändern der Enum Suchkriterien (Wenn im Setup eingestellt)

Ändern des Enthält Textes bzw der Bedingung (Beginnt, Endet, enthält) Eingestellt wird dies hier:

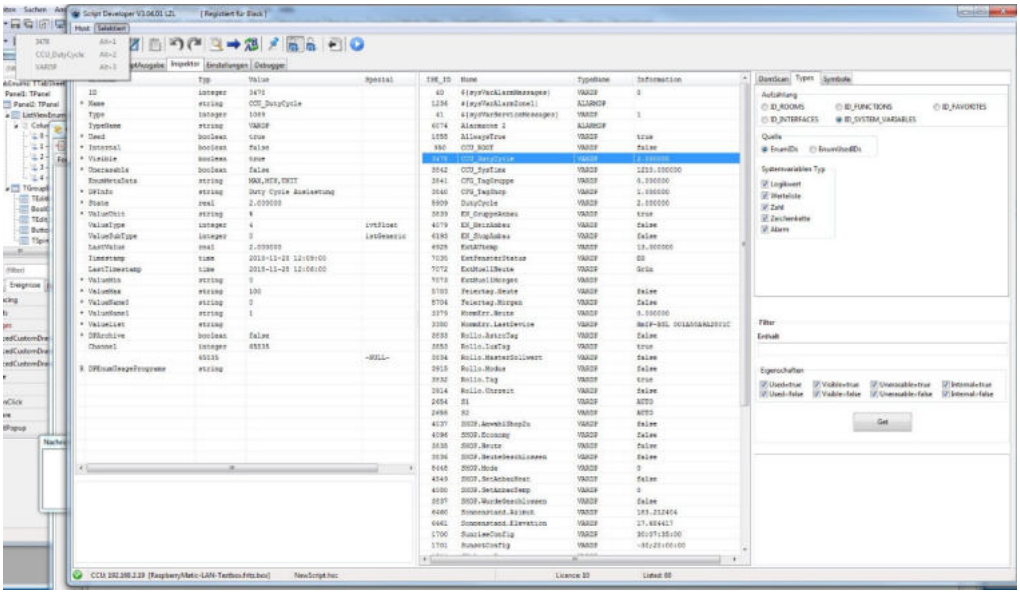


4.4 Daten aus Inspektor in Editor übernehmen

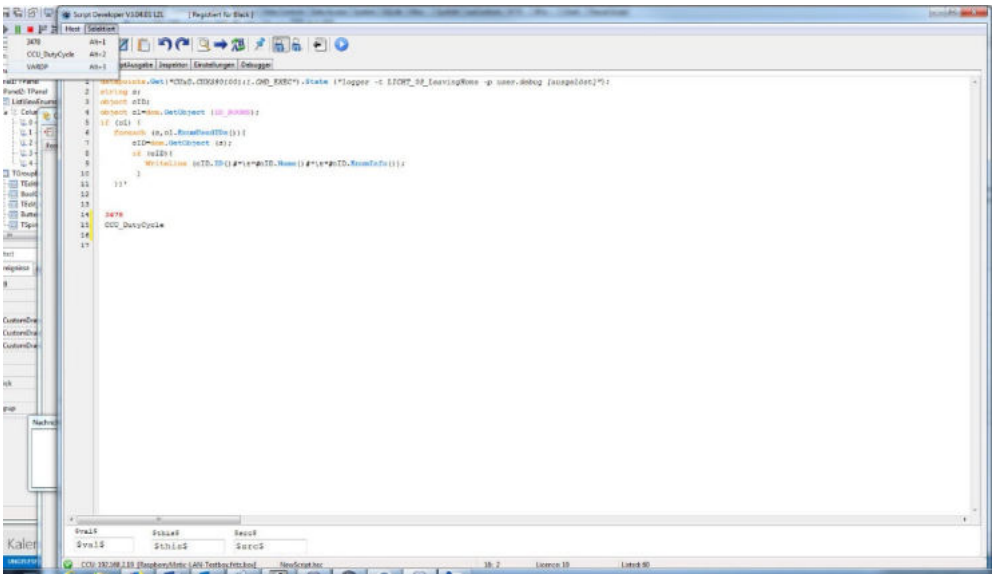
Zur schnelleren und auch möglichst fehlerfreien Bearbeitung besteht die Möglichkeit, Daten aus dem Inspektor direkt in den Editor zu übernehmen.

Immer wenn im Inspektor in den beiden Listviews auf eine Eigenschaft geklickt wurde, stehen diese Daten dann im Editor unter Selektiert zur 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, Akt-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

Ab 3.06.01

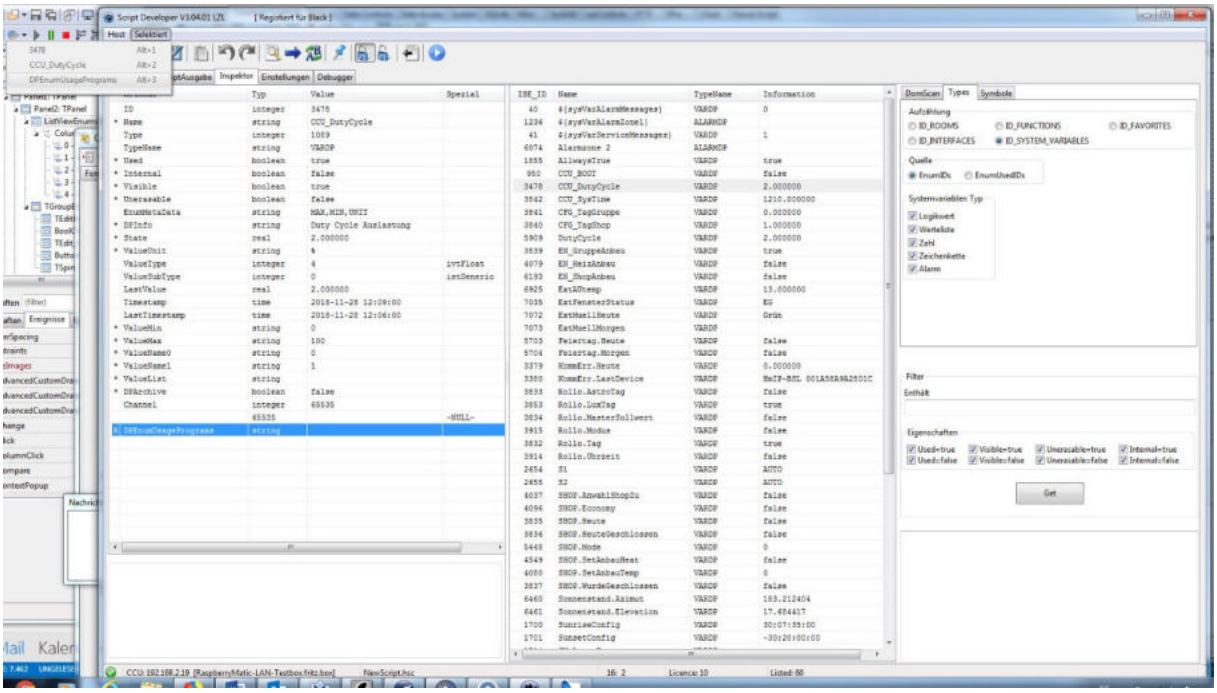
Mit den Tasten lassen sich folgende Spalteninhalte, welche vorher im Detailfeld selektiert worden sind, in den Editor übernehmen

Alt+m: Inhalt von Methode

Alt+t: Inhalt von typ

Alt+v: Inhalt von Value

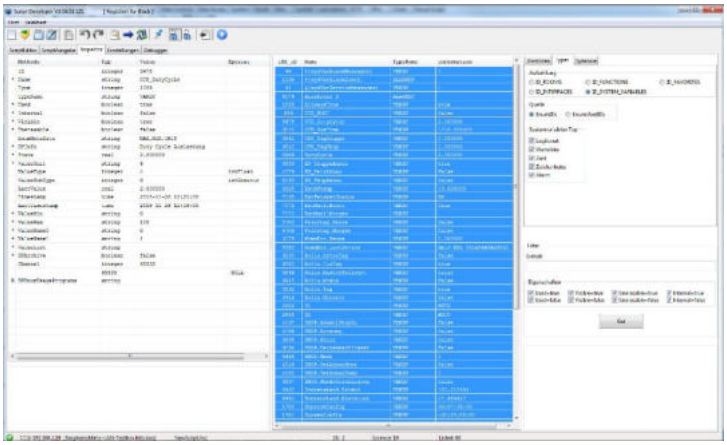
Alt+p: Inhalt von Spezial



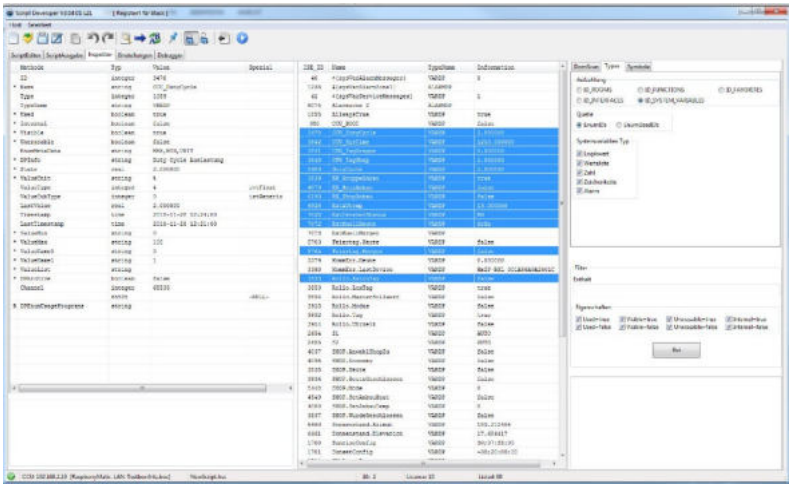
4.4.1 Mehrfachauswahl als Enum String

Es lassen sich im Hauptauswahlfeld Mehrfachselektionen vornehmen.

Ctrl-A : alle auswahlen.



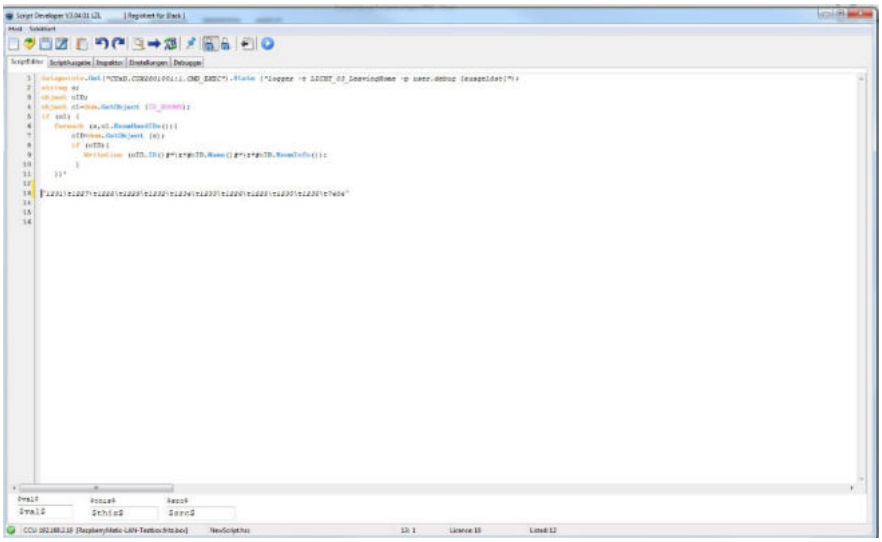
Oder die Übliche Mausbedienung:



Ab Version 3.07.07.

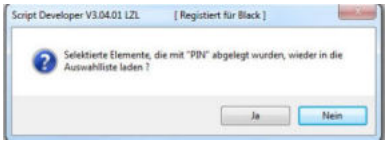
In der Listendarstellung kann nun mit den Cursortasten navigiert werden, ein Druck auf Return öffnet von dem selektierten Objekt die Detaildarstellung.

Mit der  Taste merkt sich der Inspektor die Auswahl, welche sich dann Skript 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 Skript 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:

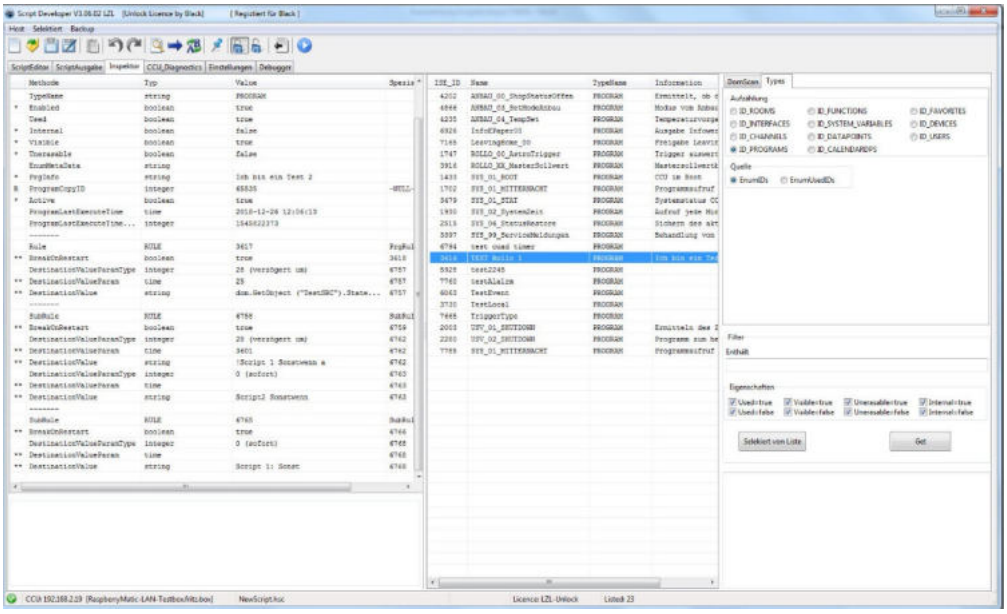
Mit rechter Maustaste im Mittleren Feld die Funktion „Einfügen aus Pinliste“ anwählen und die Sicherheitsabfrage bestätigen,



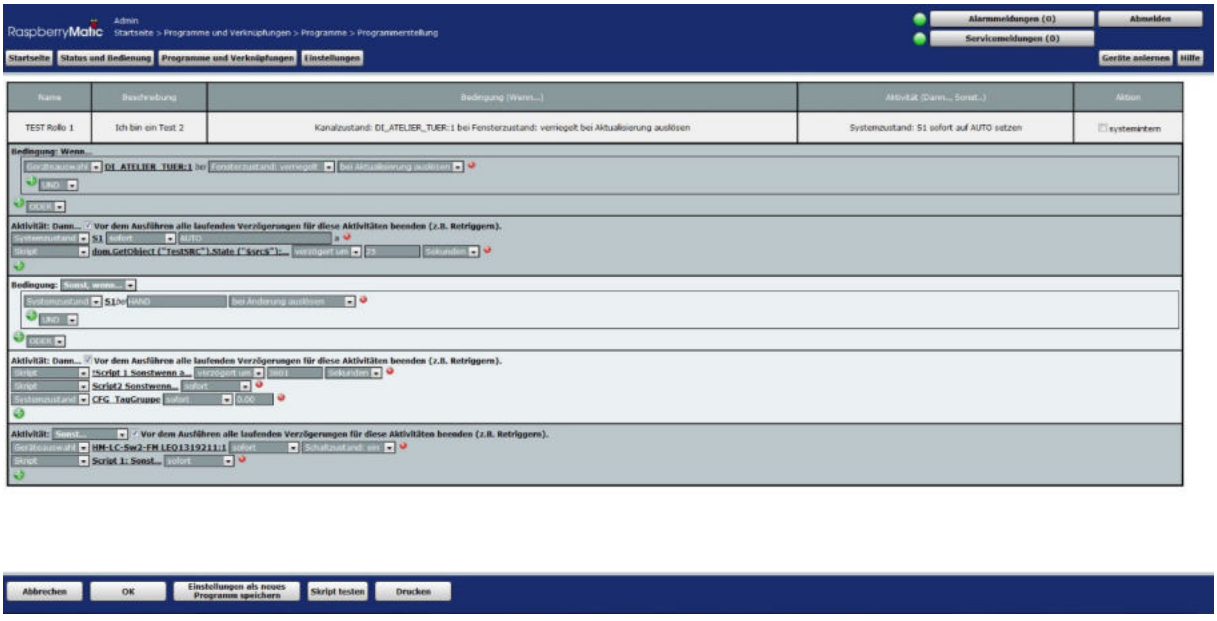
dann befinden sich die Selektierten Elemente wieder im mittleren Feld.

4.4.2 Übernahme von einem Skript aus einem Programm direkt in den Editor

In der Darstellung eines Programmes werden in der Detaildarstellung auch die Rules und Subrules mit ihren Destinations/SingleDestinations aufgelöst, wenn diese ein Skript enthalten



Dies entspricht der tabellarischen Darstellung dieses Programmes

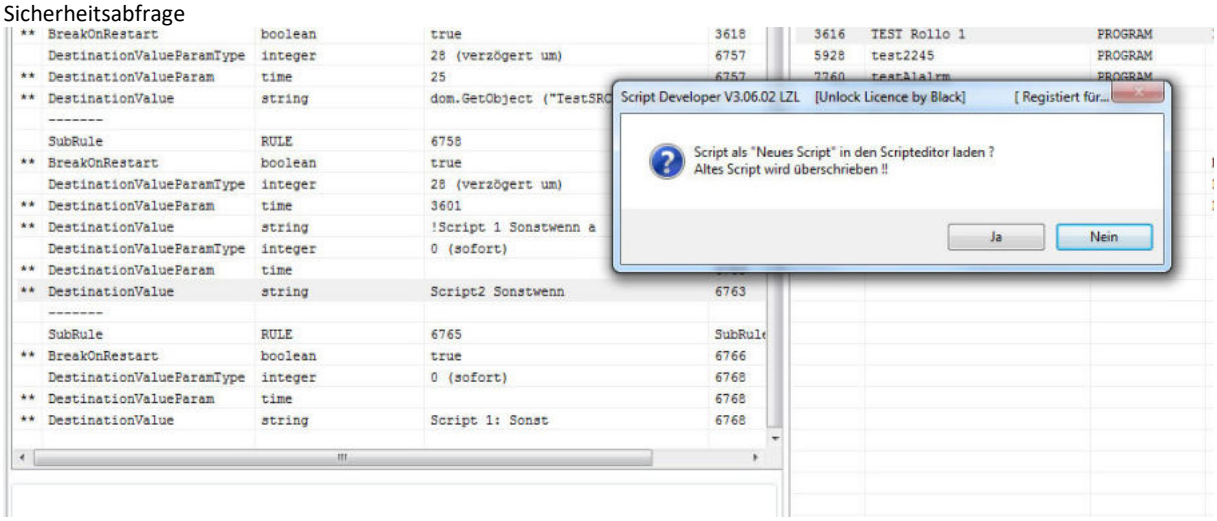


Rule entspricht hier der DANN Aktivität, dort wurde auch das Skript gefunden, welches mit dom.GetObject („TESTSCR……“ beginnt

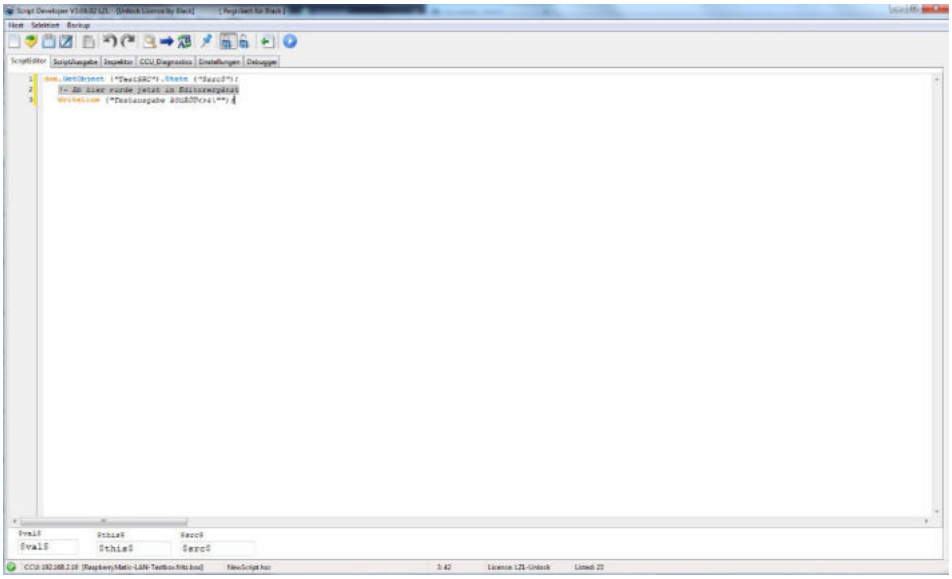
Die erste Subrule enthält dann die beiden Skripte aus der SonstWenn Bedingung.

Die Letzte Subrule entspricht der Sonst Bedingung. Nach dieser Auflistung lassen sich die Skripte tabellatisch im SDV wiederfinden. Zur Hilfe wird in der Spalte Value die ersten 70 Zeichen des Skriptes darsgestellt.

Klick auf den Doppelstern (Doppelstern= Element wurde rekursiv aus dem Hauptelement aufgelöst) führt nach einer Sicherheitsabfrage



Zum Laden des Skriptes in den Skripteditor. Dort können dann die Änderungen durchgeführt oder Tests gemacht werden.



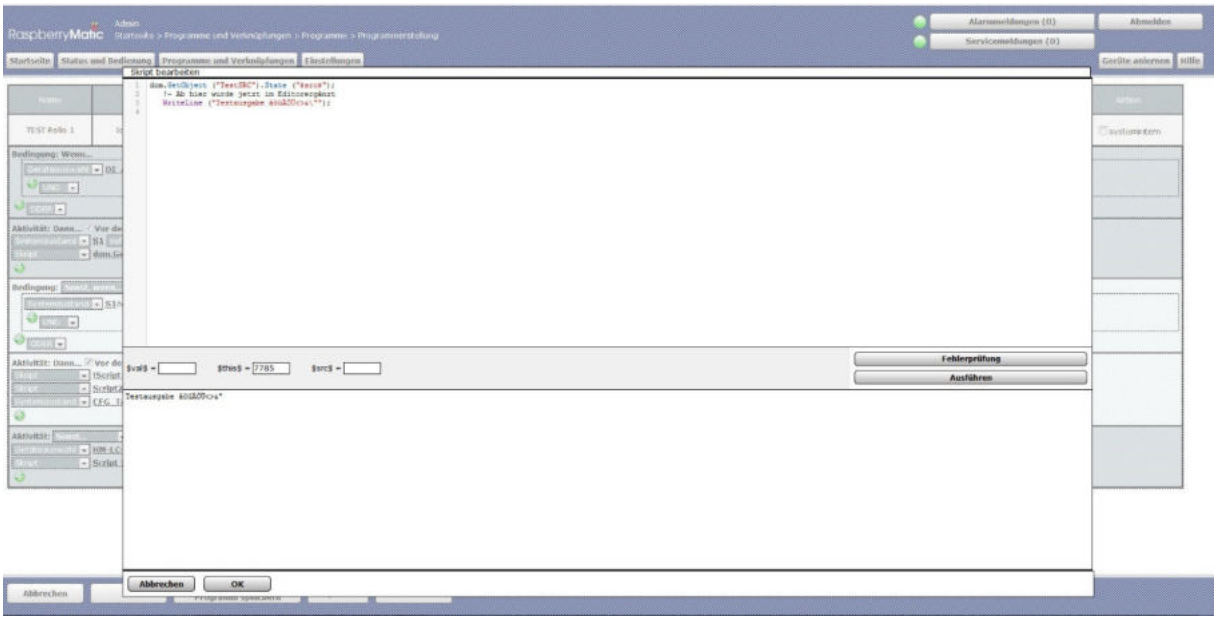
Danach kann durch Drücken der nun nicht mehr grauen Taste

Das Programm aus dem Editor direkt in das Programm der CCU geladen werden. Im Hint wird als Gedächtnisstütze der Name des Programmes mit angezeigt.

Neu ab 3.10.01A: Der SingleDestinationbezug wird gemerkt. Es kann also der Inspektor ganz normal weiter benutz werden. Hostwechsel, neues Skript oder Skript laden löscht diesen gemerkten Bezug. Aus Sicherheitsgründen wird vor dem Rücksichern in das rega Objekt nochmal ein Konsistentest gemacht, ob es sich wirklich um das Quellobjekt handelt. Wenn nicht erfolgt ein Hinweis:

Der Editortext befindet sich dann in der Windows Zwischenablage und muss manuell über die WebUi in das Programm geladen werden, oder aber das Skript muss nochmal aus dem Inspektor extrahiert werden und durch den Inhalt der Zwischenablage ersetzt werden.

Auf der CCU lässt dich dann das geänderte und wieder upgeloadete Skript öffnen, überprüfen und auch Ausführen.



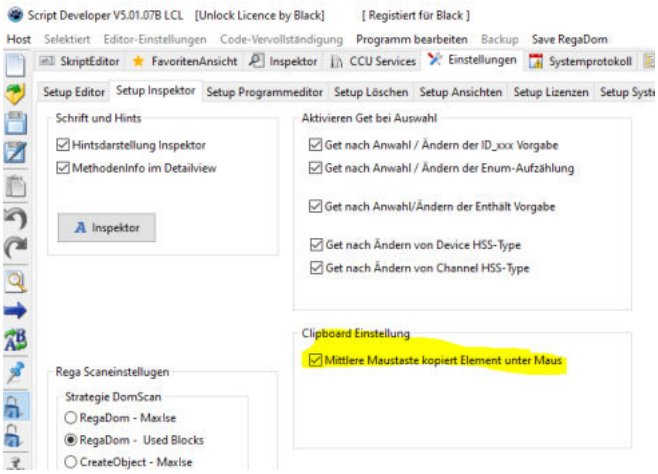
Das Programm würde nun das neue, geänderte Skript bei Triggerung ausführen.

4.4.3 Einzelne Spalten in Zwischenablage kopieren

Einzelne Elemente aus den beiden Listenansichten lassen sich in die Zwischenablage kopieren.

1. Möglichkeit
   Das entsprechende Listen-Control muss aktiv sein. Maus über das zu kopierende Element bringen. Mit Ctrl+Shift+C wird das entsprechende Element in die Zwischenablage kopiert

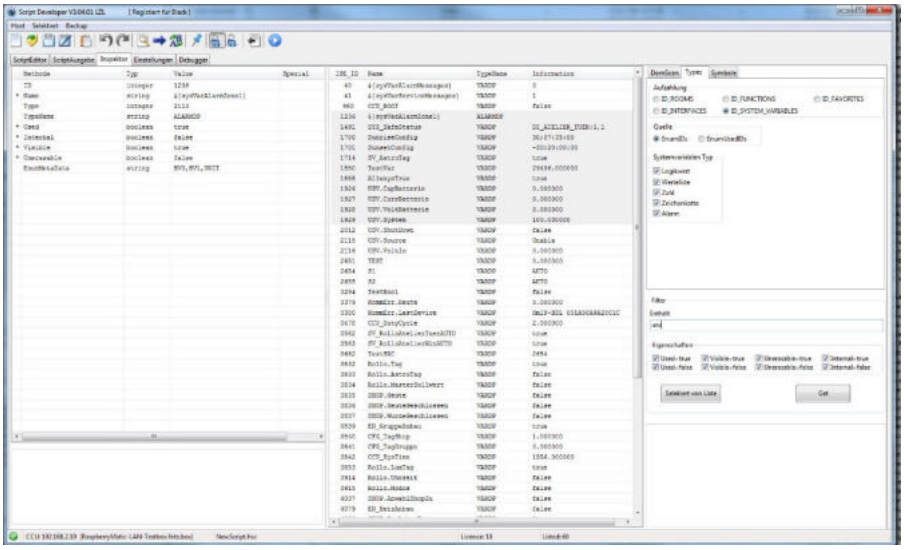
2. Möglichkeit
   Diese Option muss aktiviert sein:
   Dann kopiert die mittlere Maustaste das Element, welches unter dem Mauszeiger liegt, in die Zwischenablage:



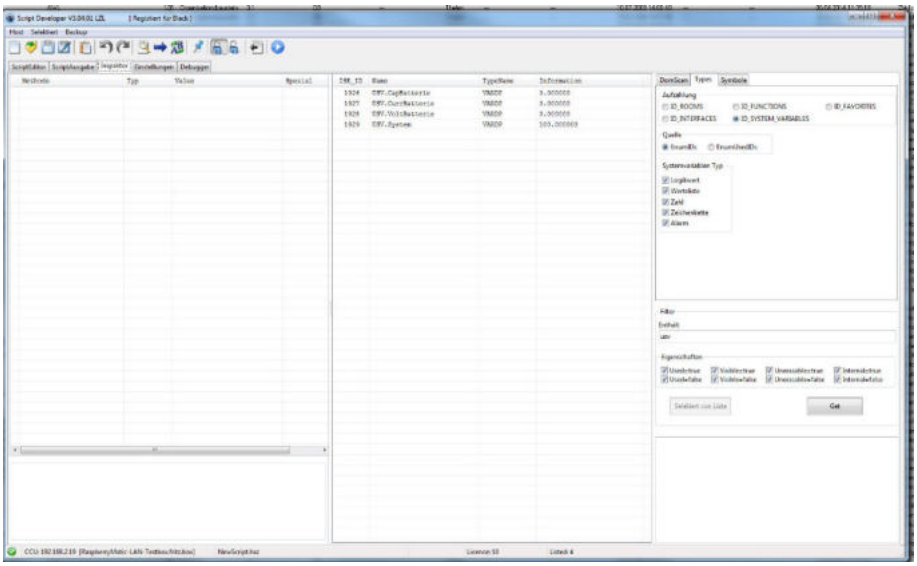
4.5 Selektion von Selektion

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

Hier Beispiel



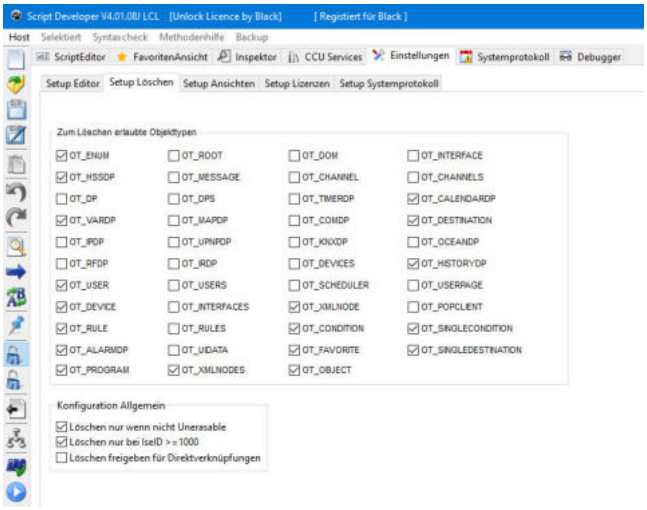
Selektierter Bereich von Systemvariablen, die hier darauf gefiltert werden sollen, dass der Name den String „usv“ enthält. Es müssen 4 Sysvars gefunden werden, die IDS 2012,2115,2116 werden hier nicht berücksichtigt, da diese sind selektiert sind. Bei <Druck auf: Selektiert von Liste: ergibt sich dann



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 unerasbale. (unlöschbar) Ist der Haken gesetzt, geht löschen nur wenn das Objekt nicht auf unerasable = 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)

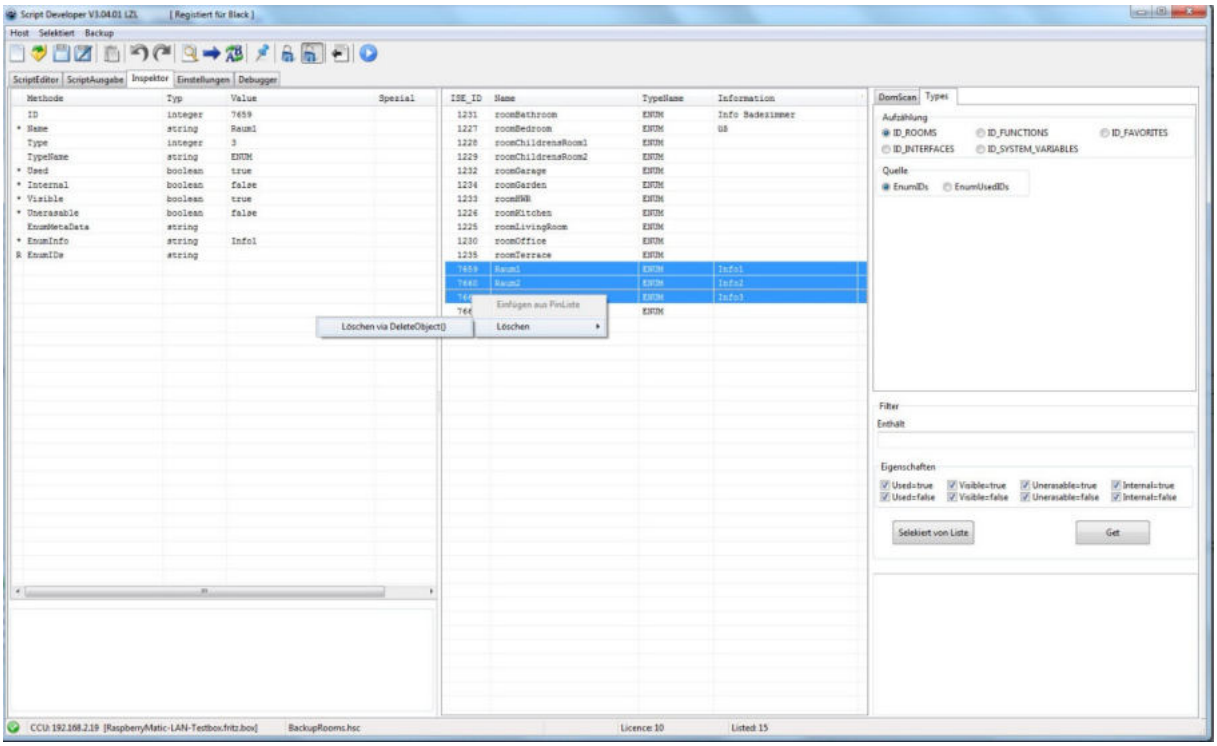
Löschen nur wenn ID>=1000. Dieser haken verhindert, dass man versehentlich Interne IDs der CCU (normalerweise unter kleiner 1000 angelegt) löscht. Will man in dem Bereich löschen, muss der hier explizit manuell unchecked werden.

Die Einstellungen werden NICHT gespeichert, bei jedem Neustarrt 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.

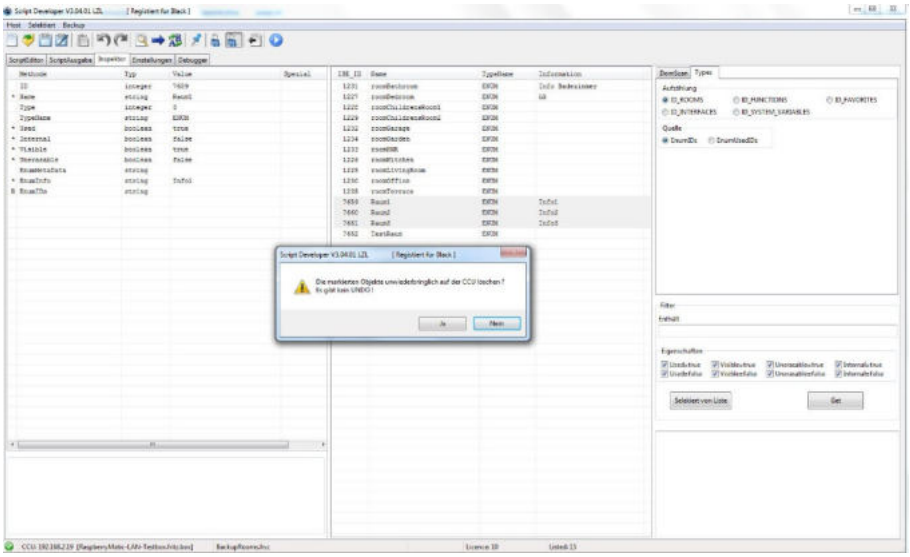
Löschen Freigeben von Direktverknüpfungen: Erlaubt, das Direktverbindungen ebenfalls gelöscht werden können

Löschen läuft so ab:



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

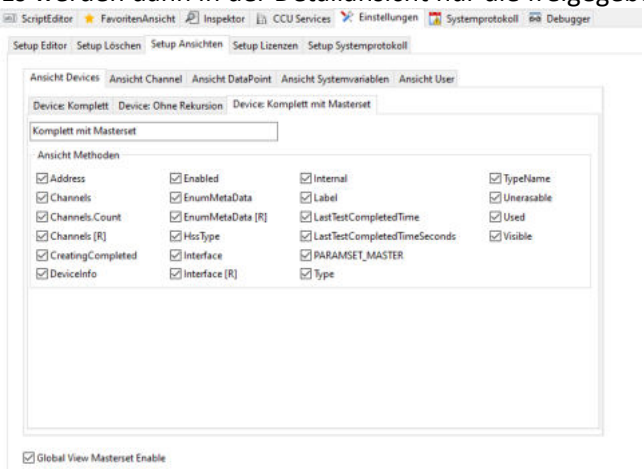
Mehrfachselektion ist möglich



Nach dieser Sicherheitsabfrage sind die Objekte dann weg.. Zurück geht's dann nur mit Restore. Bei einem Device Objekt gibt es noch die zusätzliche Möglichkeit Löschen über xmlrpc.deleteDevices ().

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.



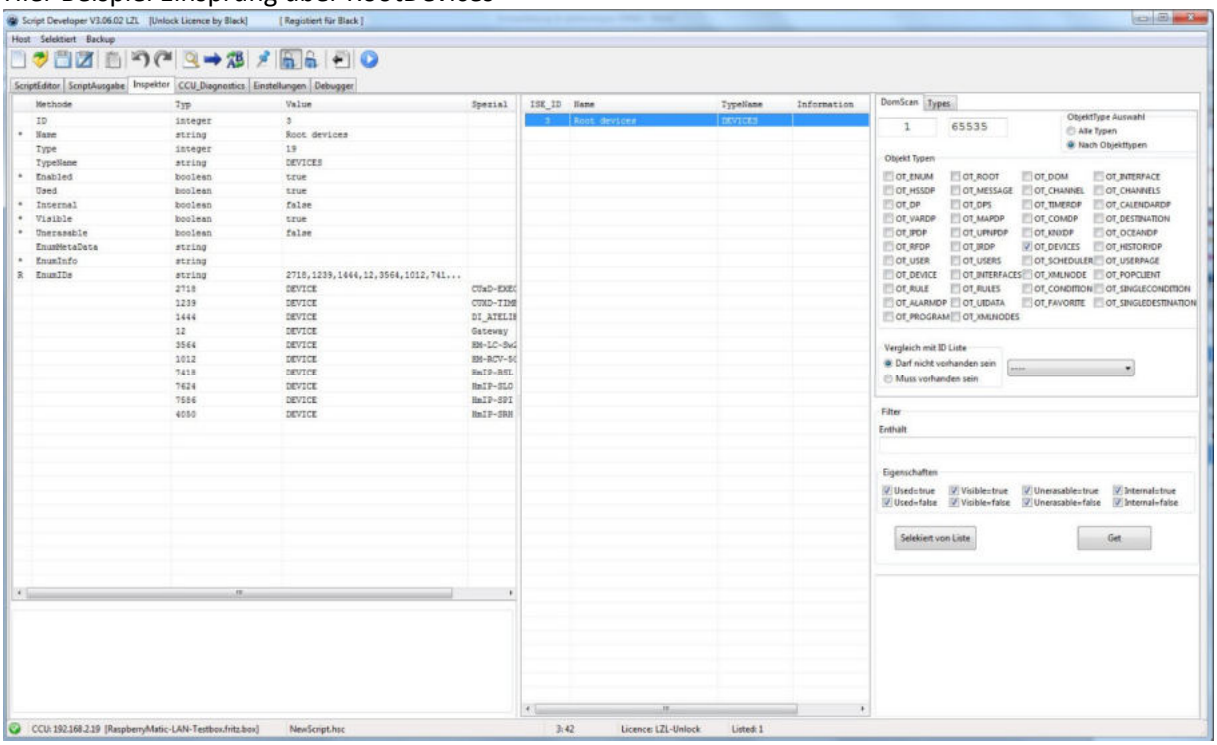
Sichten können mit eigenen Namen versehen werden.

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

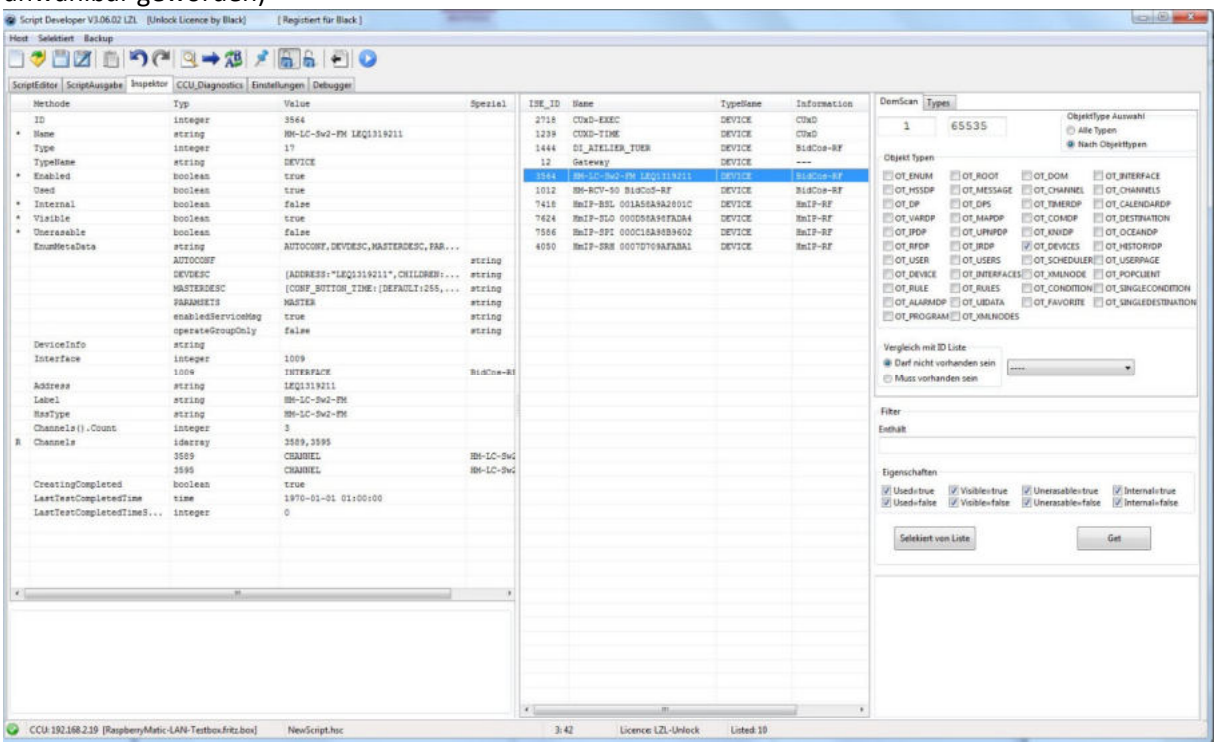
4.8 Browsing durch Rekursionsebenen

Seit der Version 3.06.04 verfügt der SDV über einen UNDO/REDO Stack im Inspektor. Dies bedeutet, dass immer, wenn in eine Rekursionsebene gesprungen wird, sich die Einträge im Selektionsfeld gemerkt werden und man über Undo / Redo dann zwischen den Ebenen hin und herspringen kann. Ein GET löscht dabei immer den Undo Stack

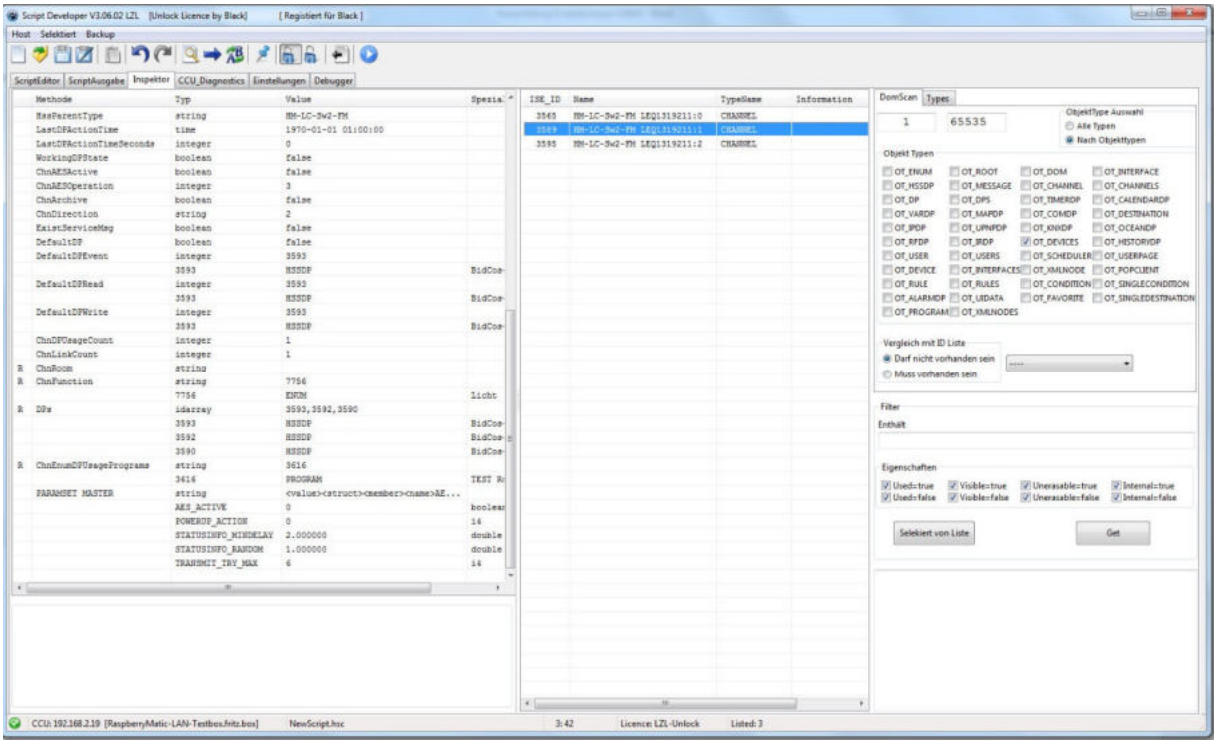
Hier Beispiel Einsprung über RootDevices



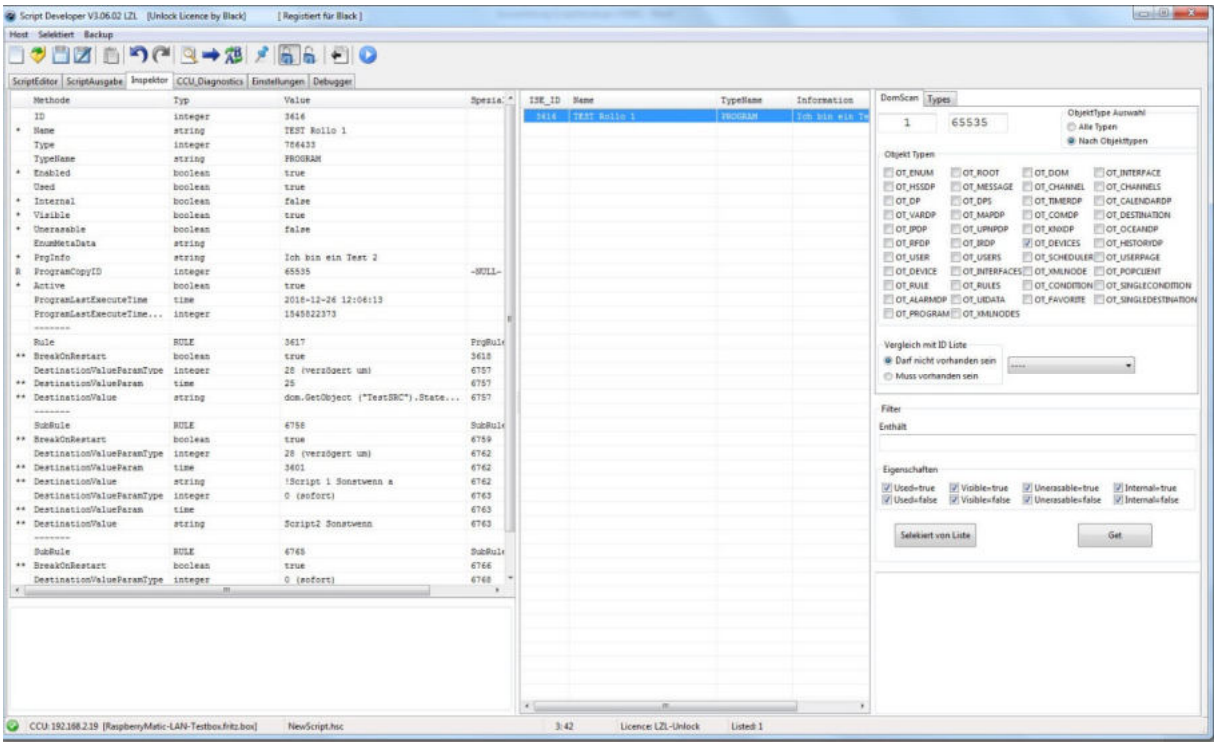
Rekursiv Aufgelöst die EnumIDs und Anwahl des 2 Fach Schaltaktors (Hier zu sehen, Undo ist schon anwählbar geworden)



Und rekursiv weiter über die Channels des Devices

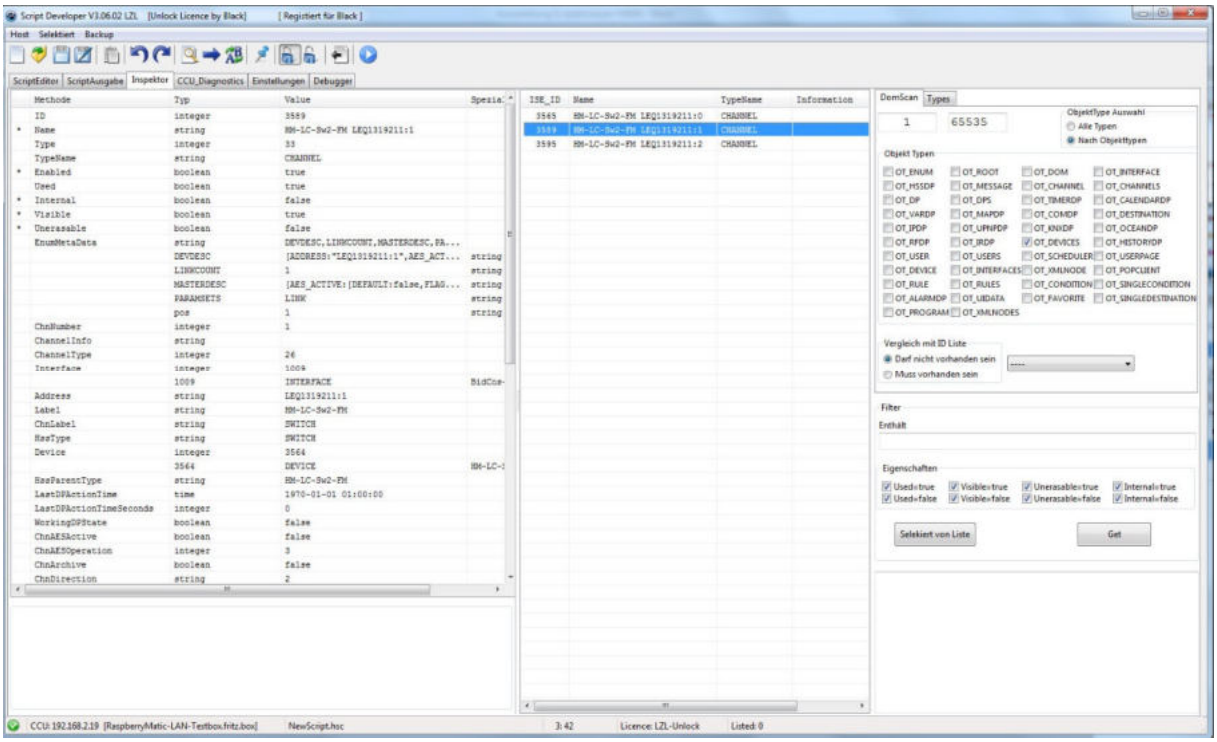


Mal in die Verwendung in dem Programm schauen über ChnEnumDPUsagePrograms



Will ich jetzt aber wieder in den Channel zurück, so war dies in der alten Version nur mit Beginn der Selektion von ganz vorne angesagt.

Ab der 3.06.04 bin ich mit Undo eine Rekursionsebene zurück, hier in der Kanalauswahl, ab der ich direkt weitermachen kann



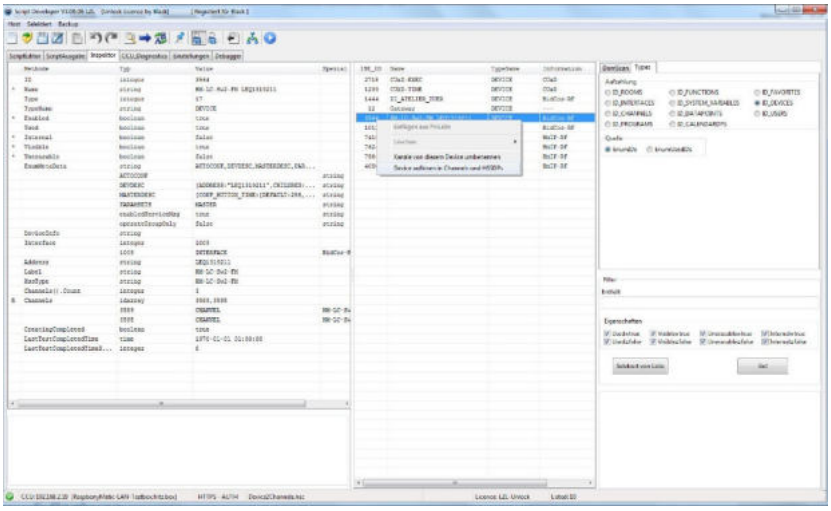
Ab Version 3.06.06 merkt sich der SDV zusätzlich zum Inhalt des Selektionsfeldes auch noch das zuletzt angezeigte Objekt in der Details view und stellt diese Ansicht auch wieder her (So das Objekt noch existent ist)

4.9 Auflösen komplexer Objekte

Der SDV wird die Möglichkeit haben, komplexe zusammen, gesetzte Objekte aufzulösen und zur Bearbeitung zur Verfügung zu stellen.
Beispielsweise Devices und auch Programme.

4.9.1 Auflösen von Devices

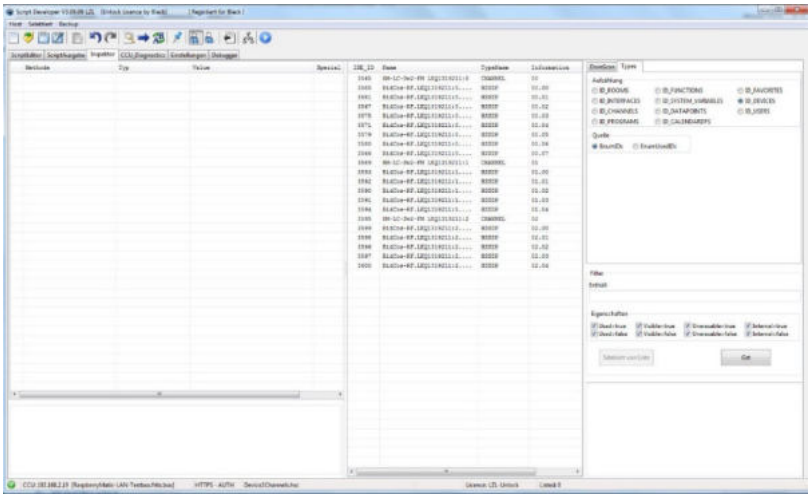
Ein Device besteht ja aus Channels und die Channels wiederrum aus Datenpunkten



Wenn das Selektierte Objekt ein Device ist, so hat das PopUp Menü auf der rechten Maustaste nun auch das Feld: Device auflösen in Channels und HSSDPs.

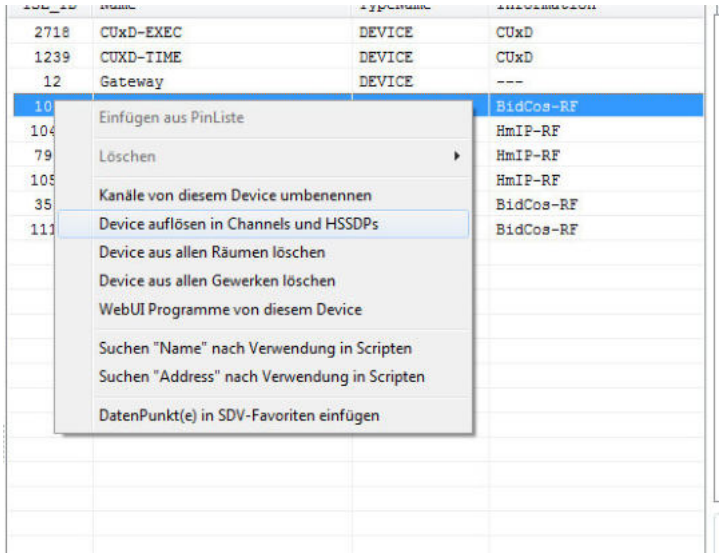
Alternativ über den neuen Menüpunkt  . Dieser ist nicht ausgegraut, wenn sich ein Objekt auflösen lässt.

Als Ergebnis erhält man :



Diese Liste lässt sich dann weiter untersuchen mit den schon beschriebenen Arbeitsweisen (Auch Undo /Redo)

Wenn ein Device selektiert wurde, stehen mit Klick rechte Maustaste einige Bearbeitungsmenüs zur Verfügung

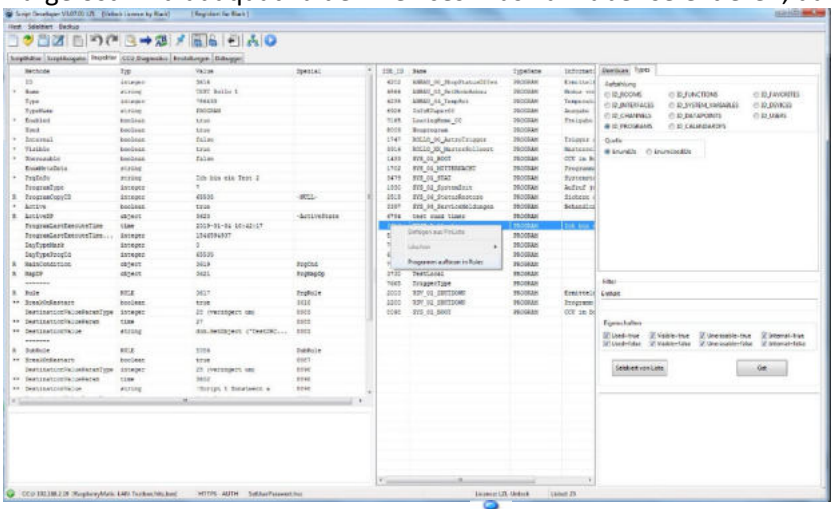


Das dargestellte Menü ist abhängig von dem selektierten Objekttyp.
Bei Devices z.b.
Lassen sich alle untergeordneten Kanäle automatisch aus allen Räumen oder aus allen Gewerken entfernen. Dies war früher immer gerne eine Tipporgie.

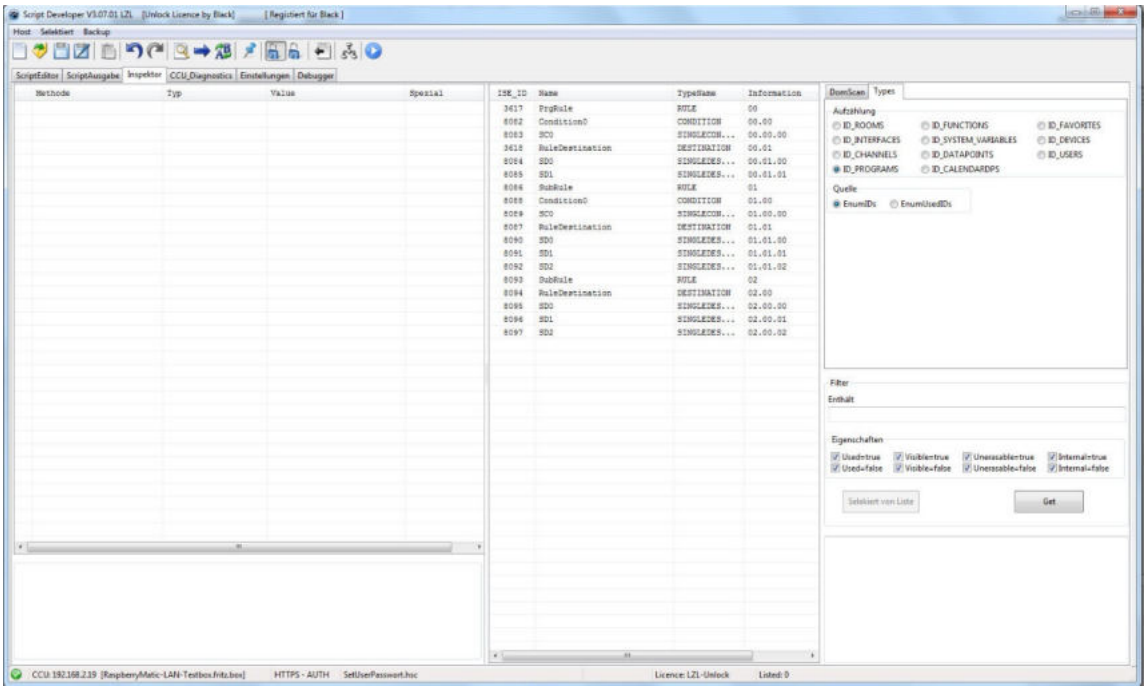
4.9.2 Auflösen von Programmen

Ein Programm besteht aus Rules (Regeln bzw den Subrules) und die jeweils aus den Conditions (und ihren untergeordneten SingleConditions sowie den Destinations und den untergeordneten Single Destinations

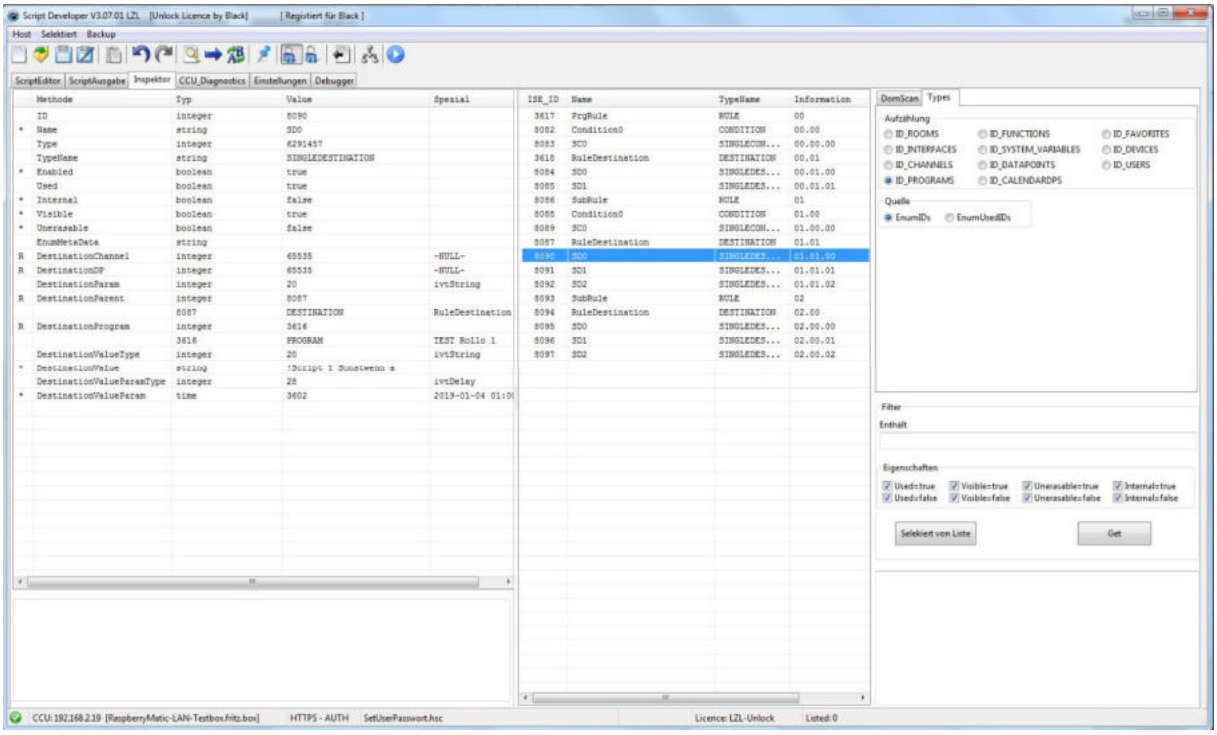
Aufgelöst wird adäquat zu den Devices: Auswahl über selektieren, dann rechte Maustaste und



Alternativ über den neuen Menüpunkt  . Dieser ist nicht ausgegraut, wenn sich ein Objekt auflösen lässt.

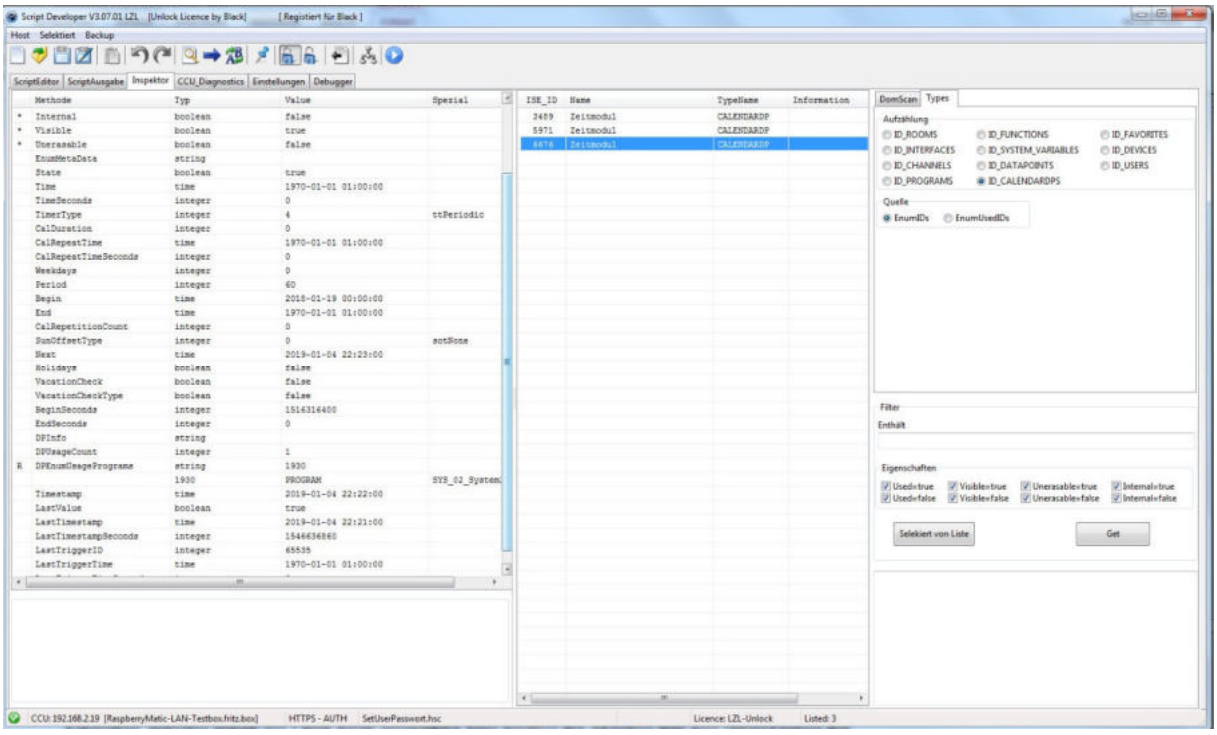


Die Sortierung über Information erlaubt eine chronologische Sortierung nach Auftreten der Objekte in einem Programm. Diese Liste lässt sich nun in der Detailansicht weiter untersuchen



4.10 Zeitmodule

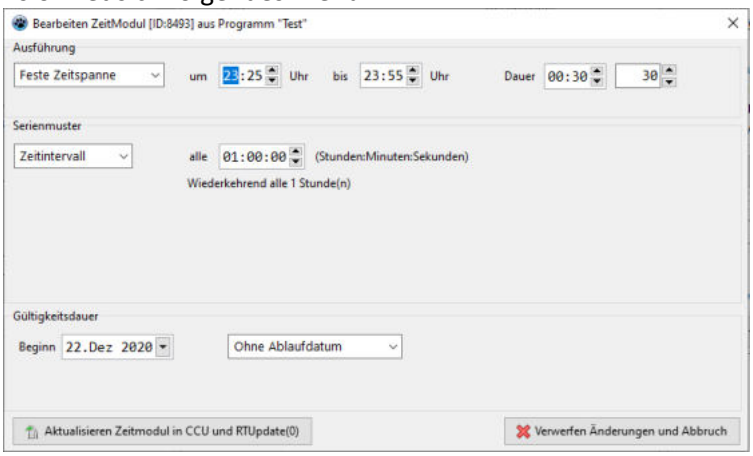
Zeitmodule lassen sich nun auch in Detailansicht darstellen



4.10.1 Ändern eines Zeitmodules

Ein Zeitmodul kann komfortabel geändert werden. Dazu entweder in der Listenansicht doppelklicken auf das Objekt, oder in der Detailansicht, wenn das Objekt ein Zeitmodul ist.

Es öffnet sich folgendes Menü:



Hier kann Intuitiv die Werte des Zeitmodules geändert werden. Aktualisieren ändert das CalendarDP direkt und führt ebenso ein dom.RTUpdate(0) durch, damit die Rega diese Änderungen übernimmt. Verwerfen beendet das Menü ohne Datenübernahme.

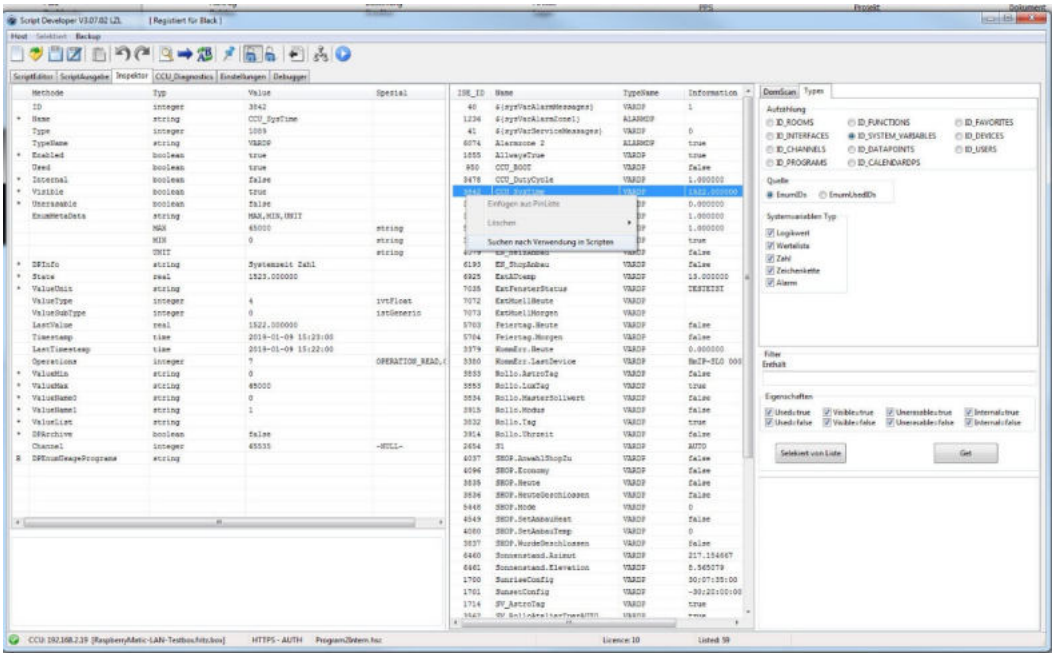
4.11 Suchen in Skripten nach Variablen, Devices etc

Ab der Version 3.07.02 kann in den vorhandenen Skripten nach dem Vorkommen von Systemvariablen, Geräten, Kanälen, Räumen und Gewerken gesucht werden.
Dazu werden sämtliche SingleConditions, die als Property ein Script enthalten, gesucht und dann via Stringvergleich nach dem oder den Vorkommenden Namen abgesucht.

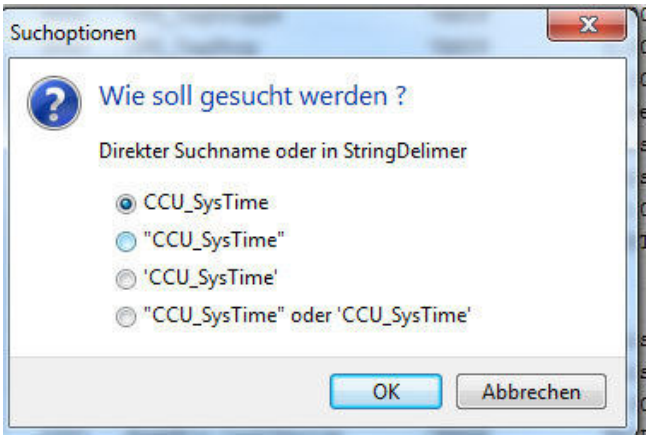
Vorgehensweise:

Markieren der oder auch mehrere Suchvariablen (Können Systemvariablen, Devices, Kanäle, Räume und Gewerke sein) Rechte Maustaste und suchen nach Verwendung in Skripten
Es kann hierbei gesucht werden:

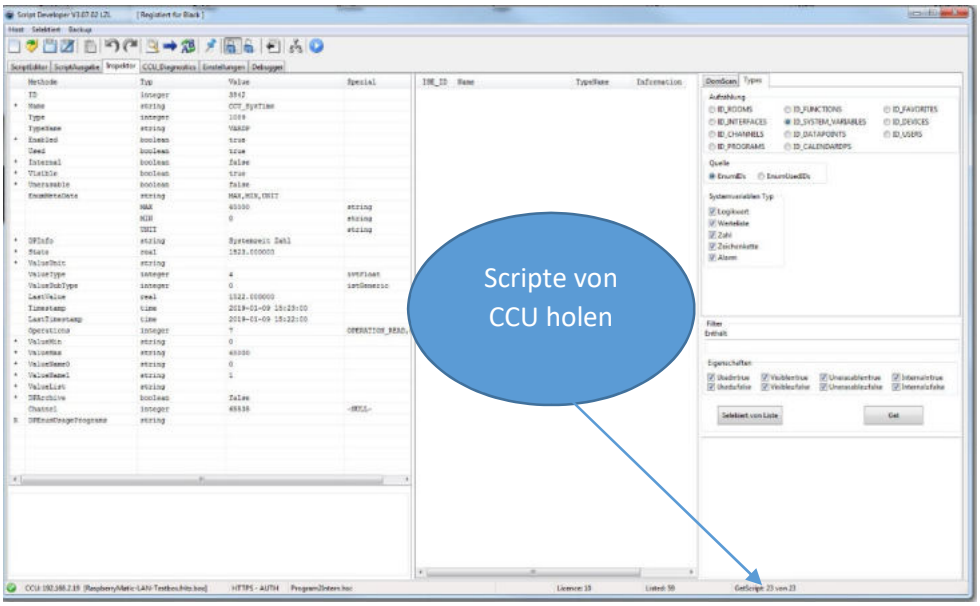
- Suchen "Name" nach Verwendung in Skripten : Suche nach dem Namen (SV, Alarme, Device, Chans)
- Suchen "Address" nach Verwendung in Skripten : Sucht nach der Seriennummer (Address) von Devices und Channels



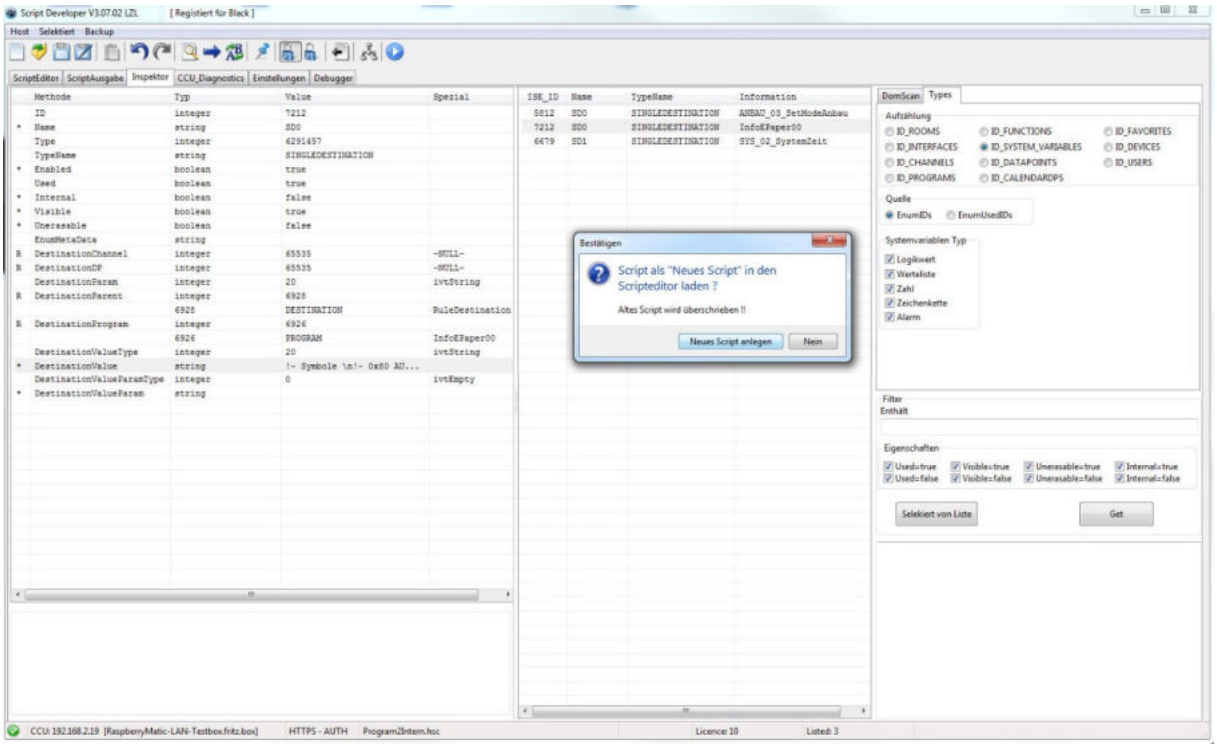
Im darauf sich öffnenden Dialog festlegen ob der reine Text oder der Text in StringDelimitern gesucht werden soll



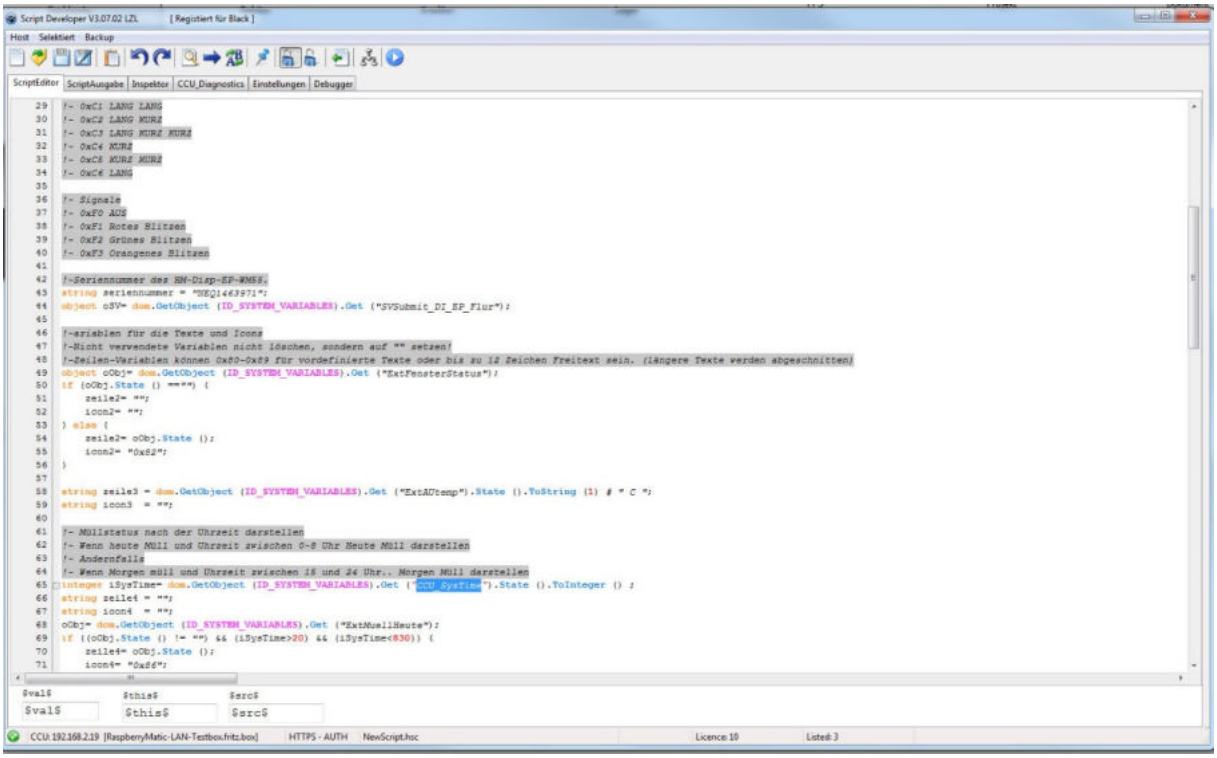
Daraufhin werden erstmal alle Rule in ihre Destinations aufgedröselt und eine Liste angelegt, wie viele Skripte es in diesen SingleDestinations dann gibt. Diese werden in den PC geladen. Das geht recht Fix, um der CCU zwischenzeitlich Zeit für Ihre Aufgaben zu lassen, wird direkt nach dem Empfang PC seitig die Stringanalyse gemacht und die Ergebnisliste aufbereitet.



Wenn fertig, gibt es eine Liste der Singledestinations, die die Skripte enthalten, in welchem der gesuchte Name vorhanden ist. Im Informationsfeld wird auch noch der zu der SingleDestination gehörendem Programm angezeigt:
Das Skript lässt sich dann nach Rückfrage öffnen:

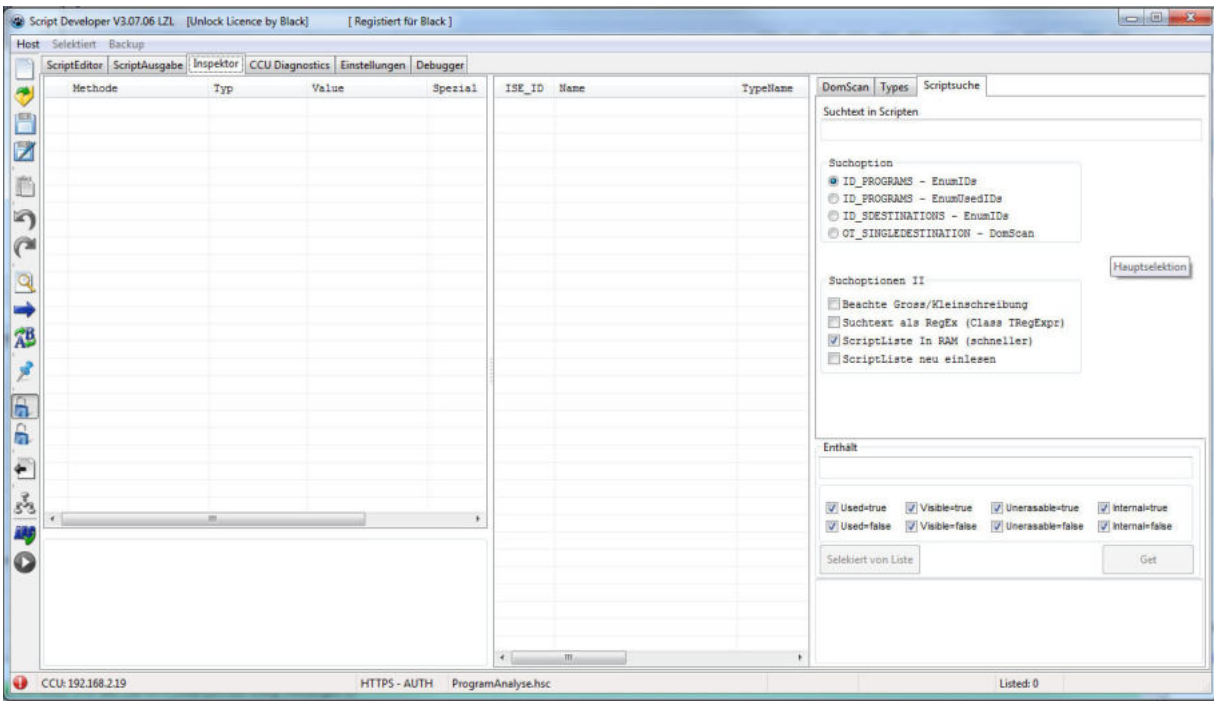


Der Editor öffnet sich und es wird direkt zum ersten Vorkommen des zu suchenden Namens gesprungen. bei mehrfachem Vorkommen sind alle Stellen gemäß den MarkUp Einstellungen hervorgehoben. Mit dem Pfeil für Weitersuchen lässt sich dann auch durch der Text nach dem Namen durchsuchen unabhängig von der Hervorhebung. Bei suchen Ersetzen ist als Suchbegriff der zu suchende Name schon vorbesetzt. Ersetzen automatisiert ist nicht vorgesehen, da sollte als letzte Instanz der Mensch das letzte Wort haben. Ein geändertes Skript lässt sich auch direkt wieder auf die CCU in das ursprüngliche Programm hochladen



4.12 Volltextsuche in Skripten

Ab Version 03.07.07 existiert eine Volltextsuche (ab Level 6)



Suchtext ist selbsterklärend.
Entweder wird nach dem Vorkommen des Textes in dem Skript gesucht oder aber, wenn der Haken bei Suchtext als Regex gesetzt wurde, wird dieser Suchtext als regular Expression ausgewertet. (Da hier die Class TRegExpr von Lazarus verwendet wurde… das ganze in POSIX Syntax)

ID_Programs – EnumIDs
Es wird in der Aufzählung ID_PROGRAMS , alle dort gelisteten IDs iteriert und in allen Skripten gesucht, die dort enthalten sind.

ID_Programs – EnumUsedIDs
Es wird in der Aufzählung ID_PROGRAMS , alle dort gelisteten EnumUsedIDs iteriert und in allen Skripten gesucht, die dort enthalten sind.

ID_SDESTINATIONS – EnumIDs
Es wird in der Aufzählung ID_SDESTINATIONS , alle dort gelisteten IDs iteriert und in allen Skripten gesucht, die dort enthalten sind.

ID_SINGLEDESTINATION – DomScan
Es wird die gesamte Regadom nach Objecten vom Typ Singledestination durchsucht und in allen Skripten gesucht, die dort enthalten sind. Diese Suchmethode dauert am längsten, findet aber auch Geisterobjekte.

Beachte Gross/Kleinschreibung
Selbsterklärend, ist der Haken gesetzt, muss das Wort genauso in Gross Kleinschreibung vorhanden sein, ansonsten ist Gross Kleinschreibung egal

Suchtext als Regex
Der Suchtext wird als Regulärer Ausdruck interpretiert. .*Son.t.* findet alle Skripte die das Wort Sonst, aber auch z.B. Sonat enthalten. Syntax nachzulesen unter
https://en.wikipedia.org/wiki/Regular_expression#POSIX%20Basic%20Regular%20Expressions

Skriptliste in Ram
Diese Suchoption ist schneller, hierbei wird nur beim ersten Suchlauf die Skriptliste in den Rechner geladen. Wenn mehrere Suchläufe gemacht werden, wird ab dem zweiten Durchlauf mit den Scripten aus dem Ram gearbeitet. Dies ist wesentlich schneller als jedes Mal die Skripte neu von der CCU zu laden.

Skriptliste neu einlesen
Wurden Skripte geändert oder Skriptänderungen zwischenzeitlich auf der CCU gemacht, so sollte dieser Haken gesetzt werden. Beim nächsten Suchlauf wird die einmalig Skriptliste neu aus der CCU geladen

4.13 Suchen Verwendungsstelle in WEB-UI Programmen (ab 3.09.04)

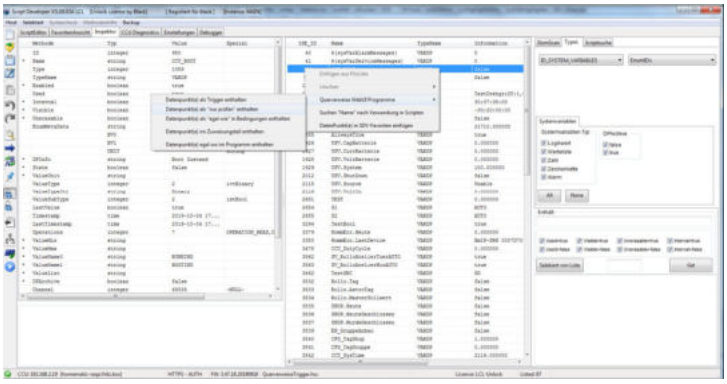
Aus dem Inspektor heraus kann nun komfortable in WebUI Programmen gesucht werden nach Verwendungen:
es kann gesucht werden:

1. über ein Device: Das Device wird über die Channels in seine Datenpunkte aufgelöst und mit dieser Datenpunktliste läuft dann die Programmanalyse. Ein Device führt also zu einem ID-Bündel
2. Über einen Channel: der Channel in seine Datenpunkte aufgelöst und mit dieser Datenpunktliste läuft dann die Programmanalyse. Ein Channel führt also zu einem ID-Bündel
3. HSSDP: dieser einzelne HSSDP wird gesucht
4. Sysvar: diese einzelne Sysvar wird gesucht

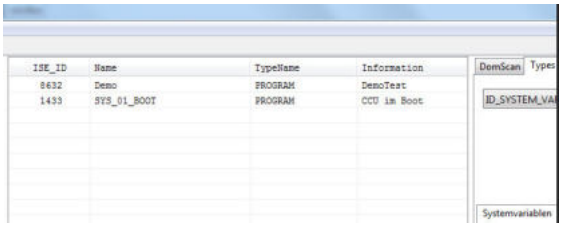
Diese ID liste (entweder eine oder auch mehrere IDs läuft dann durch die Programmanalyse). dabei wird geprüft:
eine dieser IDs ist ein Trigger des Programms -> Ergebnis in Inspektor
eine dieser IDs ist "nur prüfen" des Programms -> Ergebnis in Inspektor
eine dieser IDs ist Im Bedingungsteil des Programms (Trigger oder nur prüfen)-> Ergebnis in Inspektor
eine dieser IDs ist im Zuweisungsteil des Programms - Ergebnis in den Inspektor
Eine dieser IDs kommt im Bedingungs oder Anweisungsteil vor --> Ergebnis in den Inspektor

(Skriptsuche ist davon unabhängig, der SDV konnte auch schon vorher Volltextsuchen in Skripten)

Arbeitsweise:
Ist das selektierte Objekt ein Device, Channel, HSSDP oder Sysvar, gibt es auf der rechten Maustaste den Menüpunkt: Querverweise WebUI Programme.
Dabei kann nun eine der Suchvergänge ausgewählt werden

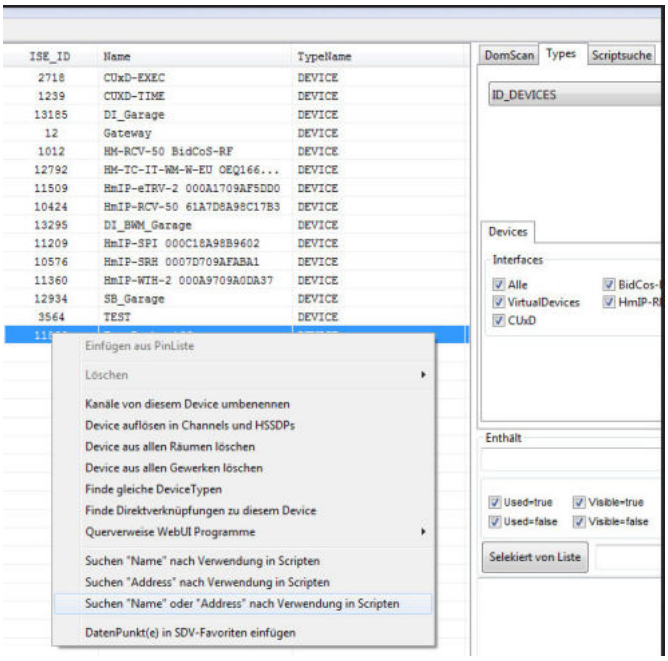


Ergebnis wäre hier die List von Programmen, wo in der WebUI eine Zuweisung auf die Sysvar Anwesenheit gemacht wird



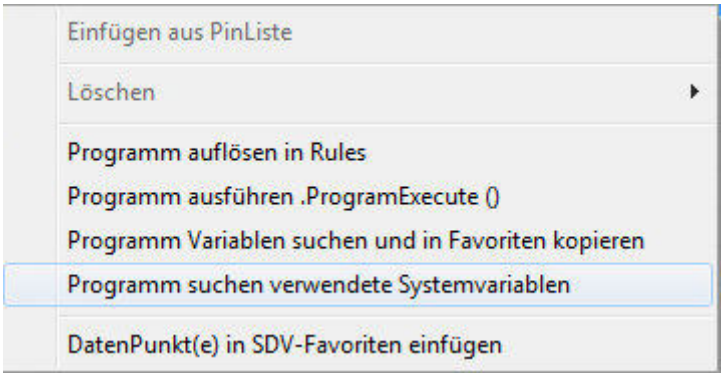
4.14 Suchen „Name“ oder „Adress“ nach Verwendung in Skripten

Das angeklickte Device im Inspektor wird nach vorkommen auf „Name“ bzw der „Adress“ (Seriennummer) in Skripten gesucht.



4.15 Programm suchen verwendete Systemvariablen

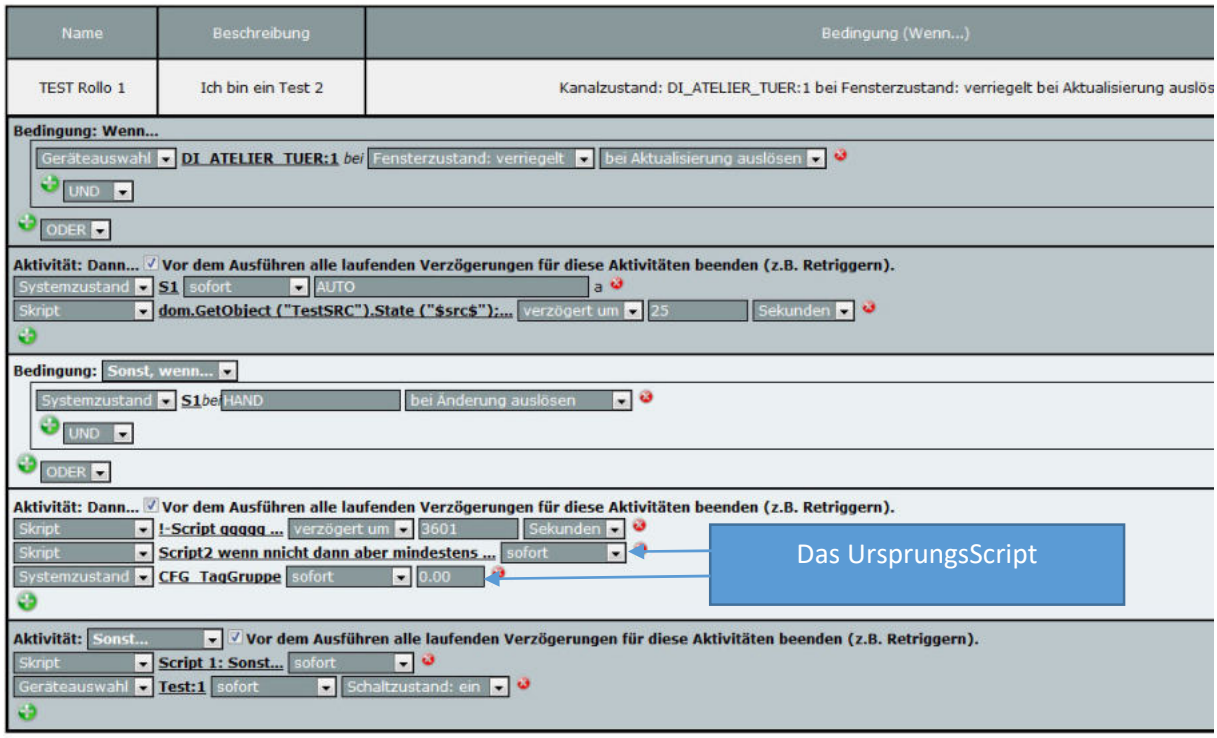
Bei einem selektieren Programm besteht die Möglichkeit, sich die im WebUI Programm verwendeten Systemvariablen anzeigen zu lassen.



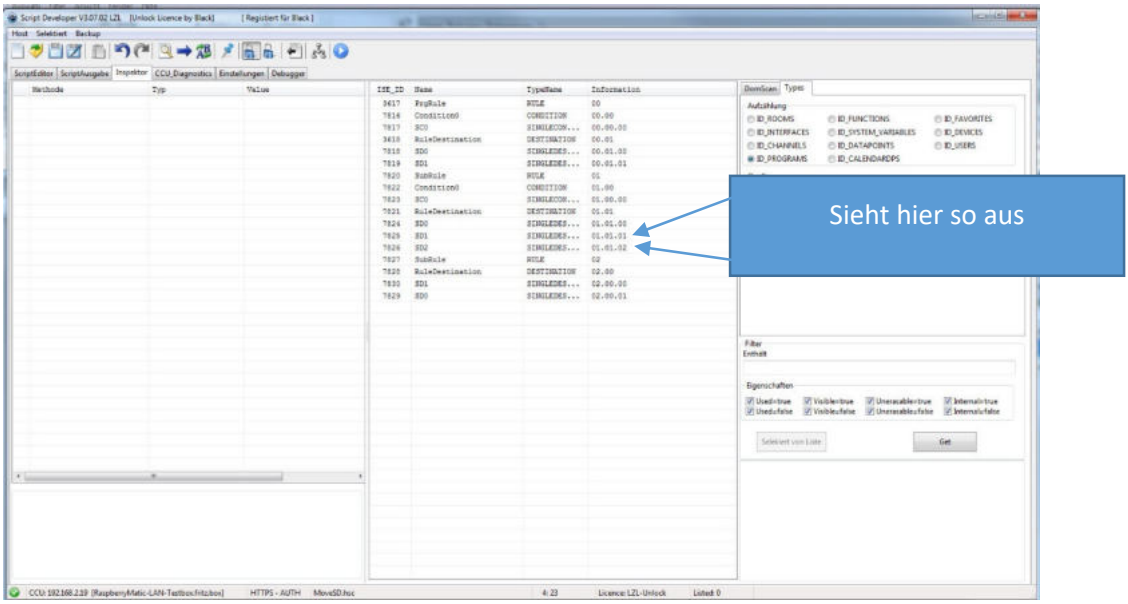
4.16 SingleDestinations in ihrer Reihenfolge ändern

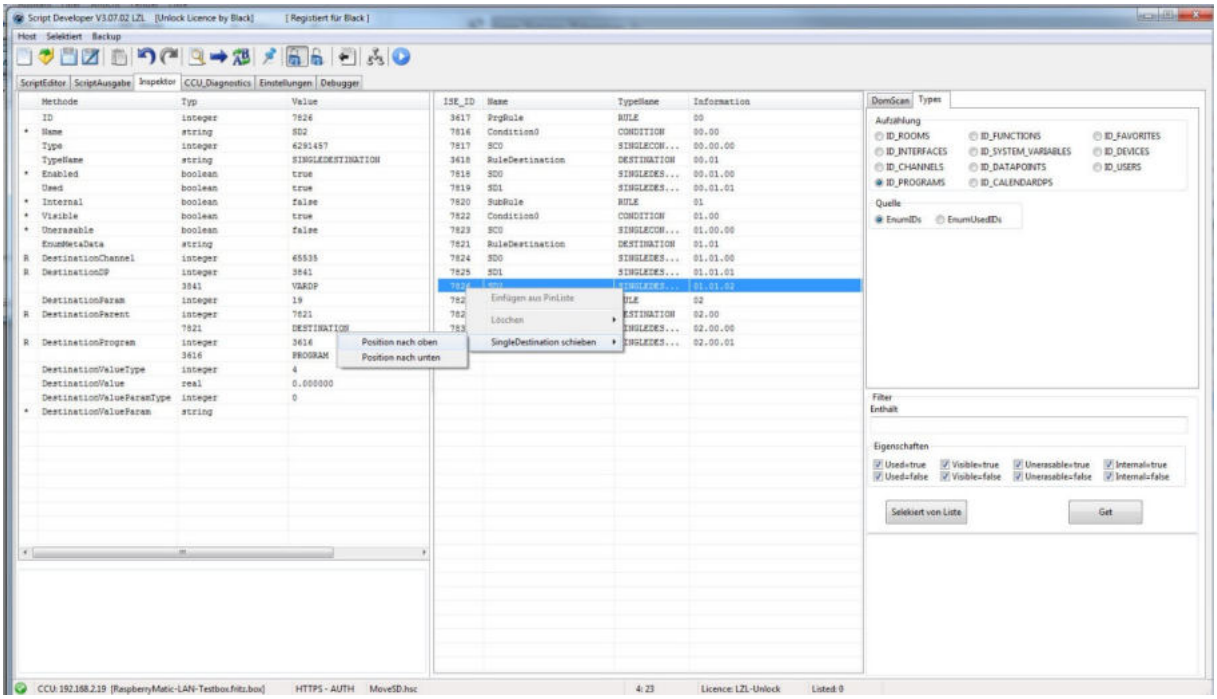
Eigentlich ein Wunsch von einem der Tester. Die Reihenfolge der Anweisungen in einem Programm lässt sich abändern.

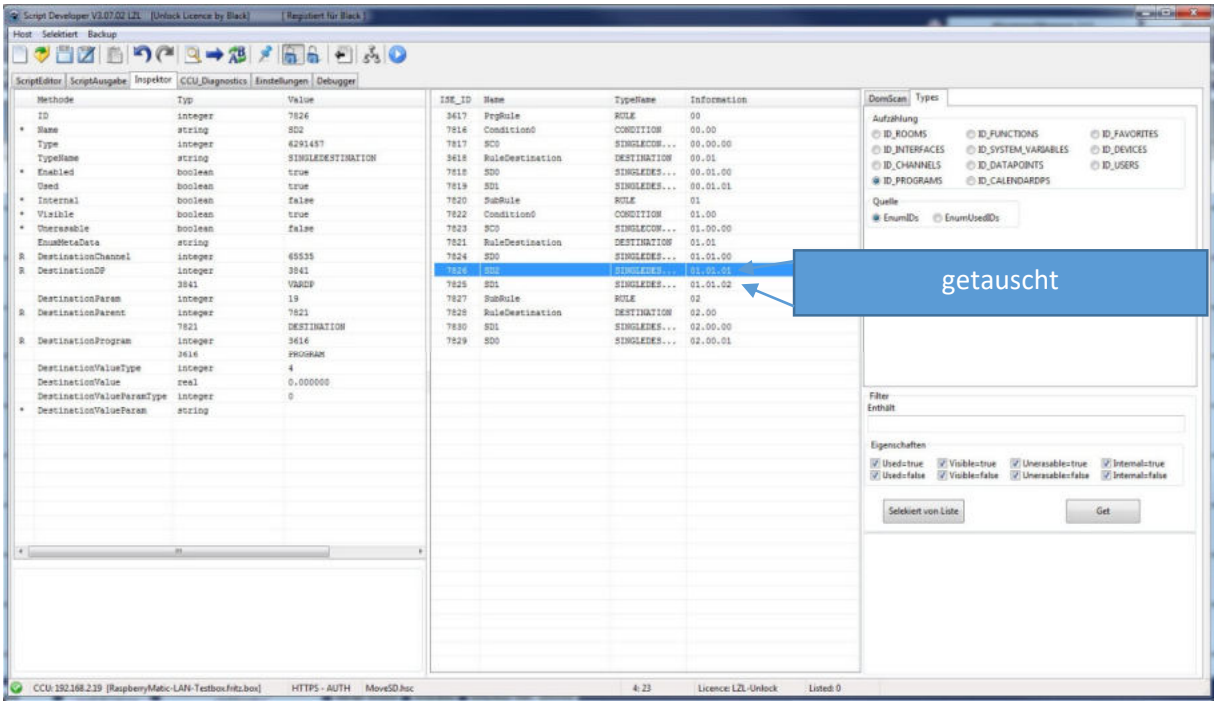
Vorher:

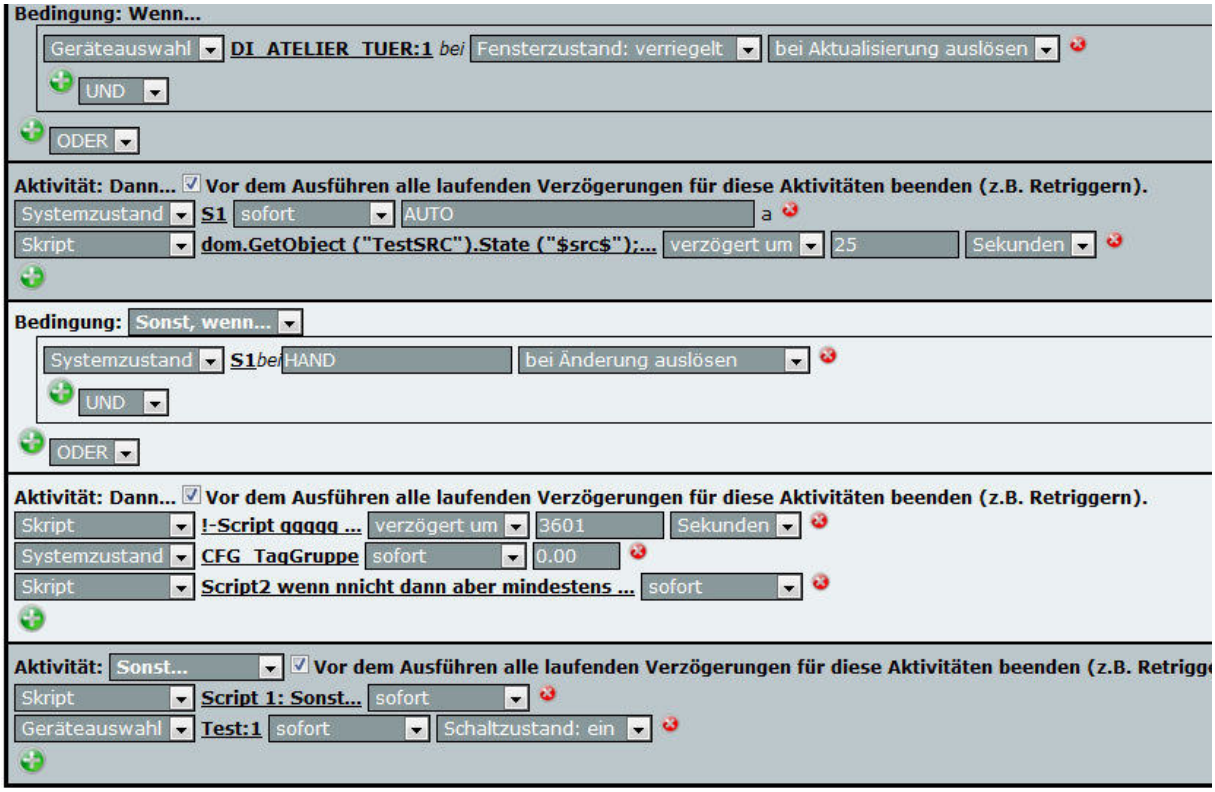


Das Aussehen im SDV





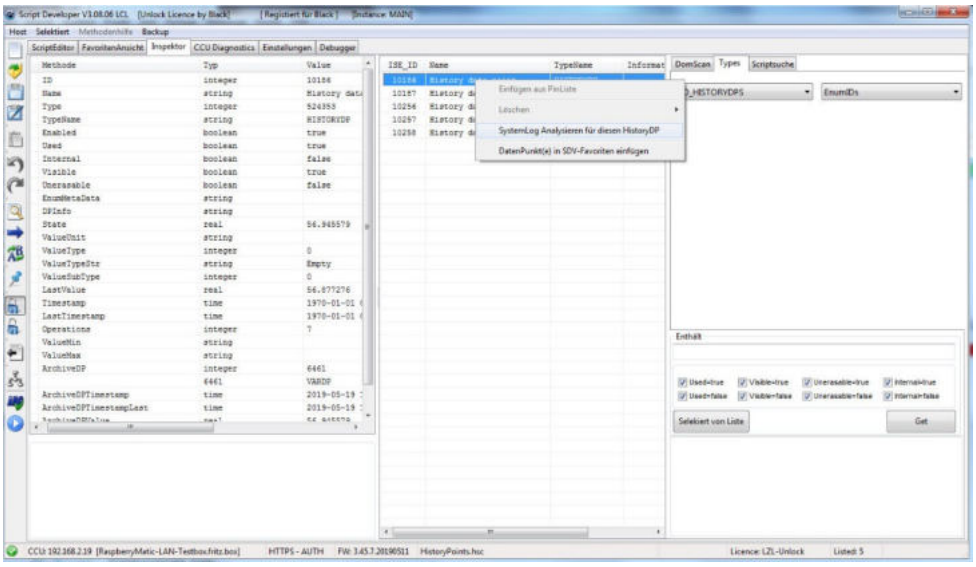

getauscht



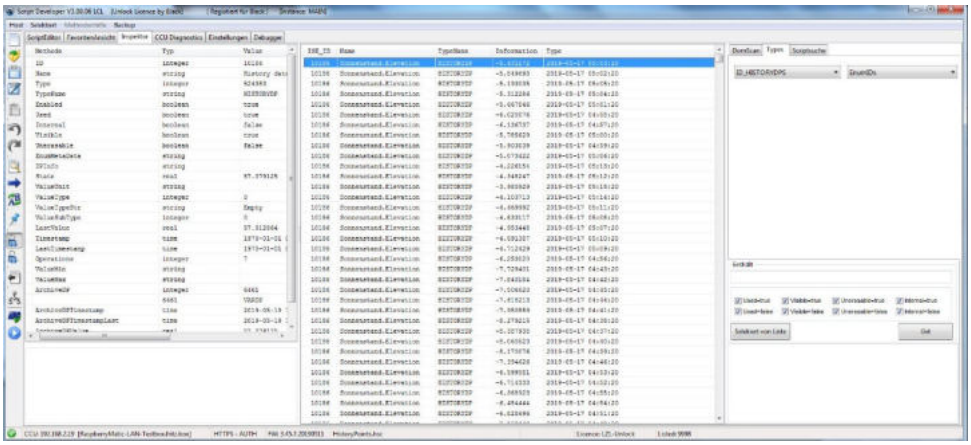
Und hier im WebUI Skript

4.17 History Data Points und Systemprotokoll

Es besteht die Möglichkeit, History Datapoints im Systemprotokoll zu Selektieren



Testweise hat der punkt hier bei mir fast 10000 Einträge, dauert dann ein paar Sekunden, bis die Werte geholt sind

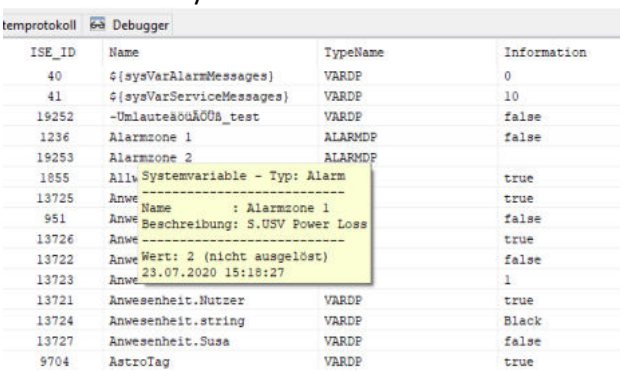


4.18 EchtZeit Hilfe Hints zu Objekten im Inspektor

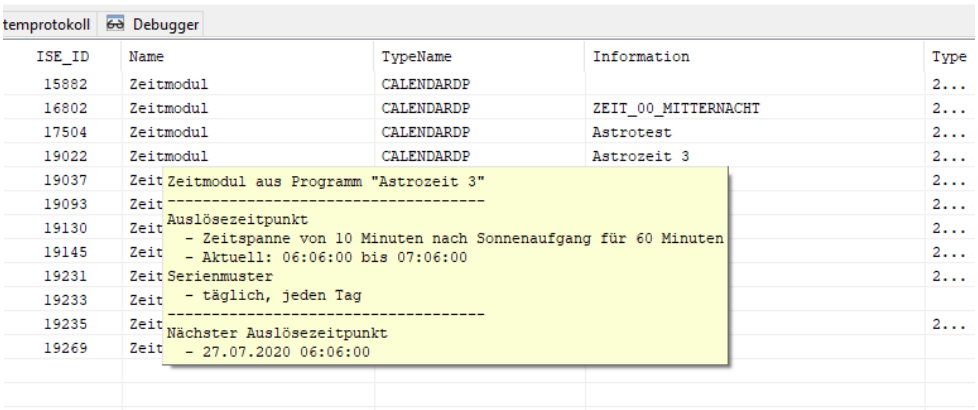
Um effektives Arbeiten und Untersuchen von Objekten zu vereinfachen, verfügt der Inspektor über Hilfehints. Diese werden in Echtzeit Ahängig von jeweiligen Objekt dargestellt:

Beispielsweise in der Listendarstellung:

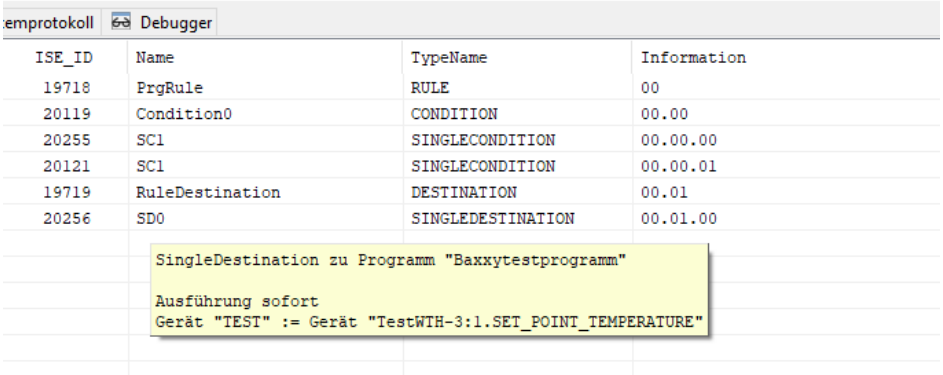
Hilfe bei einer Systemvariablen:



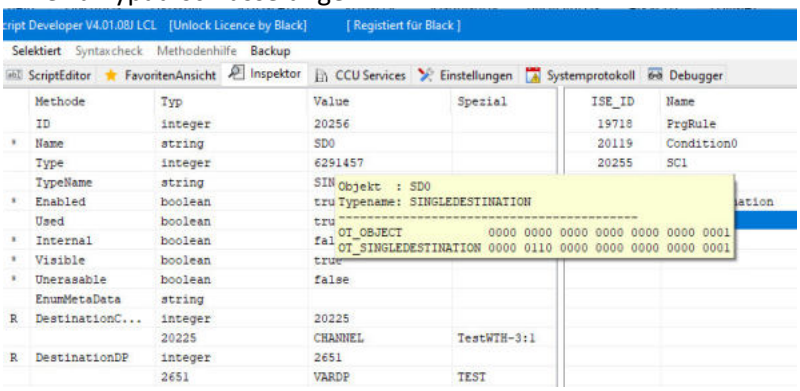
Oder bei einem Zeitmodul:



Schnelle Information zu einer Singledestination
Zuweisung, Ausführung und zugehöriges Programm



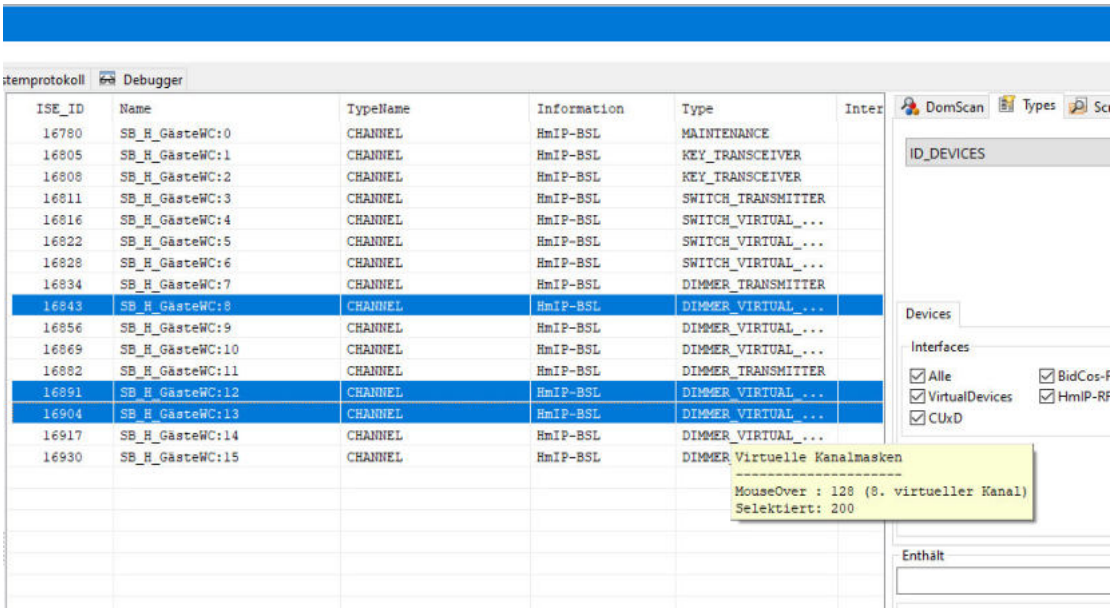
Hilfe zu Typaufschlüsselungen:



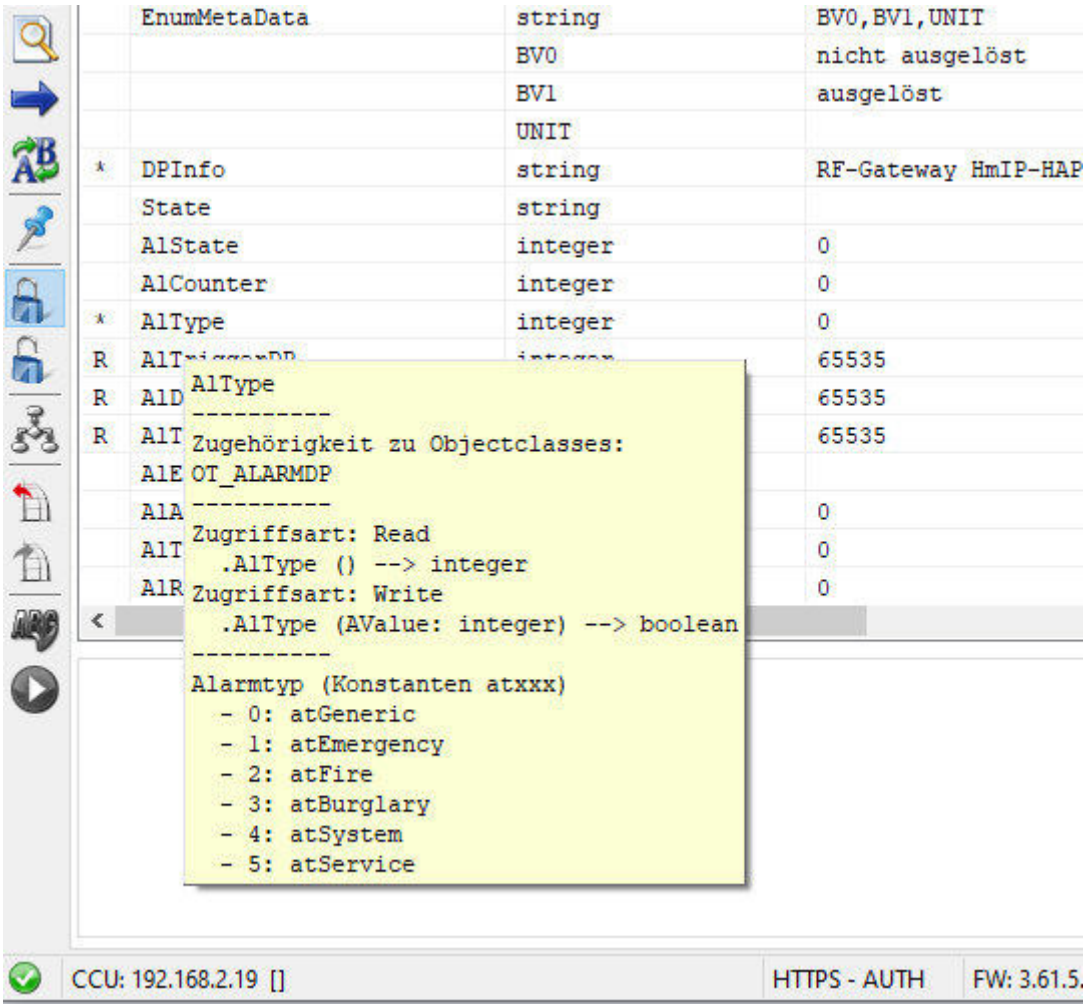
Bestimmung von Binärmasken bei virtuellen Kanälen

Zu bearbeitende Kanäle markieren und dann mit der Maus über einen der Dimmer Virtual gibt einem:

Den Binärwert des Kanales über dem die Mouse steht (Mouseover, hier der 8. virtuelle Kanal)
Bzw die Binärmaske zu den markierten Kanälen (der SDV berücksichtigt hier aber auch nur die virtuellen Kanäle)

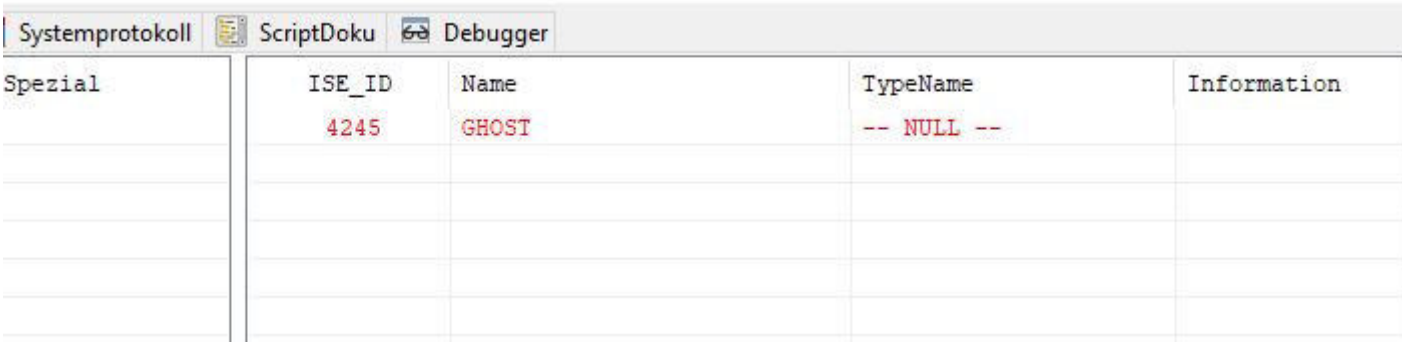


Einstellbar ist auch, ob in der Detailview Hilfetexte zu den Methoden eingeblendet werden sollten



4.19 Ghost Objekte (Bezüge in Aufzählungen auf Objektleichen)

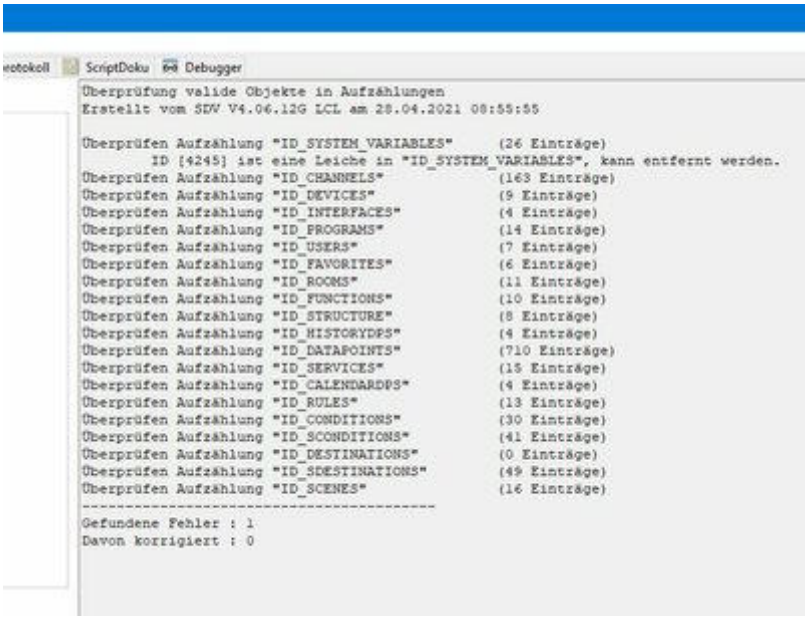
Ghost Objekte wurden im Inspektor früher weggefiltert. Gab dazu ja nix anzuzeigen. erweis sich als verbesserungswürdig. früher ergab die Direkt Eingabe einer ID eines nicht existierenden Objektes ein leeres Listenfeld im Inspektor. Die neue Version stellt ein solches Objekt nun in einer roten Zeile mit der ID, dem Namen Ghost und dem Typ -- NULL -- dar.



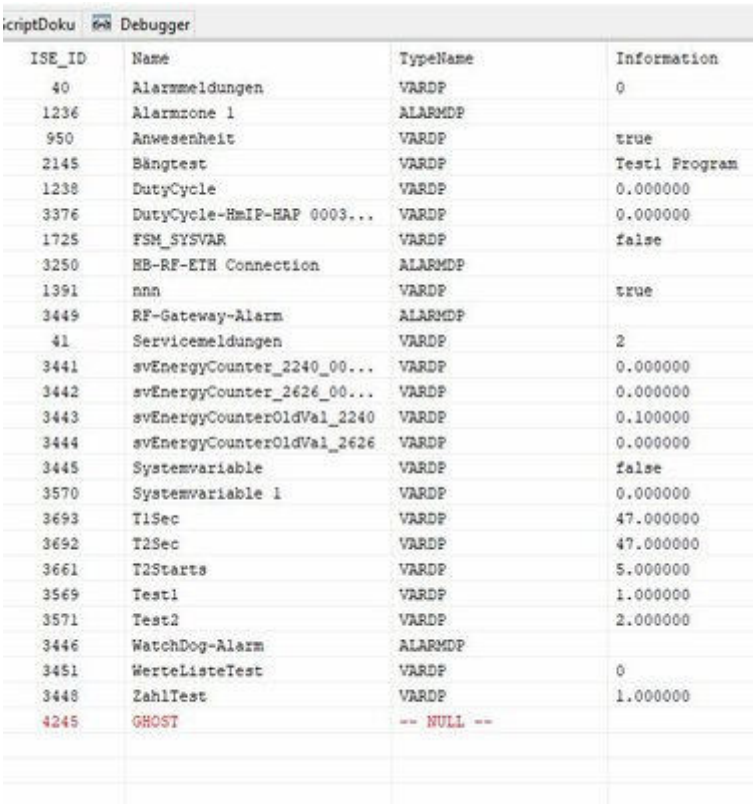
klingt erstmal nach wenig nützlich...
Aber.

wir machen mal absichtlich die Aufzählung ID_SYSTEM_VARIABLES kaputt, in dem wir dort einfach mal einen Ghost einhängen. Also den case, den ich letztens mal bei einem hatte, die Middleware stürzte beim Synchronisieren ab.

gefunden wird dieses beim valid Enum Check

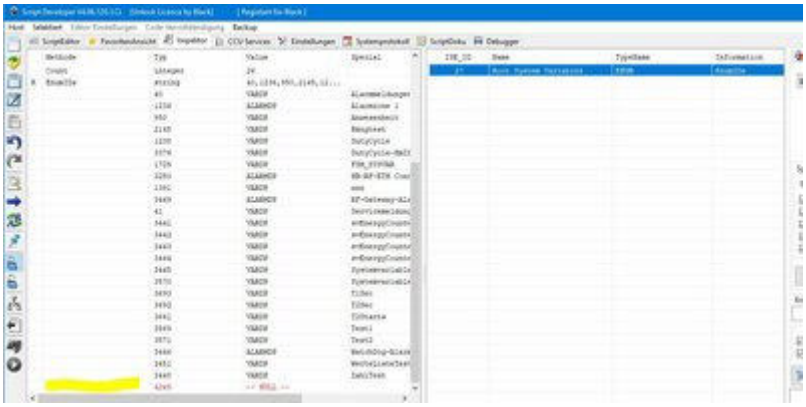


Für erfahrene Nutzer wären nun die weiteren Schritte mit ein paar zusätzlichen tests klar gewesen. Ums nun etwas zu vereinfachen sind die neuen Funktionalitäten nun da. Der Ghosts hängt in ID_SYSTEM_VARIABLES, also erstmal da schauen:

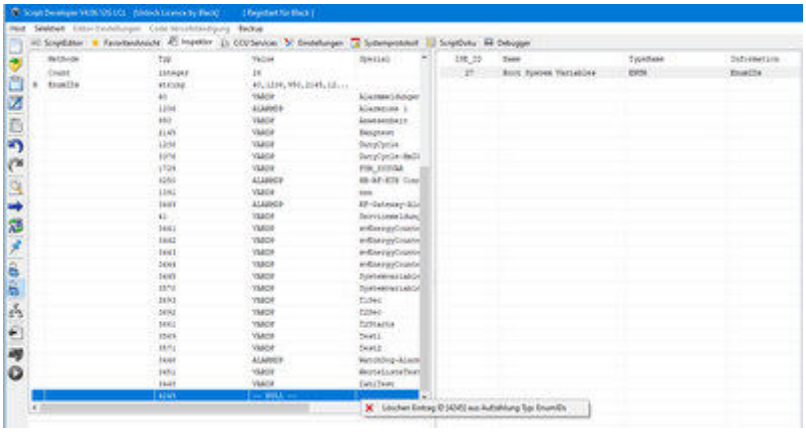


ISE_ID	Name	TypeName	Information
40	Alarmmeldungen	VARDP	0
1236	Alarmzone 1	ALARMDP	
950	Anwesenheit	VARDP	true
2145	Bängtest	VARDP	Test1 Program
1238	DutyCycle	VARDP	0.000000
3376	DutyCycle-HmIP-HAP 0003...	VARDP	0.000000
1725	FSM_SYSVAR	VARDP	false
3250	HB-RF-ETH Connection	ALARMDP	
1391	nnn	VARDP	true
3449	RF-Gateway-Alarm	ALARMDP	
41	Servicemeldungen	VARDP	2
3441	svEnergyCounter_2240_00...	VARDP	0.000000
3442	svEnergyCounter_2626_00...	VARDP	0.000000
3443	svEnergyCounterOldVal_2240	VARDP	0.100000
3444	svEnergyCounterOldVal_2626	VARDP	0.000000
3445	Systemvariable	VARDP	false
3570	Systemvariable 1	VARDP	0.000000
3693	T1Sec	VARDP	47.000000
3692	T2Sec	VARDP	47.000000
3661	T2Starts	VARDP	5.000000
3569	Test1	VARDP	1.000000
3571	Test2	VARDP	2.000000
3446	WatchDog-Alarm	ALARMDP	
3451	WerteListeTest	VARDP	0
3448	ZahlTest	VARDP	1.000000
4245	GHOST	-- NULL --	

die alten Versionen stellten den Ghost nicht dar. hier wie beschrieben: in rot markiert. Nun das Object anklicken, es erfolgt natürlich keine Detailansicht, weil es kein existierendes Objekt ist. rechte Maustaste und suche Referenzbezüge in regadom. Als Ergebnis kommt dann in diesem Fall nur eine Verwendung in ID27, root system variables, heraus, diese nun anklicken und in der Aufzählung erscheint im Detailfeld nun auch der Ghost in rot.



die Kindersicherung rausmachen (Schloss auf) rechte Maustaste auf den Ghost ergibt ein neues Menü



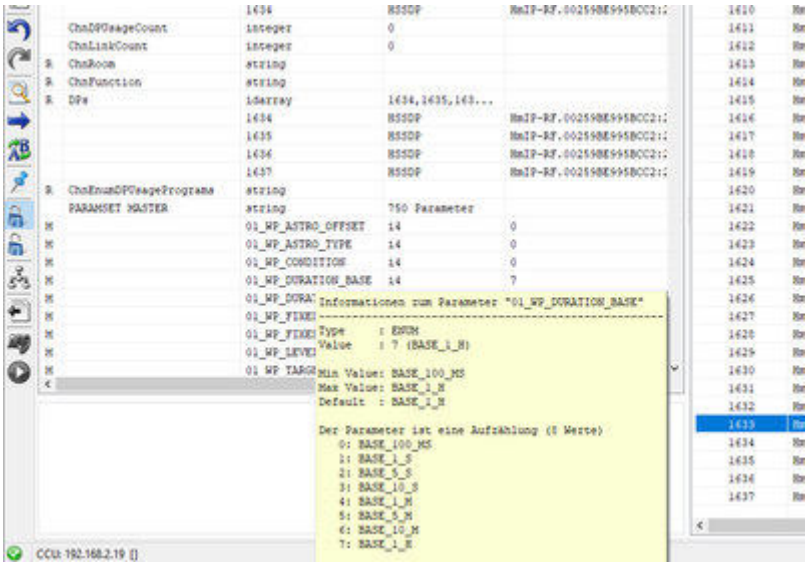
Anhand des Objecttypes des Basisobjectes und der Bezeichnung des IDarrays wählt der SDV die zum entfernen nötige Löschmethode aus und eliminiert das angewählte Object aus der Aufzählung. (nach Bestätigung !)

Mit dieser Vorgehensweise sollten sich nun auch für nicht ganz erfahrene Nutzer Schrott Einträge in Aufzählungen finden, prüfen und auch eliminieren lassen.

4.20 Master und Linksets

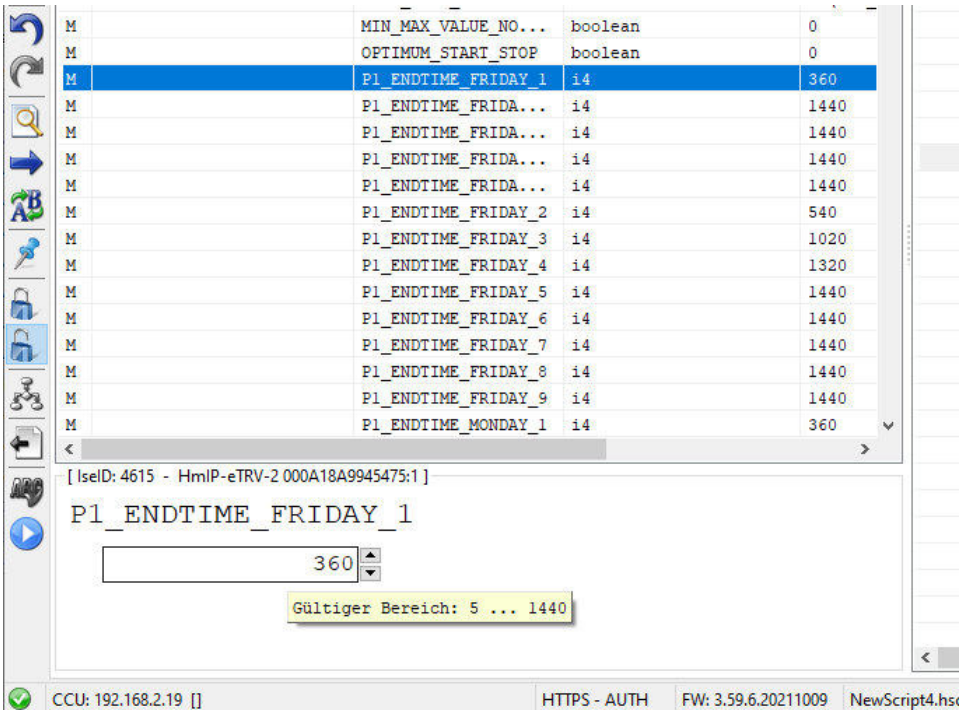
Der SDV kann mit Mastersets und auch Linksset umgehen. Mastersets werden automatisch dargestellt, wenn ein Gerät bzw Kanal betrachtet wird, der Mastersets enthält.

Bei Master und bei Linksets in der Anzeige nun komplette Informationen über den Parameter im Hint, auch wenn dieser eine Aufzählung ist

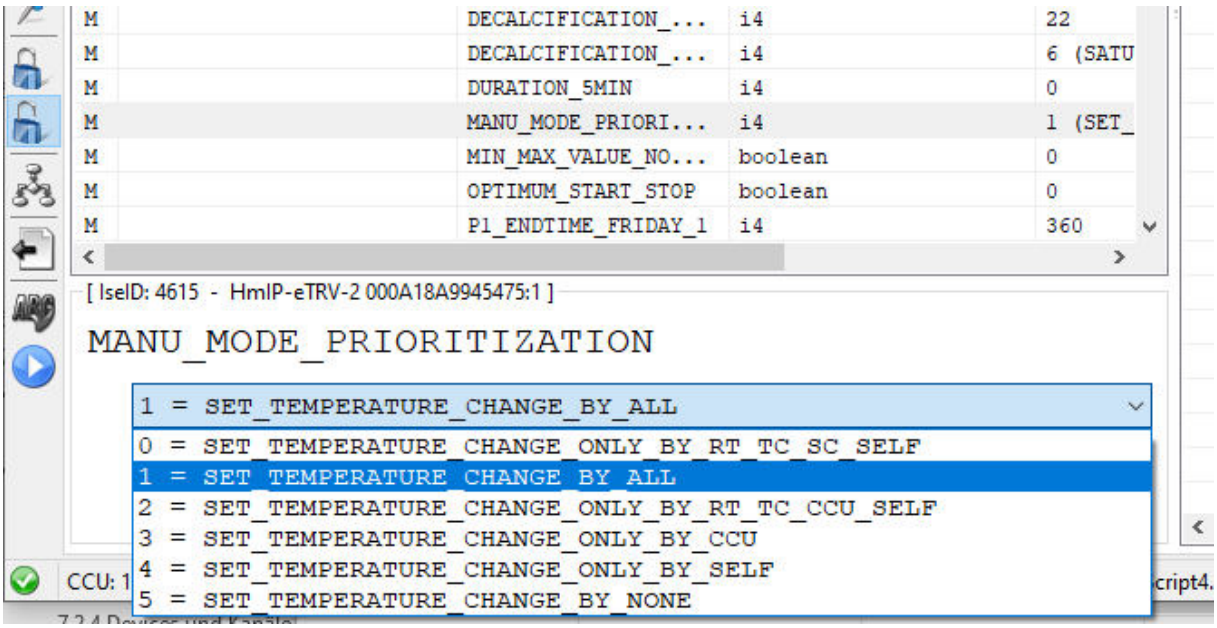


Nach dem großen „Erfolg“ der CCU Software, was die Nicht-Funktionalität der Änderung von Master oder auch Linkparametern angeht, Möglichkeit geschaffen, auch im SDV direkt die Master bzw Linkparameter zu ändern.

Um gerade für „Nicht-Profis“ derartige Änderungen einfacher zumachen, unterstützt der SDV hierbei, um Bedienfehler bestmöglich auszuschliessen.

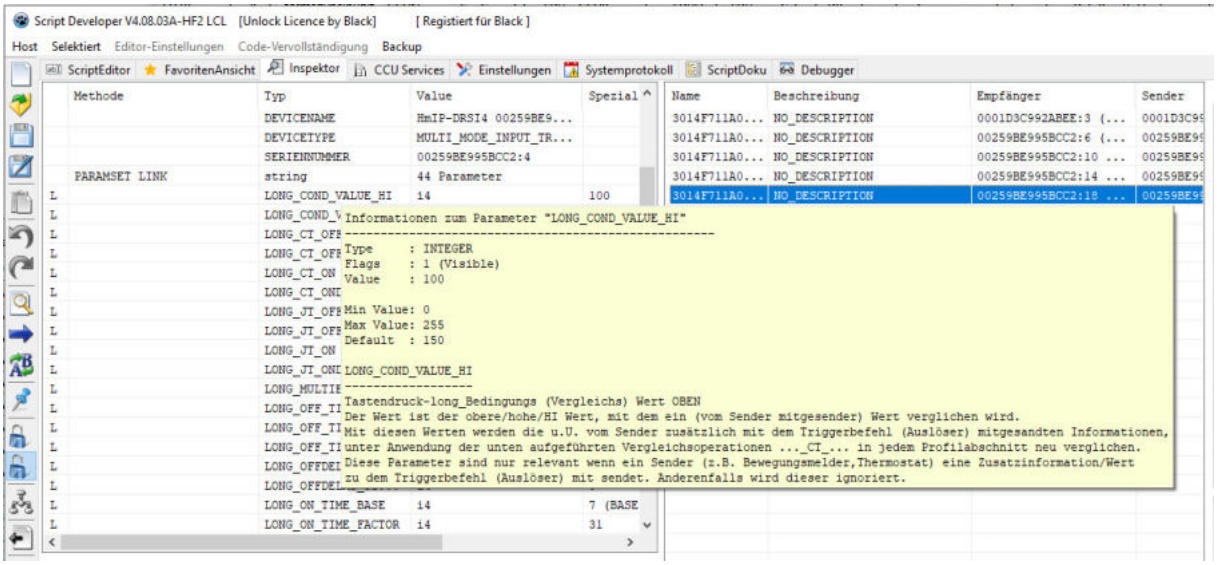


Richtiger Typ und von der Firmware gesetzter Wertebereich werden vorgegeben und überwacht. Bzw bei Aufzählungen wird der aufgelöste Text der Werteliste in einer Combobox vorgeschlagen.

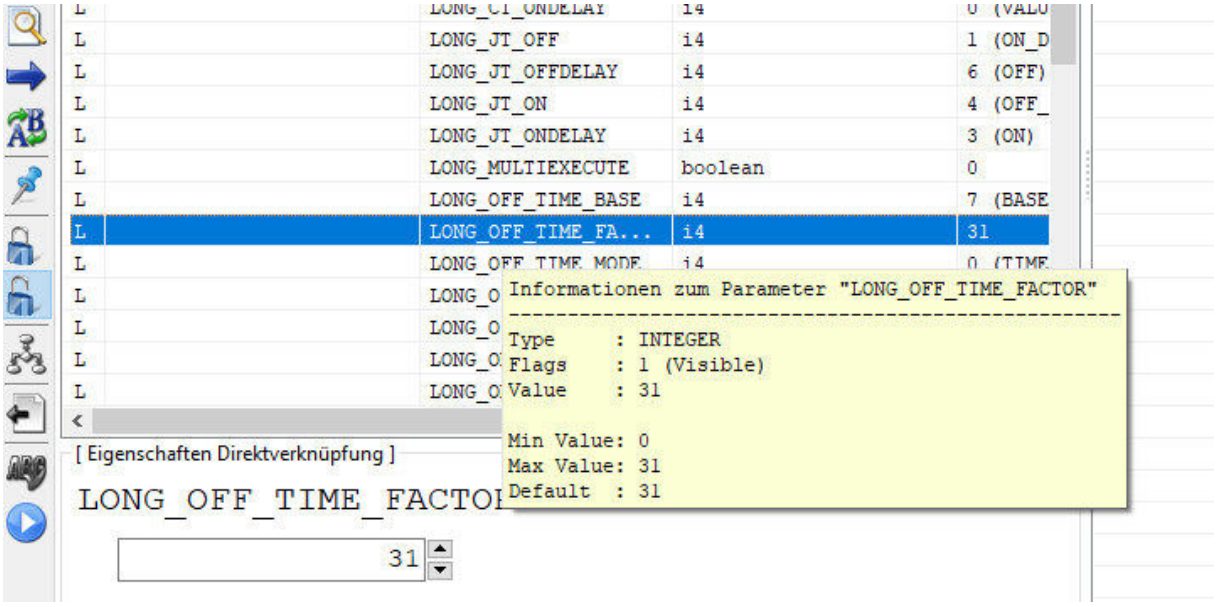


4.20.1 Linksets

Direktverknüpfungen lassen sich auf diese Art und Weise genauso Auflösen und verändern. Zu einigen der Parameter (bin noch nicht fertig) exitieren auch schon Beschreibungen in den Hints

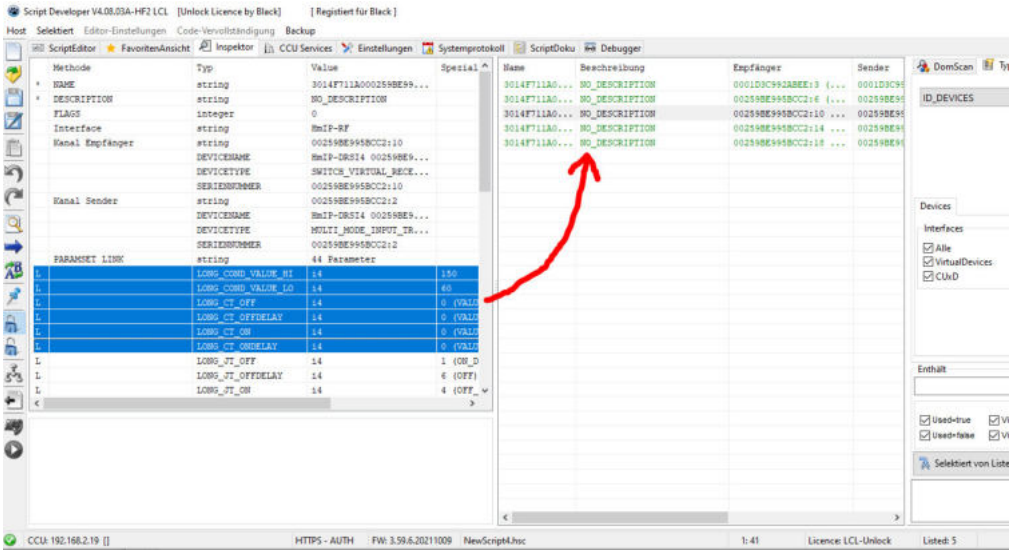


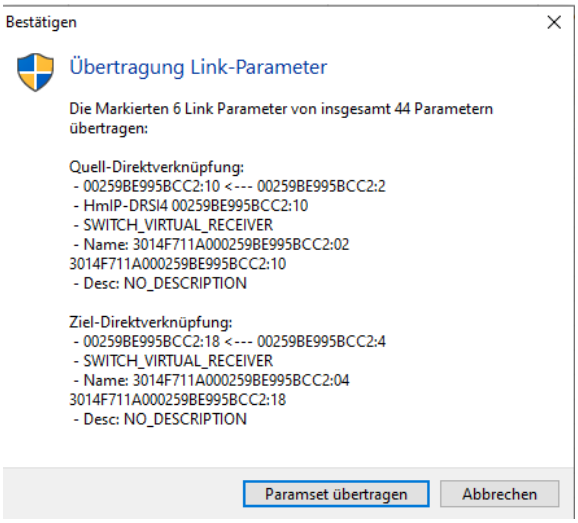
Die Werte jedes Parameters werden auch hier dargestellt, der Parameter kann wie bei den Mastersets beschrieben, auch geändert werden, oder wie im Editor beschrieben automatisiert in ein Skript zum setzen bzw. auslesen verwendet werden.



Einer der häufigen Anwendungsfälle bei mir ist Teile eines Linksets von der angewählten Direktverknüpfung in eine andere Direktverknüpfung zu verschieben.

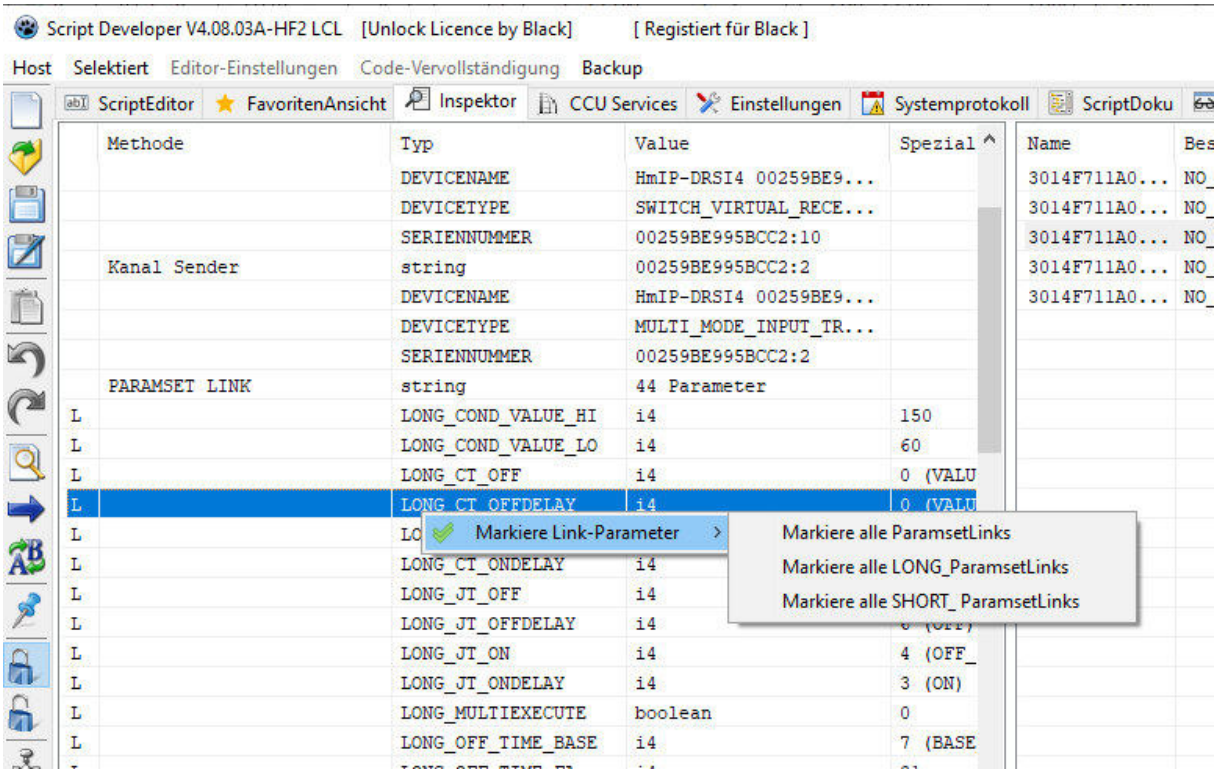
In der Quell Direktverknüpfung werden die werden die Parameter markiert. Mit Druck linke Maustaste und bewegen der Maus färben sich die DVs in der Listenansicht, in die die Parameter verschoben werden, nun grün. Mit der Maus kann nun via Drag and Drop die Ziel DV ausgewählt werden und mit Loslassen der Maustaste erfolgt der Transferprozess. Es erfolgt aber noch eine Sicherheitsabfrage.





Nach Druck auf „Paramset übertragen“ werden die ausgewählten Parameter von der QuellDV in die ZielDV übertragen. Sollten Parameter in der ZielDV nicht übertragen, so erfolgt eine Meldung über den nicht übertragenen Parameter.

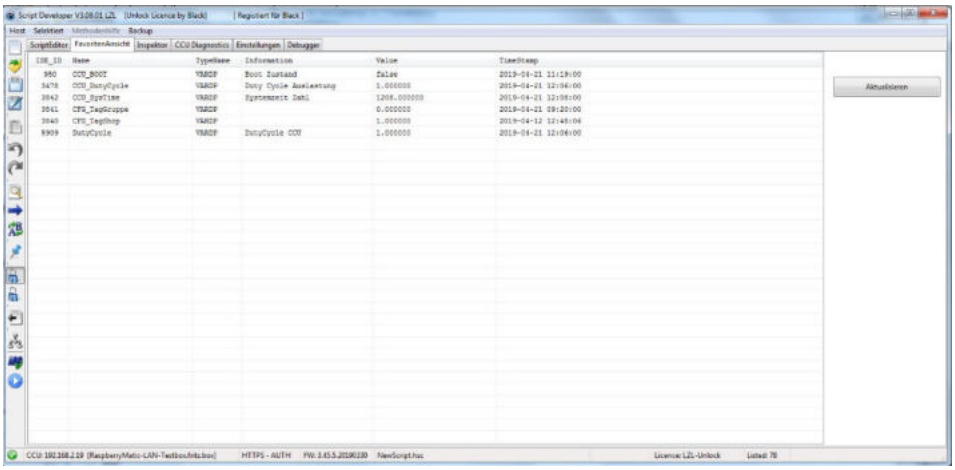
Will man nur alle LONG, Alle SHOTZ oder absolut alle Parameter übertragen so gibt es die schnelle Markiermöglichkeit. Rechte Maustaste in die Detailansicht der Direktverknüpfung rechte Maustaste, dann die Markierung auswählen:



Nach dem Markieren beginnt direkt der Drag and Drop Prozess, mögliche Ziele färben sich grün sie Maus kann bewegt werden. Abgebrochen wird mit Escape.

5 Favoritenansicht

Seit der Version V3.08.01 gibt es die Favoritenansicht. Diese soll dazu dienen, einen VariablenPool zusammenzustellen , dessen Werte und Zeitstempel sich übersichtlich darstellen und beobachten lassen.



Aktualisieren führt zum einen Nachladen der aktuellen Werte aus der CCU. In die Favoriten können Systemvariablen, Alarme, Datenpunkte und Programme übernommen werden.

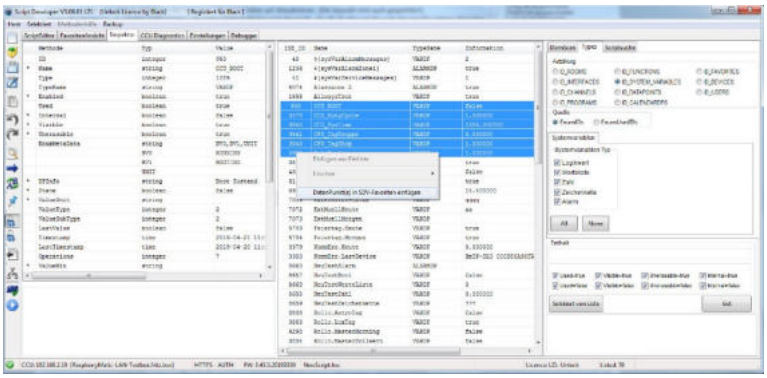
Will man einen Datenpunkt mal eben ändern, einen Doppelklick auf die Zeile der Favoriten, daraufhin wird unter Berücksichtigung des Undo / redo Stacks die Favoriten in die Listenansicht geladen und das angeklickte Objekt zur Bearbeitung in der Detailansicht geöffnet.

Der Inhalt der Favoritenliste wird in der SDV.INI automatisch gespeichert, und steht bei einem Neustart auch wieder zu Verfügung.

Mit Öffnen des Reiters FavoritenAnsicht werden die aktuellen Daten frisch von der CCU geholt

5.1 Hinzufügen von Objekten in die Favoriten

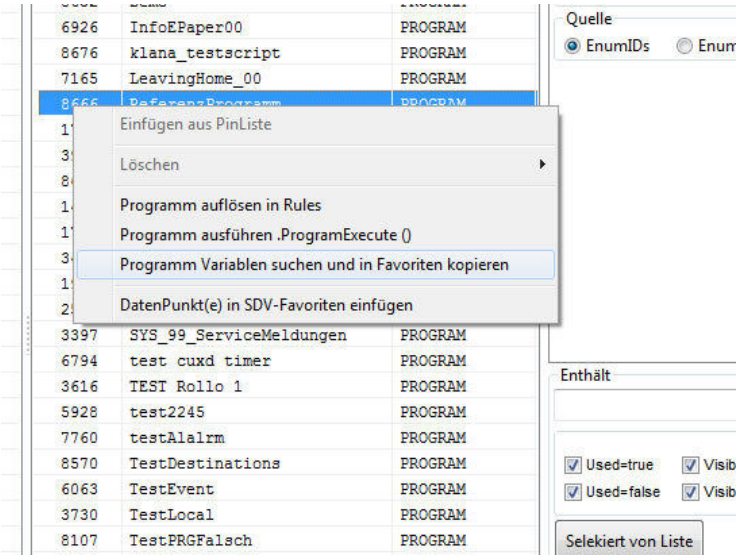
Objekte können aus der Listenansicht des Inspektors in die Favoriten geladen werden. Dazu Objekte Selektieren rechte Maustaste und



Neue Datenpunkte werden immer angehängt. Doppelte IDs werden unterdrückt, zweimal die gleiche Sysvar in den Favoriten geht also nicht.

5.2 Verwendete Objekte eines Programmes in die Favoriten laden

Zur Fehlersuche oder zum Testen möchte man öfters alle Werte, die ein Programm beeinflusst oder die von einem Programm beeinflusst werden, im Überblick haben. Dies geht mit dem SDV recht einfach. Dazu in der Listenansicht im Inspektor ein Programm selektieren, rechte Maustaste und „Programm variablen suchen und in Favoriten kopieren“



Darauf löst der SDV das Programmobjekt intern rekursiv auf und schreibt alle gefundenen Systemvariablen und Datenpunkte von Geräten in die FavoritenListe.

5.3 Favoritenliste löschen

In die Favoriten reinklicken, rechte Maustaste – Favoritenliste löschen

Es wird natürlich nur die interne Favoritenliste des SDV gelöscht, auf der CCU wird da nicht gelöscht oder verändert.

5.4 Einschränkungen

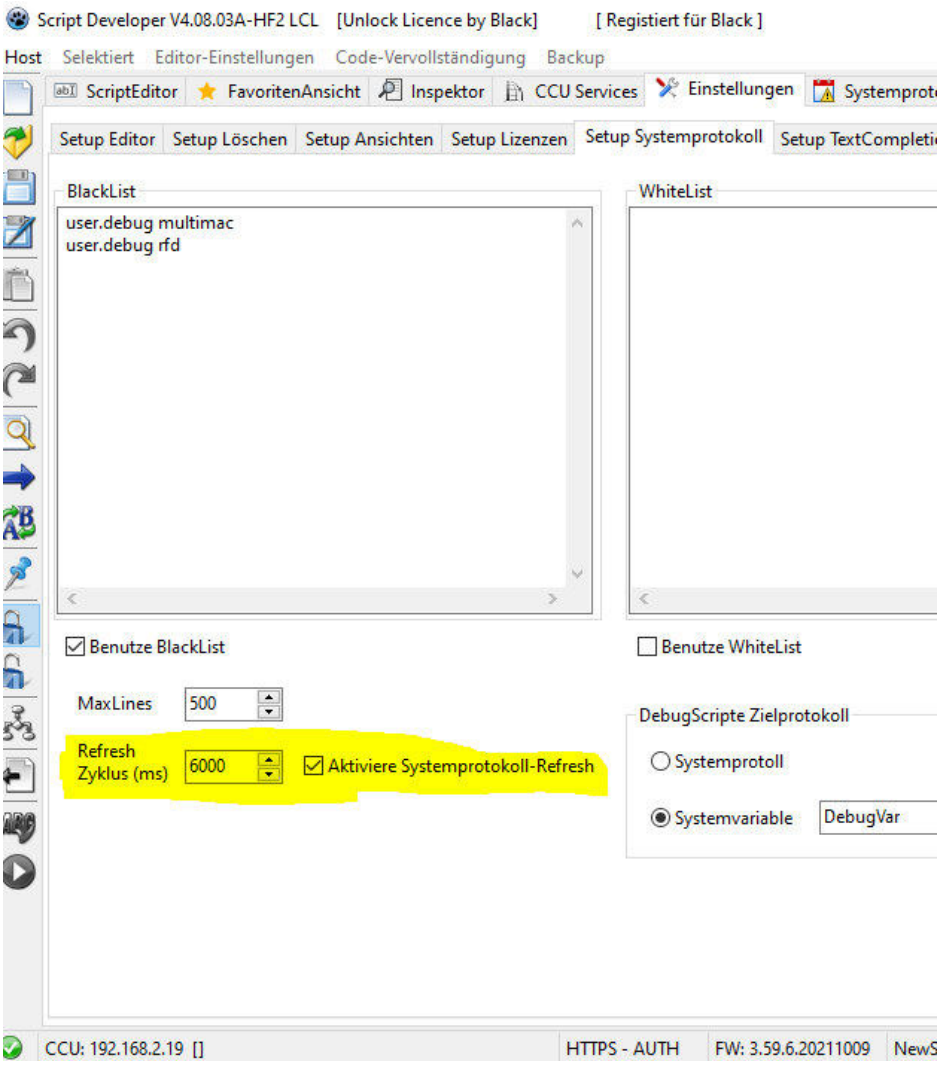
Die Favoritenliste arbeitet IseID orientiert. Wird von der CCU ein Object gelöscht, beispielsweise eine Systemvariable, so wird mit dem nächsten Aktualisieren diese auch von der Favoritenansicht gelöscht, da der ISEID Bezug nun ins Leere greift.

Beim Arbeiten mit 2 CCU´s bedeutet das:
Ein Wechsel der CCU als Host im SDV führt immer auch zu einer komplett anderen ISEID Liste. So wird es dann vorkommen, dass die Favoritenliste auf einmal leer sein wird, wenn der Host gewechselt wird.

5.5 Dynamisches Auffrischen von Favoriten

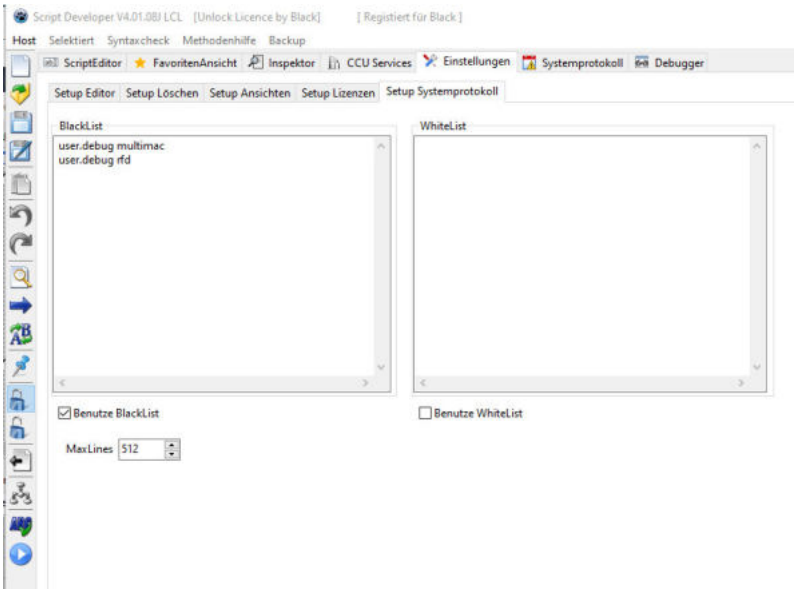
Es kann automatisches aktualisieren des Systemprotokolls angewählt werden. Die Refresh Zeit ist ebenfalls einstellbar. Das Refreh findet nur statt, wenn auch das Systemprotokoll geöffnet ist. Ist automatischer Refesh nicht gewünscht, ist es deaktivierbar.

Manueller Refresh ist mit F5 möglich



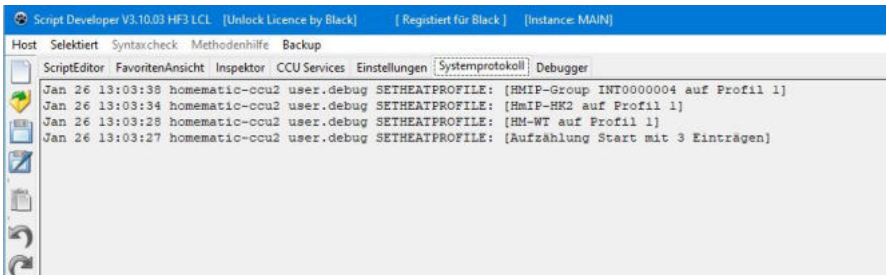
6 Systemprotokoll

Wird der Reiter angeklickt, so wird das Systemprotokoll der CCU abgerufen in den SDV und dort aufbereitet:
- der aktuellste Eintrag steht immer oben.
- es gibt eine Blacklist, LogEinträge, deren Textausschnitte in der Blacklist vorkommen, werden ignoriert (so benutze Blacklist angehakt ist). Damit blende ich z.B. zum programmieren auf meinem Spielesystem Meldungen des Multimac aus.
- es gibt eine Whitelist. Wenn benutze Whitelist angehakt ist, so wird ein Logeintrag angezeigt, wenn Textausschnitte davon in der Whitelist vorhanden sind (muss vorkommen, sonst wird ignoriert)



Nicht ganz reproduzierbar gibt es manchmal Probleme bei der Übertragung sehr langer Texte, so dass sich die Maximale Zeilenlänge des Protokolls einstellen lässt.

Nützlich in dem Fall, wo Zeitverzögerte Aktionen und Einträge überwacht werden sollen. Die Skriptausgabe des Editors nimmt ja nur den Moment des Skriptausführens. Konkreter Anwendungsfall: das Skript zum zeitlich versetzten Umschalten von Wochenprofilen in verschiedenartigen Geräten:

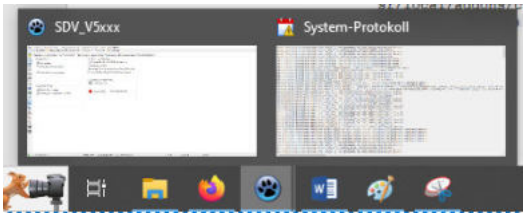


Folgende Schlüssel dazu in der INI nachtragen (bei Bestandsinstallationen)

```
[SYSLOG]
USEBLACKLIST=True
USEWHITELIST=False
BLACKLIST=user.debug multimac\r\nuser.info multimac\r\nuser.debug rfd
WHITELIST=
```

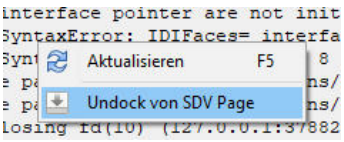
6.1 Undock und Dock des Protokollfensters

Wenn gewünscht (z.b. bei Verwendung von zwei Monitoren) kann der Reiter Systemprotokoll ausgedockt werden und in einem separaten, frei verschiebbaren Fenster dargestellt werden. Im ausgedockten Zustand werden in der Taskleiste auch beide Symbole dargestellt:



6.1.1 Undock

Undock löst das Systemprotokoll aus dem Reiter des SDV und stellt es in einem separaten Fenster dar. Dazu rechte Maustaste ins Systemprotokoll und Undock von SDV Page auswählen.



6.1.2 Dock

Dock löscht das Separate Fenster wieder und stellt das Systemprotokoll wieder in dem SDV Reiter dar.
Dazu gibt es 2 Möglichkeiten

1. Das separate Fenster einfach Schließen
2. In dem Systemprotollfenster rechte Maustaste und Dock an SDV Page auswählen:

7 Kleine Helfer im Alltag

7.1 Debugging von WebUi Programmen

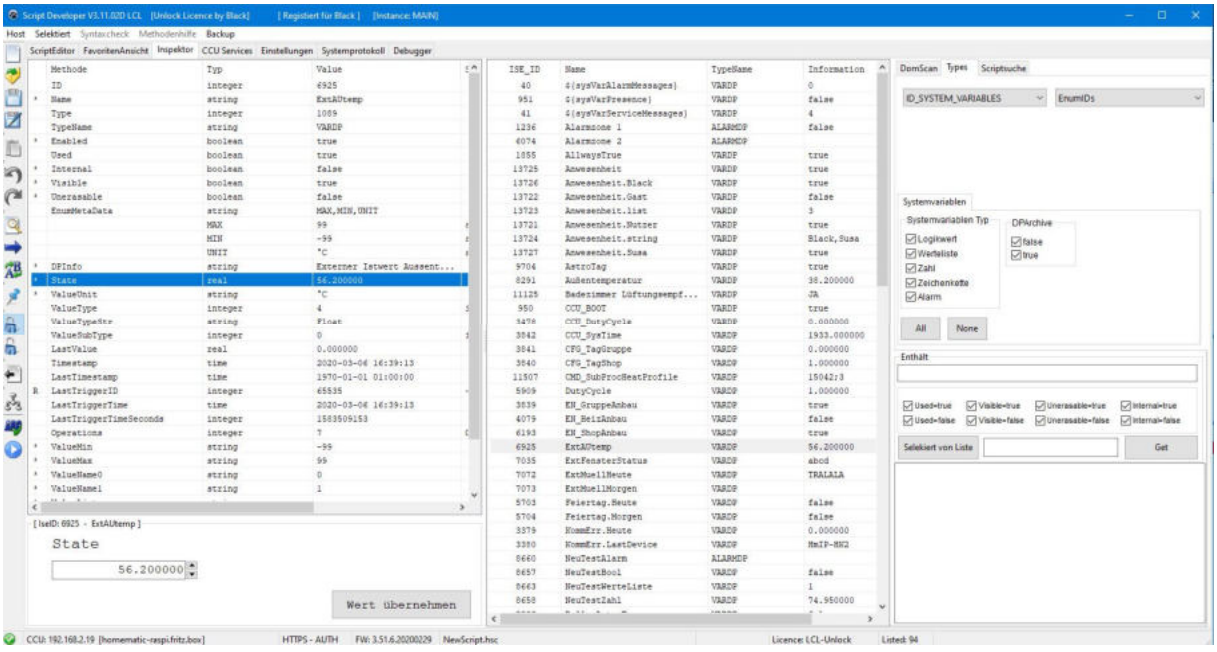
Mit der Version 3.11.02 wurden die ziemlich weitreichen Test und Eingriffsmöglichkeiten auch für niedrigere Lizenzlevel freigeben. Da ich schon einige PN Anfragen bezüglich der Möglichkeiten dazu hatte, nun mal eine allgemeine Anleitung für die breite Masse. Die Stammtischmitglieder kannten diese Möglichkeiten ja schon seit langem.

Ansatz.

Man hat ein WebUI Programm geschrieben und möchte dieses nun testen in Realumgebung, also mit den Geräten etc. Wenn man nun Dinge testen möchte wie z.B Helligkeiten, Temperaturen, Luftfeuchtigkeit und vieles mehr war oft der weg, entweder warten bis sich ein entsprechendes Scenario einstellt, den Geräten mit Feuchtigkeit, Heißluft Fön etc zu Leibe rücken, sämtliche Werte über Systemvariablen zwischenkoppeln (so die allgemein üblichen Lösungswege)

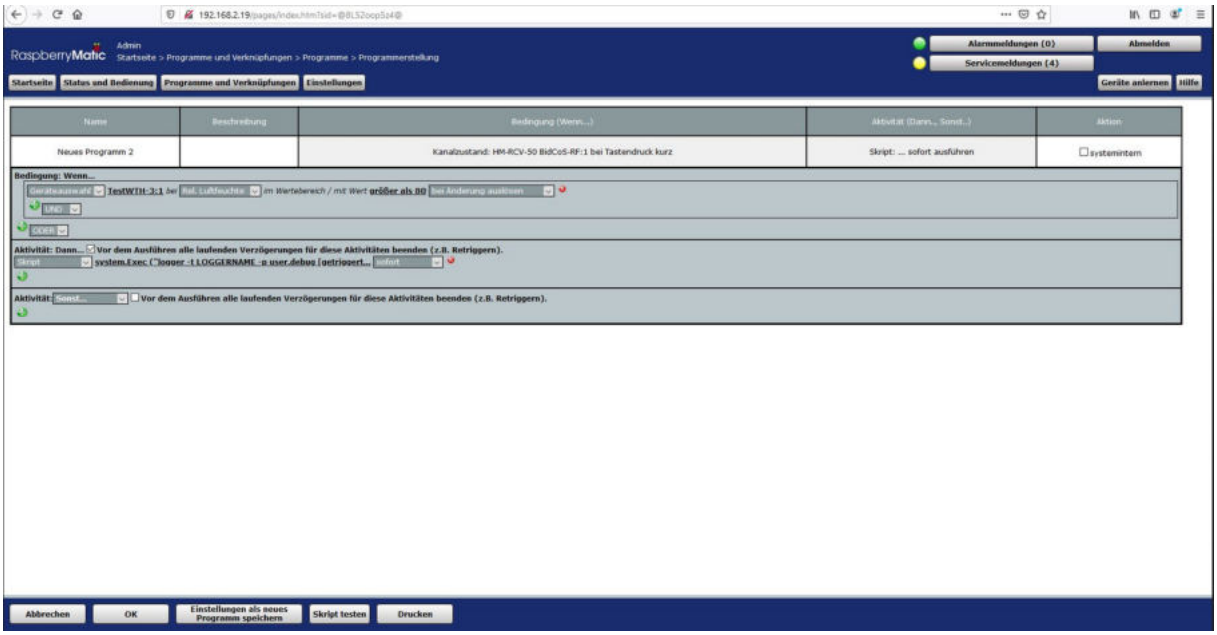
Es gibt noch einen anderen, eigentlich ist dieser sogar von EQ3 dokumentiert (offizielle xmlrpc Dokumentation, Kapitel 5.1 Methoden der Logigschicht - event. also keine Geheimnisse, an die man nur mit einem Disassembler kommt oder die man hüten müsste wie die Abschusscodes von Atomraketen  über diese Logigschicht Methode teilen die Schnittstellenprozesse der Rega Änderungen ihrer Zustände mit. Und das ist dann der Punkt an dem sich der SDV auch einklinkt.

Anwender des SDV kannten ja schon die Möglichkeiten Datenpunkte oder Systemvariablen mit der Eigenschaft operation_write mit dem SDV zu verändern.



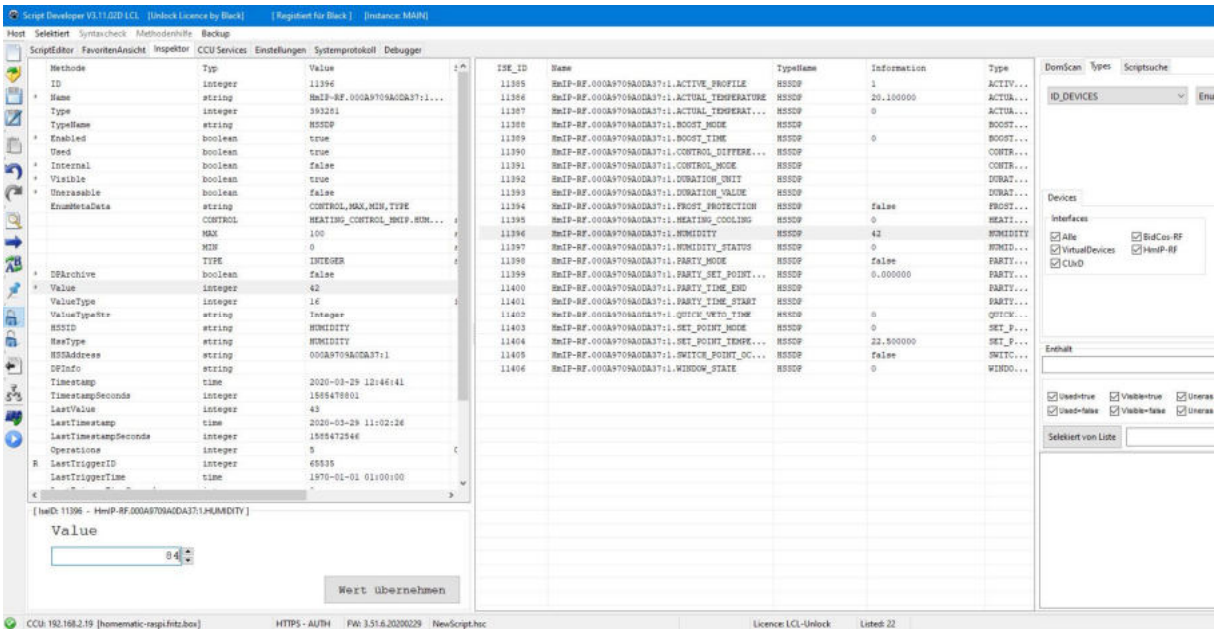
Die Systemvariable bzw der Datenpunkt kann so einfach geändert werden zum Testen. Das ist auch nicht so die Große Schwierigkeit. interessant wird nun, ich möchte ein Programm schreiben, welches mit bei Änderung der Luftfeuchtigkeit eines WT2 auf mehr als 80 Prozent ein Programm triggert

Eigentlich ein kleines WebUI Programm nun, um dies zu demonstrieren.

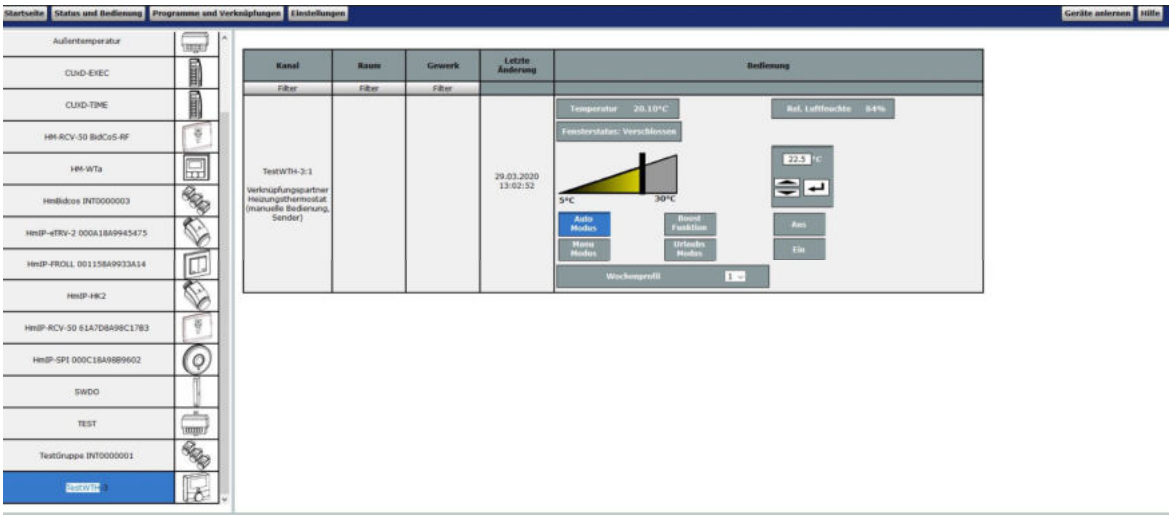


Nun wollen wir allerdings die Funktion dieses Programmes prüfen. Dazu müssen wir den Humidity Datenpunkte ändern. Der Datenpunkt Humidity ist allerdings einop eration_read, operation_event, also kein Datenpunkt, den wir mit state oder value irgendwie geändert bekommen. trotzdem lässt sich im SDV nun dieser Datenpunkt bei Value anwählen und editieren:

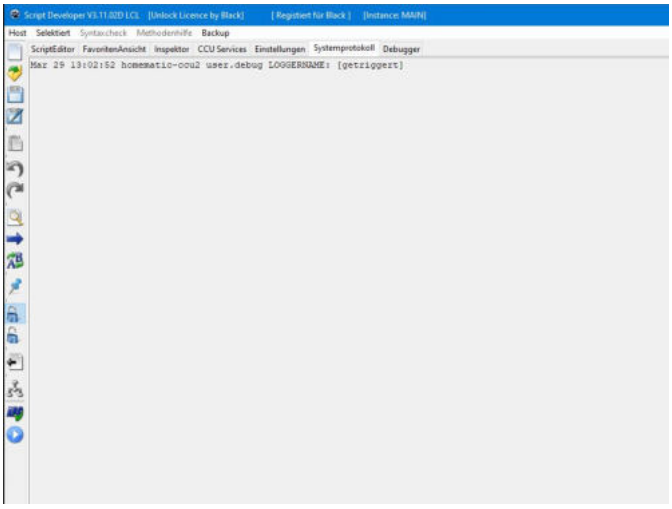
wir wollen testen, also stellen wir anstatt der gemessenen 42% nun mal 84% ein und sagen Wert übernehmen. Die Funktion Des SDv erkennt selbstständig, ob dieses ein Datenunkt ist der sich normal über .State () oder .Value () umschreiben lässt, oder aber ob der etwas brutalere weg über xmlrpc.event zu nehmen ist.



ein Blick in die WebUi verrät uns: die Rega hatte die zwangsweise untergeschobenen Daten übernommen.



wir konnten sehen, das Programm hatte getriggert:



der Zustand bleibt natürlich so lange bestehen bis... logischerweise der xmlrpc process über den gleichen Weg der Rega ein neues Messergebnis mitteilt. es ist ja der gleiche Weg. Allerdings lassen sich so sämtliche Datenpunkte der Rega setzen bzw verbiegen. Auch zum Testen von Systemmeldungen auf dem Channel:0 funktioniert diese Vorgehensweise. Wenn einer beispielsweise Skripte für Kommunikationsstörungen etc. schreibt, so kann man auf dem Wege zum Testen Störungen setzen und auch wieder wegnehmen.

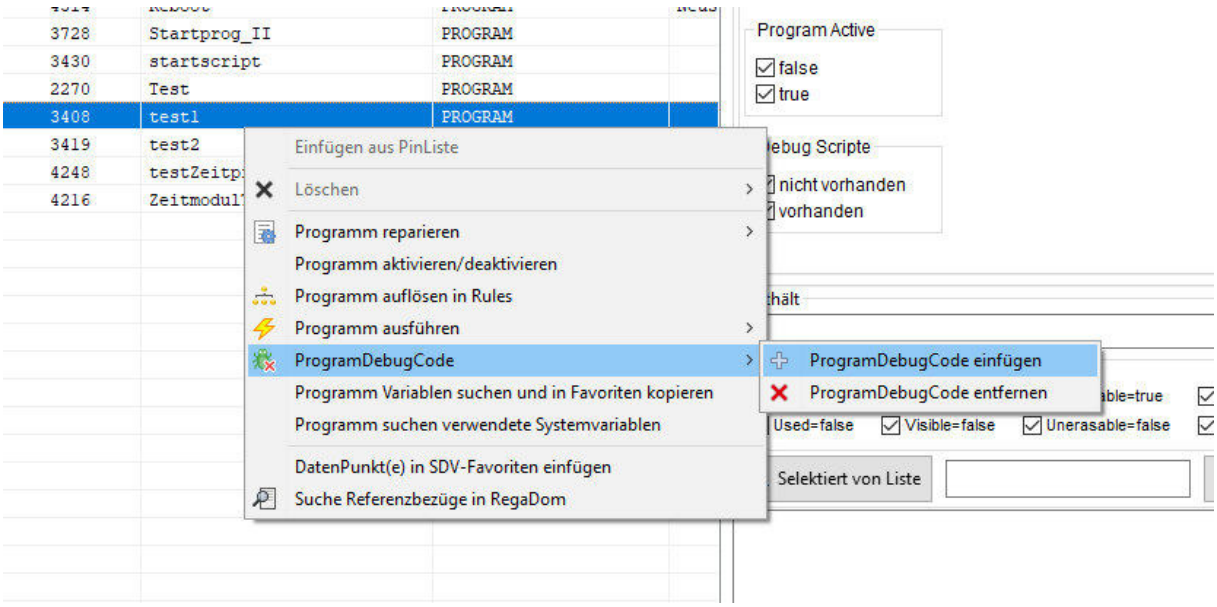
Logischerweise hier der Einwurf:
Das ist kein Spielzeug um irgendwelche Störungen wegzuklicken (geht) oder wahllos irgendwelche Daten zu setzen (geht auch) sondern um Sinnvoll mit der CCU zu arbeiten bzw Programme zu testen oder Fehler zu suchen und zu simulieren.

Aus dem Grunde war diese Funktion früher auch höhreren Level vorbehalten.

und: ein zwangsumschreiben der Rega auf dem Weg wirkt sich nicht auf den xmlrpc aus, ein IObroker z.b. bekommt von der IchÄnderEinenDatapointperEvent nix mit, da IOBroker direkt auf den RPC Prozess aufsetzt und von diesem die direkten Daten geliefert bekommt.

7.1.2 Automatisiertes Einfügen von DebugSkripten

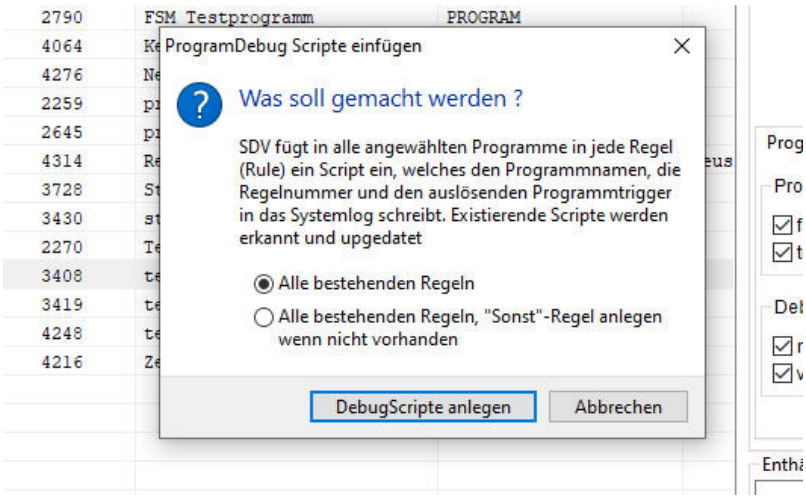
In ein oder mehrere, selektierte Programme lassen sich im Inspektor Debugskripte einfügen. Dieses Debugskript schreibt entweder in den „richtigen“ Syslog oder in das CCU Systemprotoll einen Eintrag in der Art: Welches Programm wurde wann mit welchem Typ Datenpunkt mit welchem Wert ausgelöst und in welcher Bedingung des Programmes wurde dies Ausgeführt (Wenn, 1. Sonst Wenn, x. Sonst Wenn, Sonst)



Der Code kann eingefügt und auch wieder gelöscht werden.

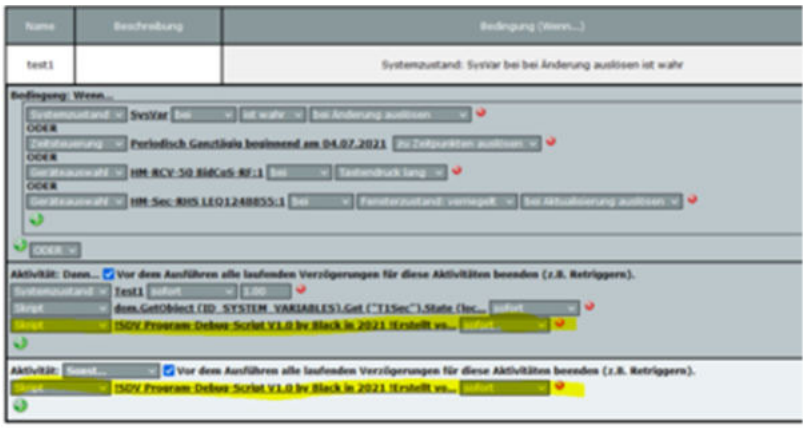
rechte Maustaste
ProgrammDebugCode - Program Debug Code einfügen.

es öffnet sich Bestätigungsmenü:

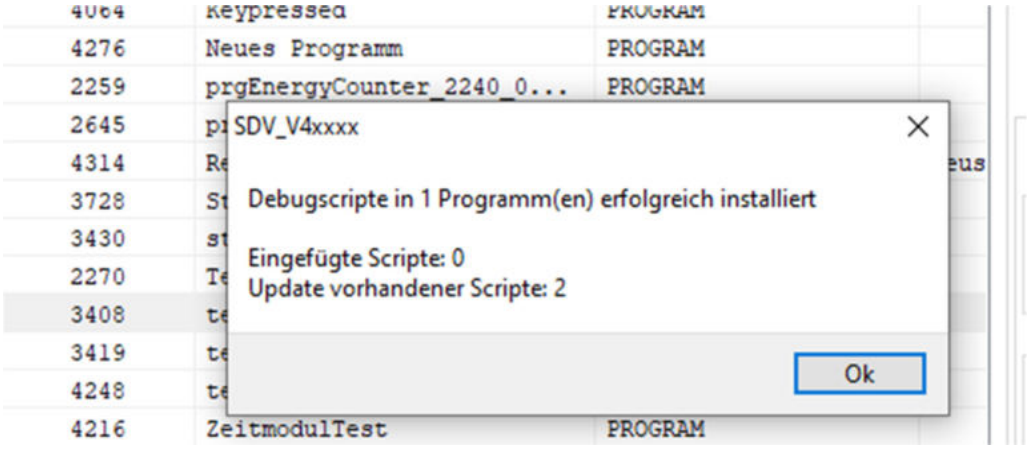


Auswahl:
- alle bestehenden Regeln:
in jede Regel wird im Ausführungsteil das Debug Skript eingefügt. Wenn dieses schon existiert, so wird das Skript nur aktualisiert, es wird kein neues angelegt. Wenn es keinen Sonst teil gibt, so wird im Sonst Teil auch kein Skript angelegt
- alle bestehenden Regeln, Sonst Regel anlegen wenn nicht vorhanden:
in jede Regel wird im Ausführungsteil das Debug Skript eingefügt. Wenn dieses schon existiert, so wird das Skript nur aktualisiert, es wird kein neues angelegt. Wenn es keinen Sonst teil gibt, so wird der Sonst Teil angelegt und dort ebenfalls das Debug Script angelegt.

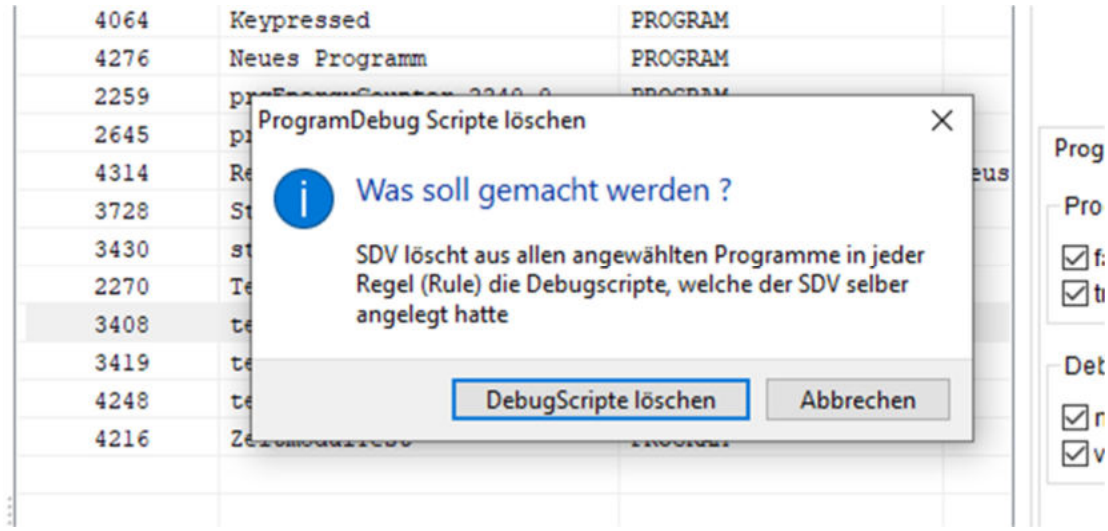
In dem Debugskript wird vom SDV auch Automatisiert die Regelposition eingetragen, Die Wenn Regel wird mit WENN gekennzeichnet,
Die "SONST WENN" werden durchnummeriert eingetragen, gibt es 3 Sonst-Wenns in einem Programm, so werden diese auch automatisch mit1. SONST WENN , 2. SONST WENN etc gekennzeichnet, ein SONST wird ebenfalls erkannt und automatisch gekennzeichnet.
Damit lässt ich im Systemprotoll sofort die ausgeführte Regel erkennen.



Nach dem Ausführen von Skripte einfügen wird ein kurzer Report dargestellt, in wie vielen Programmen wurde in wie vielen Regeln das Skript eingeführt bzw in wie vielen Regeln wurde das Skript geupdatet.



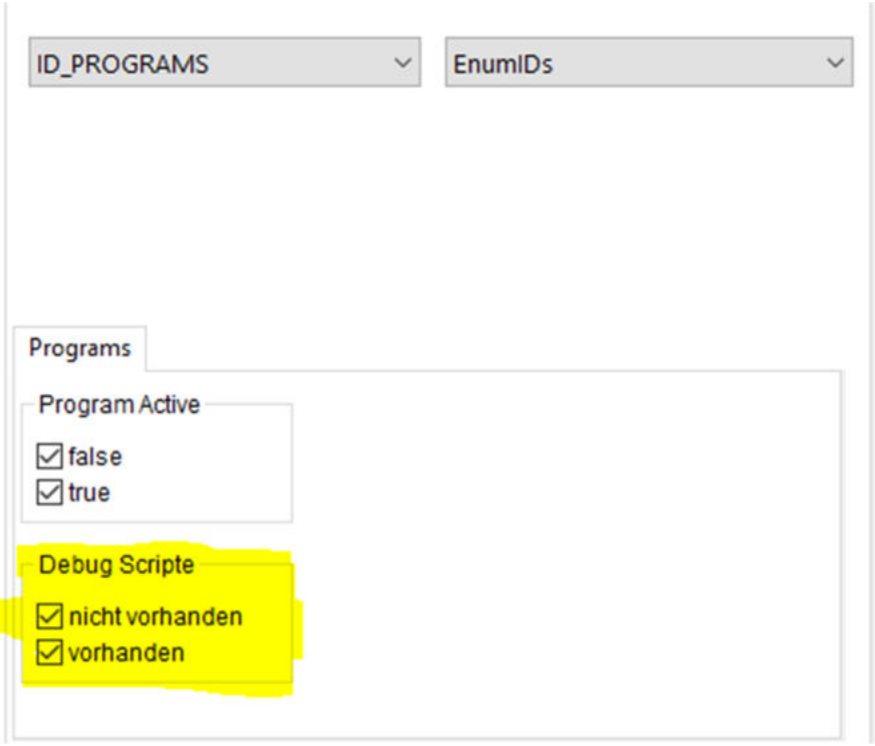
DebugSkripte wieder entfernen.
Programme auswählen rechte Maustaste
ProgrammDebugCode - Program Debug Code entfernen.



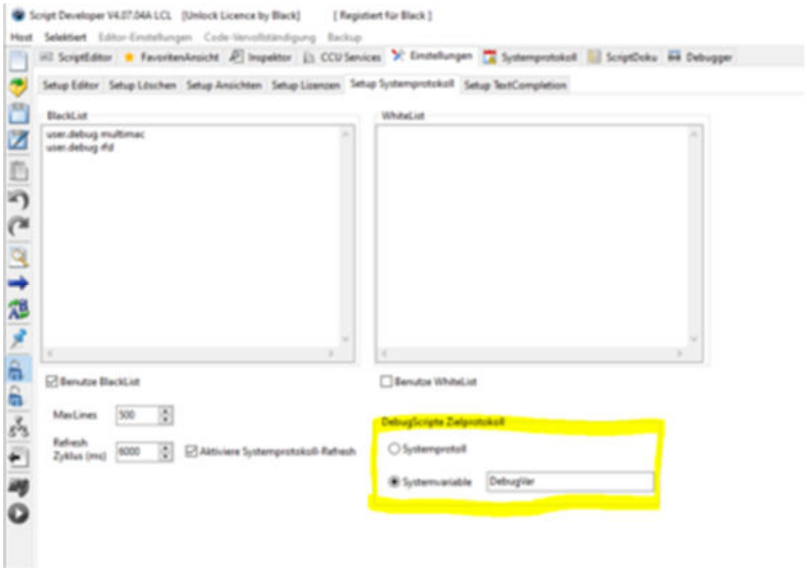
in dem sich öffnenden Bestätigung Menü das Löschen bestätigen.

Nach dem Löschen von Skripte einfügen wird ein kurzer Report dargestellt, in wie vielen Programmen wurde in wievielen Regeln das Skript gelöscht.

Selektieren von Programmen mit Debugskripten / ohne Debugskripten
im Inspektor lässt sich unter ID_PROGRAMS anwählen ob nach
- Programmen mit enthaltenem Debugskript
- Programmen ohne enthaltenem DebugSkript
oder egal ob enthalten oder nicht selektiert werden soll



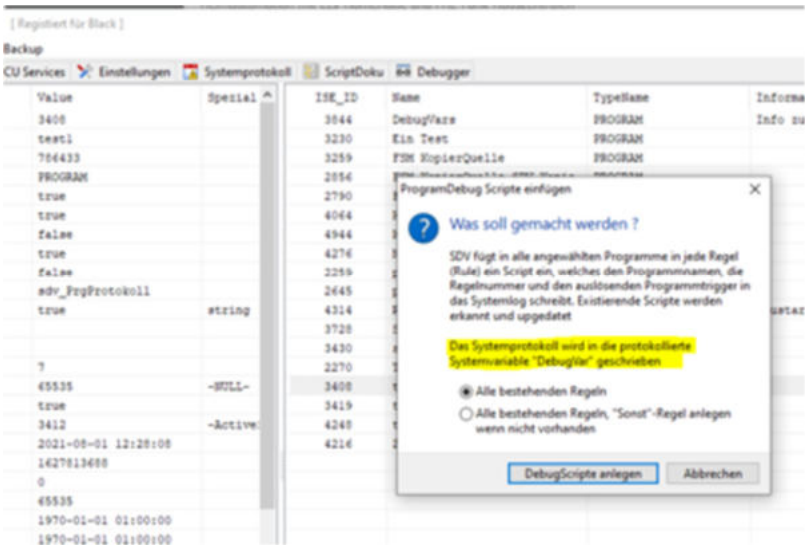
DebuggigScript optional mit Möglichkeit, den Debugtext auch in eine protokollierte Systemvariable zu schreiben.
dazu kann hier definiert werden:
wie bisher: schreiben ins "richtige" Systemprotokoll
oder
schreiben in die hier angegebene Systemvariable.



Hilfetexte sind dazu hinterlegt, die Einstellungen werden in der XML automatisch persistiert

Ist Systemvariable ausgewählt aber keine Systemvariable angegeben: dann gibt es auch keinen Menüeintag: Debugtext schreiben
existiert die Systemvariable nicht auf der CCU, so wird der SDV beim Schreiben der Debugskripte diese Systemvariable typ string anlegen.
existiert die Systemvariable schon ist aber kein String, so wird diese gelöscht und als String neu angelegt.
SDV stellt die Systemvariable automatisch als protokoliert ein.
Der eingegebene Name sollte schon möglich sein... $%&[]§§ lässt sich zwar eintragen, das wäre allerdings dämlich.

Im Menü für das Bestätigen des Skripteintragens kommt noch der Hinweis, wenn Systemvariable angewählt, wo dieses eingetragen wird.



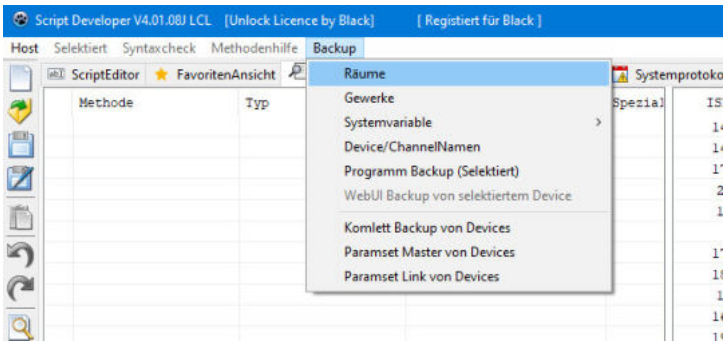
7.2 Backups

Von relevanten Objekten können Backups gemacht werden. Diese ersetzen KEIN richtiges SystemBackup an der CCU !!!
Vielmehr dienen diese im Falle eines Umzuges von einem alt 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.
Ab Version 3.09.03 besteht die Möglichkeit, jeder CCU ein separates Backup Verzeichnis zuzuordnen
Damit werden Verwechslungen vermieden, wenn man mit 2 CCU arbeitet. (Beide CCUs haben ein jeweils eigenes Backup-Verzeichnis)

z.B.

```
[CCU1]
IP=192.168.2.19
USERNAME=ScriptAdmin
PASSWORD=xxxxxxxxxxxxxxxxx
USEHTTPS=true
SSHUSERNAME=root
SSHPW=xxxxxxxxxxxxxxxxxx
BACKUPDIR=C:\MTH\Homematic\Backups\CCU_19\
```

Den passenden Lizenzlevel vorausgesetzt, finden 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 an.

Hinweis an die Bestandsnutzer nach Wechsel auf die breaking Change Rega

Aufgrund des geänderten Escaping Verhaltes ist es erforderlich, alle Backups nochmal laufen zu lassen. Grund: Die Wiederherstellungsroutinen können sonst aufgrund des Scritpruntimeerrors, welcher durch das geänderte Escaping kommt, abbrechen

7.2.1 Räume

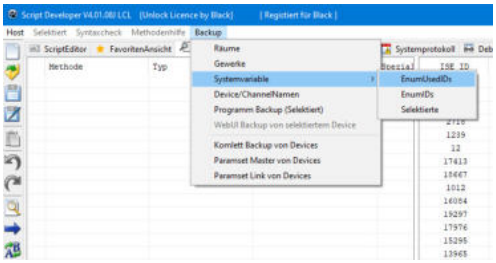
Auf dem Bestandssystem wird ein Restore Programm erzeugt und lässt sich anschließend auf dem PC abspeichern. Der vorgeschlagene Dateiname ist dabei backup_Rooms_ + datum und Uhrzeit der Generierung.
Auf dem „Neusystem“ lässt sich dieses Programm über den SDV via Skripteditor dann starten.
Dabei passiert 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.

7.2.2 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 Skripteditor 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.

7.2.3 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.
Die zu sichernden Systemvariablen können selektiert werden:



EnumUsedIDs (): Alle Systemvariablen, die in der Aufzählung EnumUsedIDs() gefunden werden, werden gesichert
EnumIDs (): Alle Systemvariablen, die in der Aufzählung EnumIDs() gefunden werden, werden gesichert
Selektierte: Alle Systemvariablen, die in der Listenansicht selektiert sind, werden gesichert.

Auf dem „Neusystem“ lässt sich dieses Programm über den SDV via Skripteditor 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 exitierte
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 Alarme 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)

7.2.4 Devices und Kanäle

Bei diesem Backup werden von den selektierten Geräten 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.

7.2.5 Backup Programme

Über diesen Punkt kann von einem oder mehreren WEB-UI Programmen Backups gezogen und als Datei (JSON bzw Ausführbares HM-Skript) abgespeichert werden. Mit diesen Programmen lässt dich ein WEB-Ui Programm vollständig wiederherstellen. Dabei arbeitet der restore nicht ISEID bezogen, mit dem Restore Programm lässt sich ein WebUI Programm auch auf einem komplett anderen System wiederherstellen, so natürlich die verwendeten Datenpunkte in symbolischer Adressierung und vom richtigen Typ vorhanden sind:

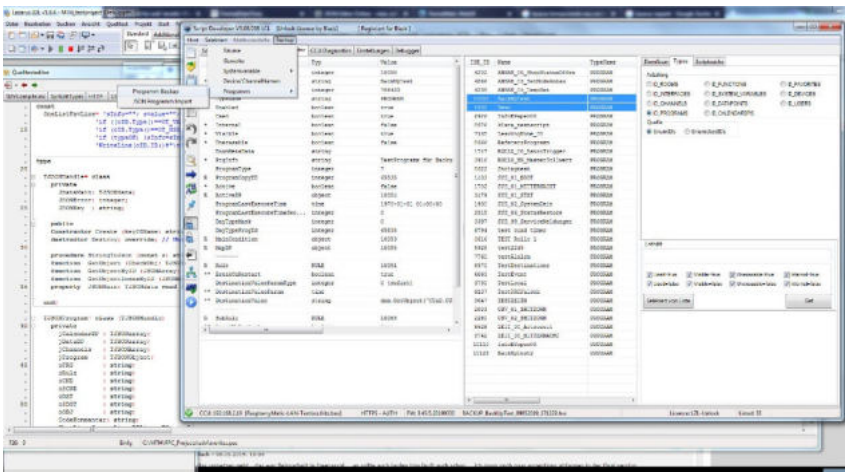
Für die Programme gibt es die Möglichkeit, den Backupnamen in der INI festzulegen.

```
[LAST]
(…)
BACKUPNAME=$NAME_BACKUP_$FW_$DATE_$TIME
```

Bedeutung der Abkürzungen

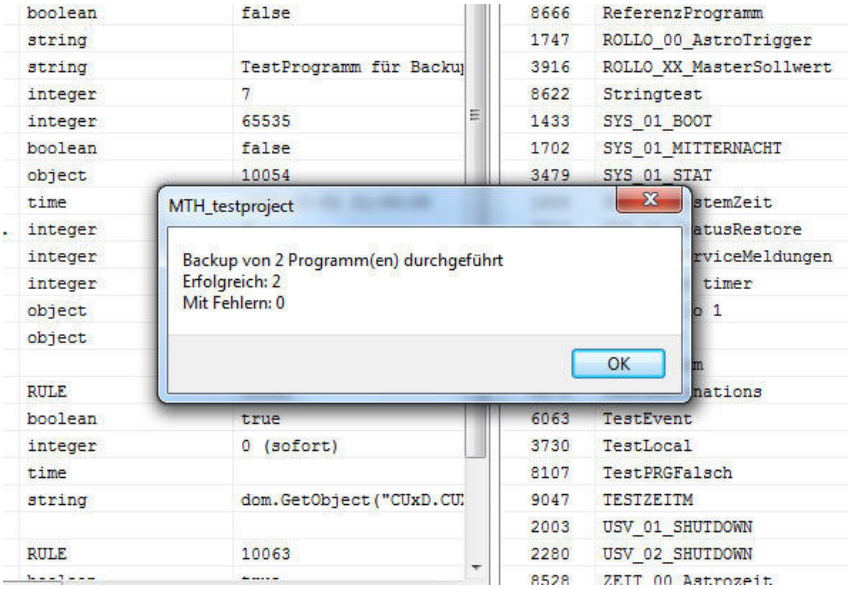
$NAME: der Name des Programmes wird eingesetzt
$FW: Firmwareversion
$DATE: Datum in der Form JJJJMMTT
$TIME: Zeit in der Form HHMMSS

Programme in der Listenauswahl selektieren, eins oder mehrere...



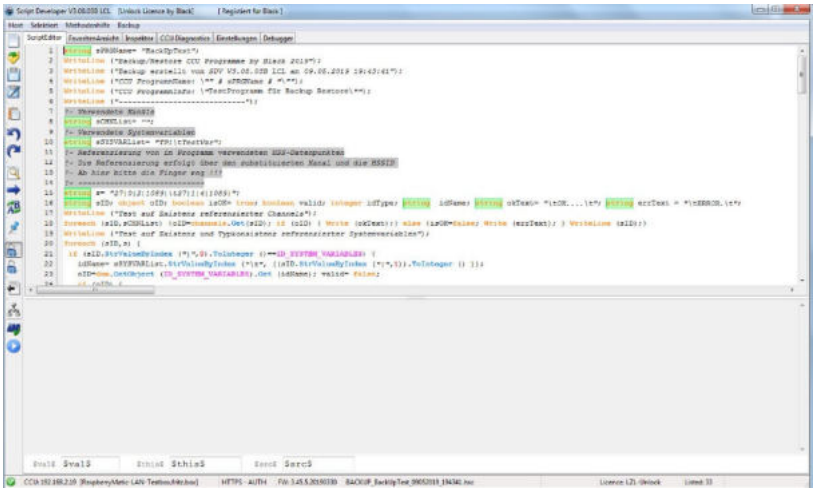
Danach in dem sich öffnenden Dialog das Verzeichnis Auswählen, wo das Backup hin gespeichert werden soll:
Der Programmname heisst 'BACKUP_ + ProgrammName + _ + TTMMYYYY_HHMMSS.hsc bzw .json
danach werden die Backups der ausgewählten Programme angelegt.
werden Objekte selektiert, die keine Programme sind, so werden diese natürlich ignoriert.

So sollte es aussehen, wenn alles durchlief:



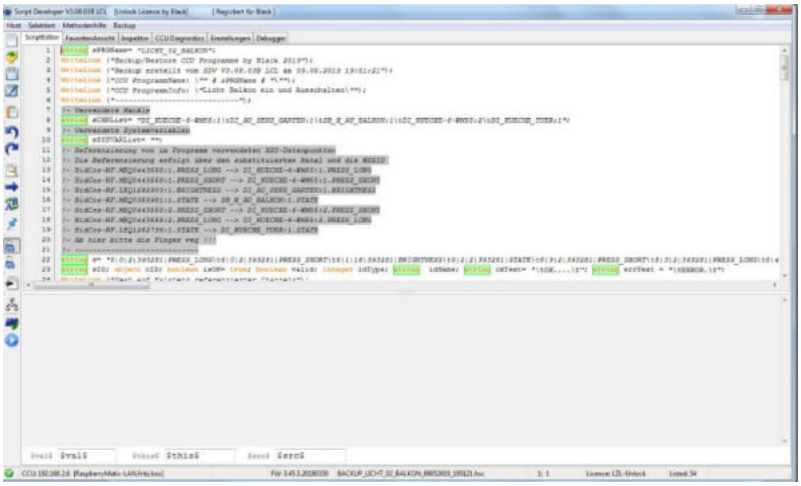
ab V8.03.04 wird auch noch die Laufzeit für das Backup in Sekunden angezeigt. In der unteren Infozeile erfolgt Anzeige ganz rechts, welches Programm von wie vielen gerade im Backupprozess ist. Danach liegen im Verzeichnis 2 Files.

das hsc kann z.b. in den SDV geladen werden und dort ausgeführt werden.
Der Kopf sieht immer so aus:



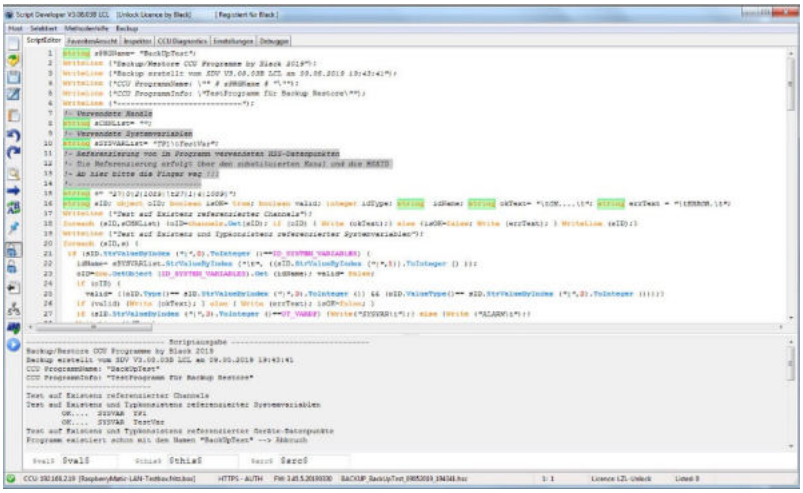
die ersten Zeilen können auch Händisch angepasst werden, wenn man den Weiß, was man tut.

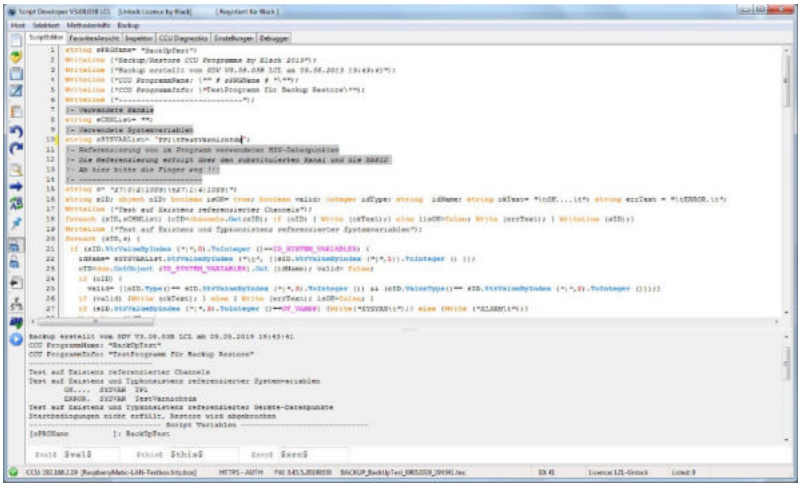
Programmname ist selbsterklärend, die Einträge unter verwendete Kanäle und verwendete Systemvariablen können verändert werden. (beispielsweise Rollo Programm, wenn man einen neuen Aktor benutzt... oder Hm auf HMIP umbaut. es muss nur der alte auf den neuen Kanal geändert werden, das ID zusammenbastel Gedöns macht dann das Programm. wo was substituiert wurde steht auch als Test im Programm. Hier sieht man es besser, Der SDV trägt in die Kommentarzeilen unter der Tabelle ein, welcher Datenpunkt wie über Kanal - HSSID zusammengesetzt wurde.



Zum Restore.

Es wird überprüft, ob das Programm schon existiert... natürlich Abbruch.
auch ob alle verwendeten Datenpunkte (Sysvars Kanäle oder DPs) vorhanden sind und auch den richtigen Typ haben. Originale Sysvar BOOL und auf dem neuen System sysvar String erkennt der SDV und bricht vor der Generierung ab.





Bei einem Erfolgreichen Durchlauf sieht die Skriptausgabe so aus:

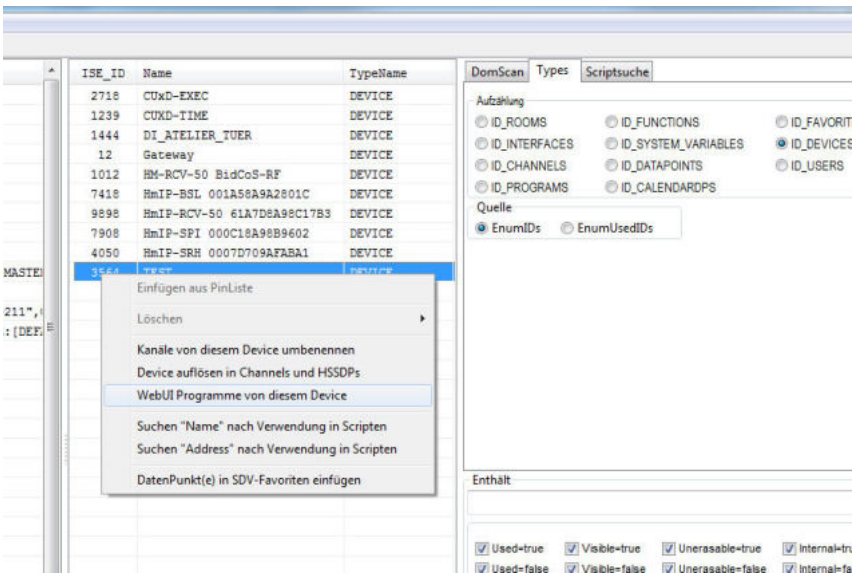
```
Backup erstellt vom SDV V3.08.03B LCL am 09.05.2019 19:43:41
CCU ProgrammName: "BackUpTest"
CCU ProgrammInfo: "TestProgramm für Backup Restore"
---------------------------
Test auf Existenz referenzierter Channels
Test auf Existenz und Typkonsistenz referenzierter Systemvariablen
        OK....  SYSVAR  TP1
        OK....  SYSVAR  TestVar
Test auf Existenz und Typkonsistenz referenzierter Geräte-Datenpunkte
Restore Programm von Program "BackUpTest" erfolgreich durchgelaufen
------------------------------ Script Variablen -------------------------
```

Im Hinblick z.B auf Gerätetausch... gerät selektieren rechte Maustaste und dann WEBUI Programme von diesem gerät.
Funktion ist unspektakulär kann die WebUI auch (habs nur programmiert, weil ein Device keine idarray hat, wo die verwendeten Programme drinstehen)
Diese gefundenen Programme kann man nun selektieren, und backup machen.

Im Editor bearbeitet lassen sich nun z.b. die Kanalbezeichner ändern (auch HM auf HMIP z.B.).

dann die gefundenen Programme löschen und die neuen Restoren.. Gerätetausch von HM-nach HMIP (Das neue Gerät mit den Kanälen MUSS natürlich vorhanden sein und die Datenpunkte auch den gleichen Namen (z.B. level) und Typ haben..)

natürlich müssen die neuen Kanal Bezeichner richtig eingegeben worden sein, sonst gibt das Shit in - Shit out Prinzip



7.2.6 WebUi Backup von selektiertem Device

Bekannter weise erfreut uns EQ3 ja ab und an mal mit der Notwendigkeit, ein Gerät mit Werksreset ab und dann wieder anlernen zu müssen. Leider wird beim Ablernen und Neuanlernen:

Master und Linksets werden hierbei nicht angepackt, dazu gibt’s die separaten Backupmöglichkeiten.

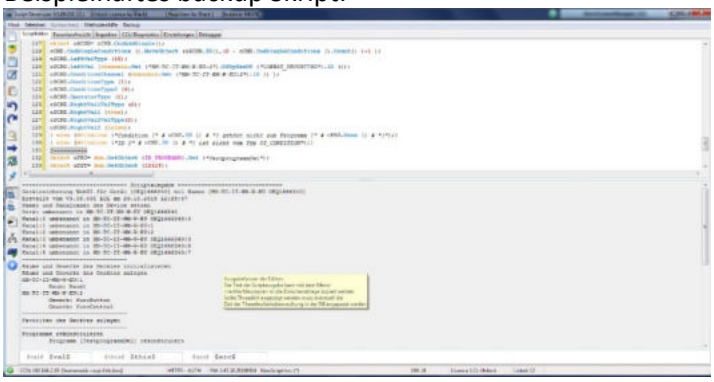
1. der Device und alle Kanalnamen auf Defaulteinstellung zurückgesetzt
2. Alle Gewerke Zuweisungen der Kanäle sind weg
3. Alle Raumzuweisungen der Kanäle sind weg
4. Alles Favoriten Zuweisungen von Kanälen sind weg
5. In Kanälen zugewiesene Systemvariablen sind weg
6. Programme, in denen das Device verwendet wurde, sind verstümmelt

Zerrupftes Programm nach Device löschen

Jedes Mal bedeute dieses eine WebUi Klicki Wiederherstellorgie.
Dies übernimmt nun der Menüpunkt WebUi Backup von Selektiertem Device:

- Vor dem Ablernen wird das Device selektiert und anschließend dieses Backup durchgeführt. Das Backup wird gespeichert, liegt aber auch im clipboard und kann direkt in den Editor eingefügt werden.
- Das Gerät nun mit Werksreset löschen
- Gerät wieder anlernen
- Das vorhin vom SDV automatisch erstellte restore Skript ausführen
- Fertig.

Beispielhaftes backup Skript:



Das Skript enthält automatisch generierten Code, nach dessen Durchlauf:

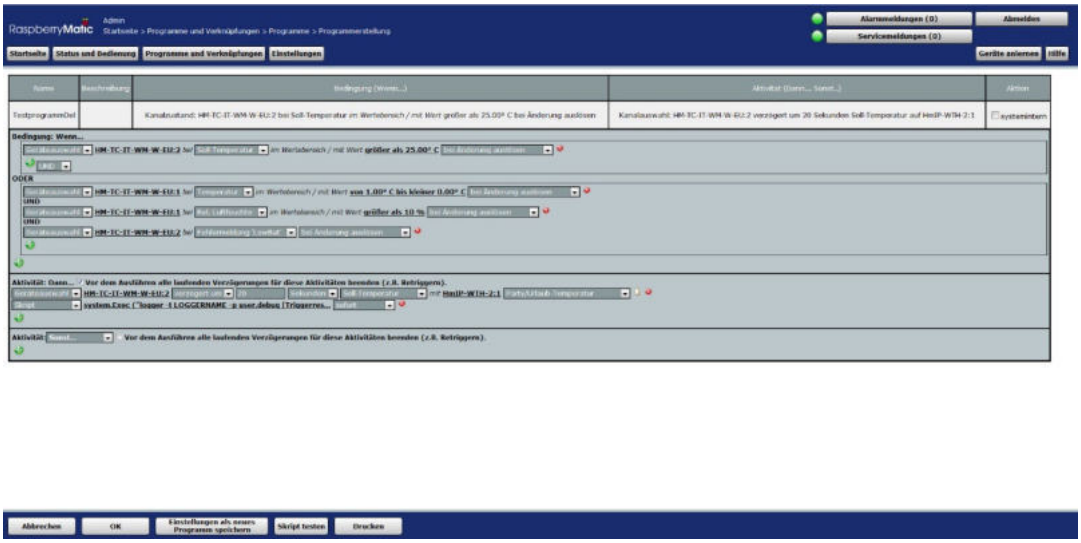
1. Stimmt der Gerätename wieder
2. Stimmen alle Kanalnamen wieder
3. Sind alle Kanäle wieder den ursprünglichen Gewerken zugeordnet
4. Sind alle Räume wieder den ursprünglichen Gewerken zugeordnet
5. Sind alle Kanäle wieder in den ursprünglichen Favoriten und an der richtigen Position
6. Systemvariablen, die vorher Kanälen zugewiesen waren, sind es nun auch wieder
7. Alle Programme, in denen Datenpunkte des Device verwendet wurden, sind wiederhergestellt und die Bedingungen auch an der richtigen Position

Alles, was mit der WebUI zu tun hatte, ist nach dem Durchlauf wieder korrigiert. . (Zum Einsatz kommen hier auch Mechanismen aus dem schon lange laufenden Programme-Backup)

Auch das Programm schaut anschließend wieder so aus wie vor dem Löschen des Gerätes

Ansonsten ist der Identifiziermechanismus
Seriennummer, wenn nicht gefunden --> Gerätename, wenn auch nicht gefunden Error

Wiederhergestelltes WebUI programm



Fortgeschrittene können mittels des restore Skriptes auch innerhalb der WebUi verschiedene Geräte tauschen, wobei da natürlich die Struktur der Geräte schon stimmig sein muss.
Erfolgreich wurde dieses im Forum schon praktiziert, dabei muss

1. Die Geräteseriennummer innerhalb des Skriptes getauscht werden
2. Die Kanal Struktur überprüft und gegebenenfalls händisch angepasst werden
3. Die Datenpunkte müssen zum Kanal passen.

Ist dieses erfüllt, ließen sich Tauschoperationen auch zwischen identischen, bei entsprechender Anpassung sogar auch zwischen ähnlichen HmIP Geräten durchführen.

7.2.7 Komplett Backup von Device

Mittels Komplett Backup von Device kann von den Mastersets / Linksets ein Komplettes Backup hergestellt werden. Dieser Punkt ist sehr hilfreich, um nach einem Ablernen mit Werksreset / Wiederanlernen alles Geräteeinstellungen sowie alle Direktverknüpfungen wiederherstellen zu können

Von den markierten Devices werden die BackUP Jsons angelegt. Dieses kann dann ganz normal in den SDV Editor geladen werden und dort ausgeführt werden.

Dieses backup ist auch in der Lage, ein Gerät auf einem frischen System wiederherzustellen, nicht nur auf dem Ursprungssystem.

Ein restore macht:

1. Gerätenamen wiederherstellen
2. Alle Channamen wiederherstellen
3. Alle Eigenschaften ChanArchive wiederherstellen
4. Systemvariablen, die in Kanälen zugewiesen wurden, werden angelegt, wenn sie sich noch nicht auf dem System befinden
5. Systemvariablen werden den Kanälen zugewiesen
6. Verwendetet Räume und Gewerke werden angelegt, wenn sie sich noch nicht auf dem System befinden
7. Räume und Gewerke werden den Kanälen zugewiesen
8. Alle Mastersets der Geräte werden wiederhergestellt
9. Abweichende Metadaten werden wiederhergestellt
10. Sonderobjekte werden angelegt wenn benötigt, oder verwiesen, wenn schon vorhanden
11. Alle bestehenden Direktverbindungen des Gerätes werden gelöscht
12. Alle Direktverbindungen aus dem Backup werden erzeugt, benannt, und parametiert.

Die Ausführung läuft als 3 Pass innerhalb des Editors und kann einige zeit dauern:
Beispiel eines restore eines einfachen 1 Kanal Aktors:

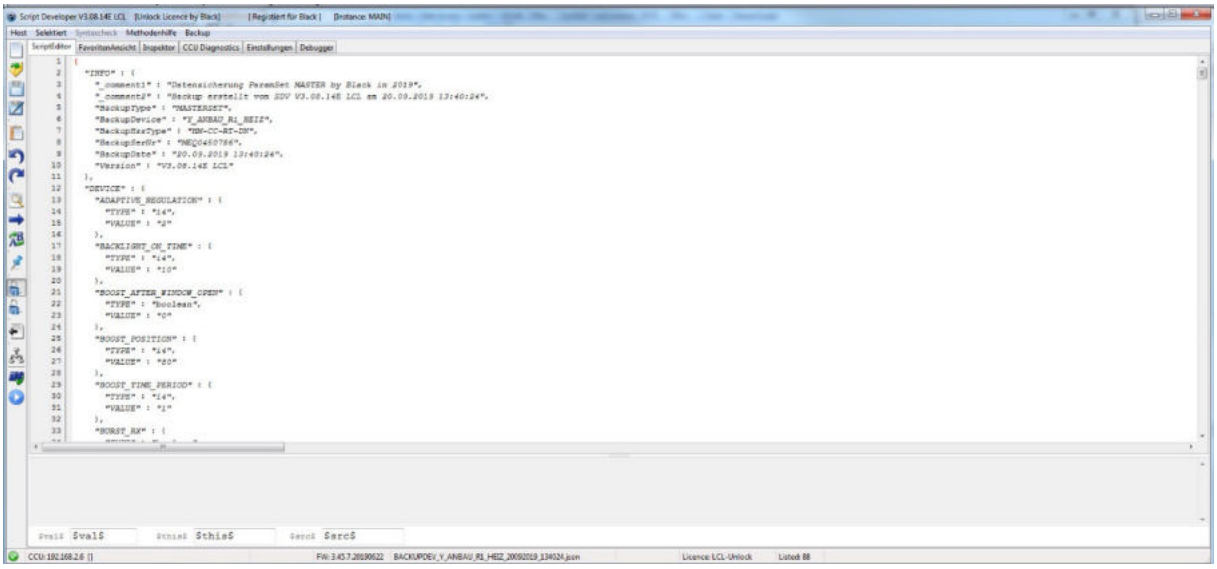
```
JSON Restore Complete Device
ThreadkillTimeout eingestellt auf 20000 ms
Complete Device Backup (PASS 1 von 3) by Black in 2020
Backup erstellt vom SDV V4.01.08J LCL am 24.07.2020 15:42:33
GeräteName  : Aktor1
GeräteType  : HM-LC-Sw1-DR
Seriennummer: PEQ0480086
------------------------------------------------------------
Device umbenannt in "Aktor1"
Überprüfung Vorhandensein Räume/Gewerke/Systemvariablen
Kanal PEQ0480086:0 umbenannt in "Aktor1:0". Zuweisungen erfolgt
Kanal PEQ0480086:1 umbenannt in "Aktor1:1". Zuweisungen erfolgt
------------------------------ Ausführung -------------------------------
15:53:14:788 Sende Script an CCU [TX 3445 Bytes]
15:53:15:141 Empfange Daten von CCU [RX 741 Bytes]
15:53:15:172 CCU Laufzeit 0.343 sec

Complete Device Backup (PASS 2 von 3) by Black in 2020
Datensicherung ParamSet MASTER by Black in 2020
Backup erstellt vom SDV V4.01.08J LCL am 24.07.2020 15:42:33
GeräteName  : Aktor1
GeräteType  : HM-LC-Sw1-DR
ChannelNames: NO
SerNummer   : PEQ0480086
------------------------------------------------------------
Paramset Device  [Aktor1] - Ergebnis: 0: OK
Paramset Channel [Aktor1:1] - Ergebnis: 0: OK
Wiederhestellung Metadaten und Property Objects
------------------------------ Ausführung -------------------------------
15:53:15:172 Sende Script an CCU [TX 4395 Bytes]
15:53:17:857 Empfange Daten von CCU [RX 1755 Bytes]
15:53:17:857 CCU Laufzeit 2.687 sec
```

```
Complete Device Backup (PASS 3 von 3) by Black in 2020
Datensicherung ParamSet LINKS by Black in 2020
Backup erstellt vom SDV V4.01.08J LCL am 24.07.2020 15:42:33
GeräteName  : Aktor1
GeräteType  : HM-LC-Sw1-DR
Seriennummer: PEQ0480086
Gelöscht wird : [BidCoS-RF:1]      --> [PEQ0480086:1]
Gelöscht wird : [LEQ1252736:1]     --> [PEQ0480086:1]
Wird nicht gelöscht : [PEQ0480086:1]       --> [PEQ0480086:1] Grund: Flag=1
Das Gerät enthielt 3 Direktverknüpfungen
2 dieser Verbindungen wurden gelöscht
1 dieser Verbindungen referenzieren intern (Flags) und wurden nicht gelöscht
-------------------------
1. Direktverknüpfung
-------------------------
Channel Empfänger: Aktor1:1
HssType Empfänger: SWITCH
Adress Empfänger : PEQ0480086:1
Channel Sender   : HM-RCV-50 BidCoS-RF:1
HssType Sender   : VIRTUAL_KEY
Adress Sender    : BidCoS-RF:1
DV (PEQ0480086:1 <<- BidCoS-RF:1) angelegt [HM-RCV-50<> BidCoS-RF:1 mit Aktor1:1 : Standardverknüpfung
Virtuelle Fernbedienung  - Schaltaktor]
DV (PEQ0480086:1 <<- BidCoS-RF:1) Parametersatz geladen - 0: OK
-------------------------
2. Direktverknüpfung
-------------------------
Channel Empfänger: Aktor1:1
HssType Empfänger: SWITCH
Adress Empfänger : PEQ0480086:1
Channel Sender   : HM-Sec-RHS LEQ1252736:1
HssType Sender   : ROTARY_HANDLE_SENSOR
Adress Sender    : LEQ1252736:1
DV (PEQ0480086:1 <<- LEQ1252736:1) angelegt [HM-Sec-RHS LEQ1252736:1 mit Aktor1:1{ : Standardverknüpfung
Fenster-Drehgriffkontakt  - Schaltaktor]
DV (PEQ0480086:1 <<- LEQ1252736:1) Parametersatz geladen - 0: OK
-------------------------
3. Direktverknüpfung
-------------------------
Channel Empfänger: Aktor1:1
HssType Empfänger: SWITCH
Adress Empfänger : PEQ0480086:1
Channel Sender   : Aktor1:1
HssType Sender   : SWITCH
Adress Sender    : PEQ0480086:1
DV (PEQ0480086:1 <<- PEQ0480086:1) war angelegt, Beschreibung angepasst [name 33 : ÄÖÜ äöüßaa]
DV (PEQ0480086:1 <<- PEQ0480086:1) Parametersatz geladen - 0: OK
Korrektur LINKCOUNT des Gerätes
-------
Folgende LinkCounts wurden angepasst:
------------------------------ Ausführung ------------------------------
15:53:17:872 Sende Script an CCU [TX 14113 Bytes]
15:53:21:061 Empfange Daten von CCU [RX 9955 Bytes]
15:53:21:076 CCU Laufzeit 3.187 sec
-------
Fertig
Ausführungszeit 6.313 sec
```

7.2.8 Backup Masterset

Es ist möglich, von einem Gerät (oder von mehreren Selektierten) ein Backup zu erstellen. Dazu werden das oder die Geräte Selektiert, dann Backup – Paramset Master von Device

Dabei wird von allen selektieren Geräten eine JSON Datei von den Geräteeinstellungen angelegt. Dieses JSON kann in den Editor geladen und dann ausgeführt werden.



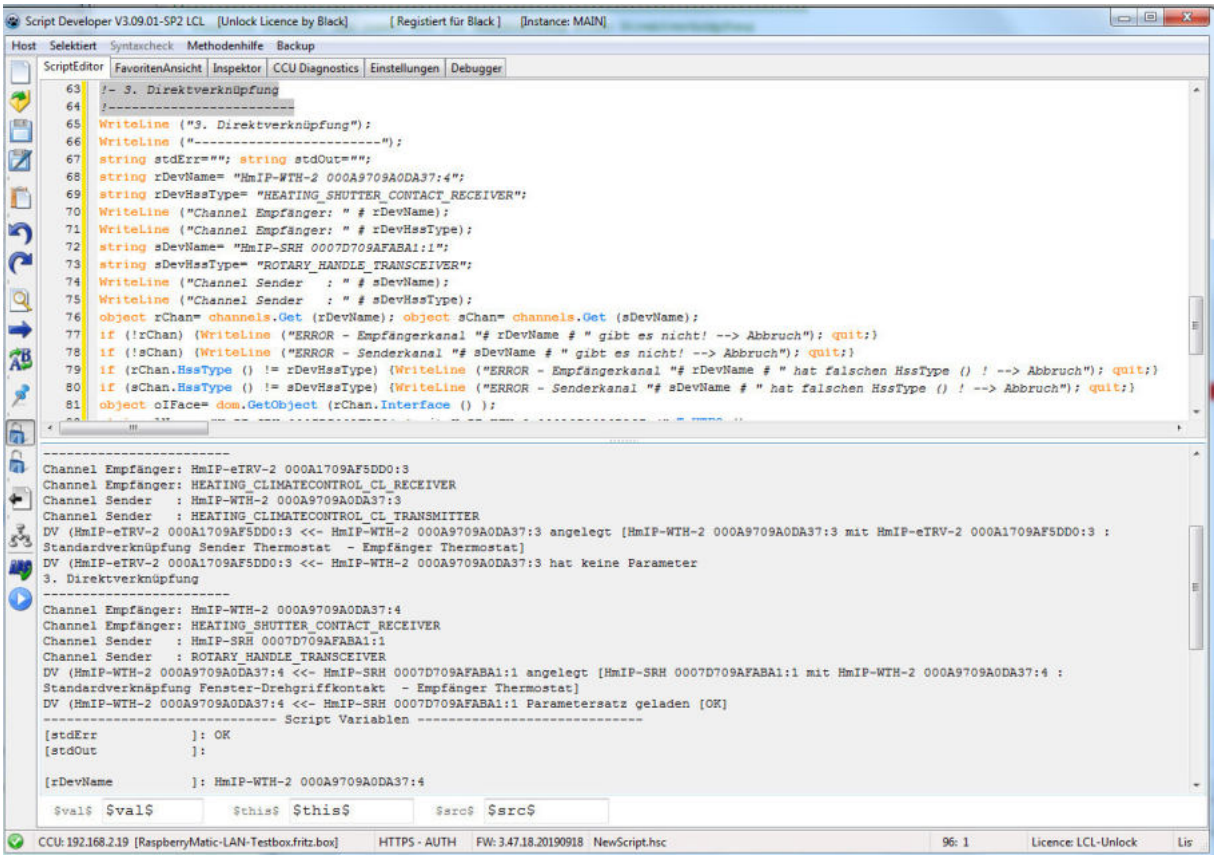
Ein Masterset Backup berücksichtigt auch:
Abweichende Metadaten
Sonderobjekte

Dabei werden die Einstellungen des JSON an das Gerät übertragen. Wenn man weiss, was man tut, kann man einzelne Werte auch manuell verändern.

7.2.9 Backup Linkset

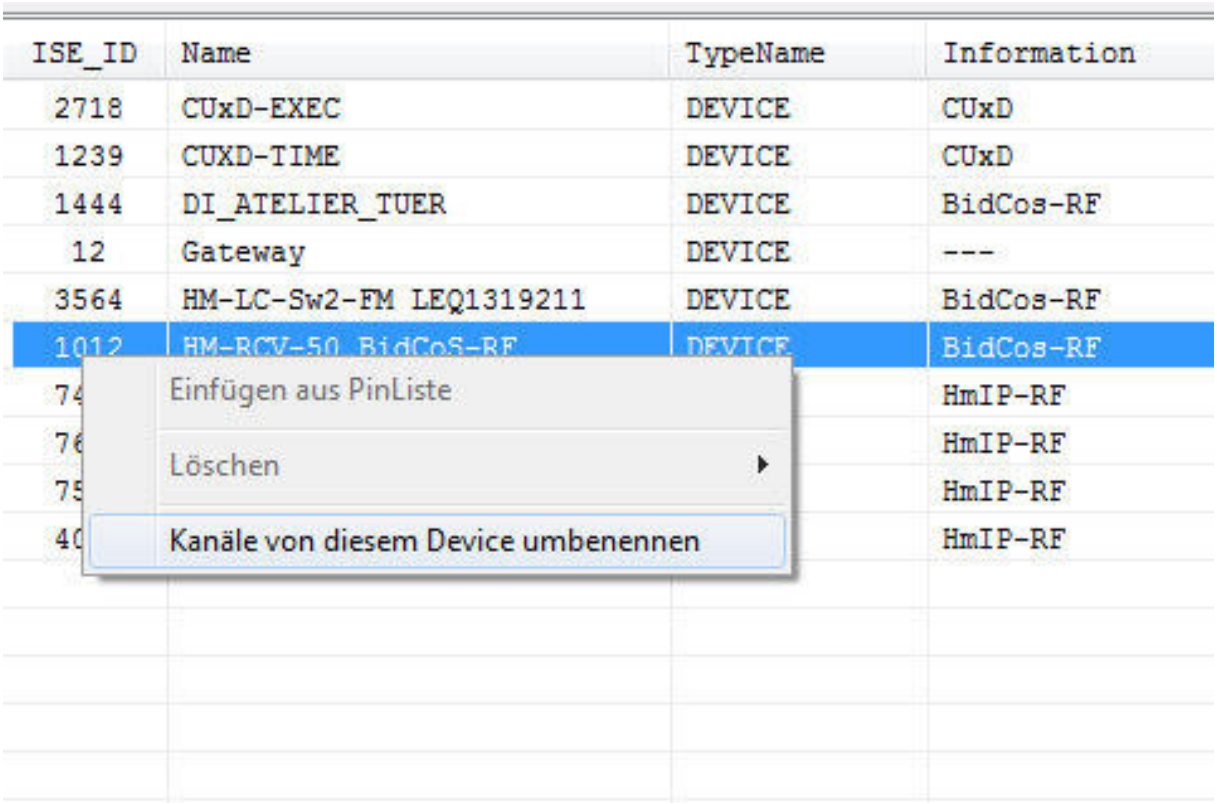
Es ist möglich, von einem Gerät (oder von mehreren Selektierten) ein Backup der zu dem Gerät definierten Direktverknüpfungen zu erstellen. Dazu werden das oder die Geräte Selektiert, dann Backup – Paramset Link von Device

Dabei wird von allen selektieren Geräten eine JSON Datei von den zu den Geräten definierten Direktverknüpfungen angelegt. Dieses JSON kann in den Editor geladen und dann ausgeführt werden. Dabei werden die Direktverknüpfungen angelegt über AddLink und anschließend die Werte der Expertenparameter, so vorhanden, geladen



7.3 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 Namen der Device geändert.
Anschliessend rechte maustaste auf das Device in der Listendarstellung und Punkt auswählen:



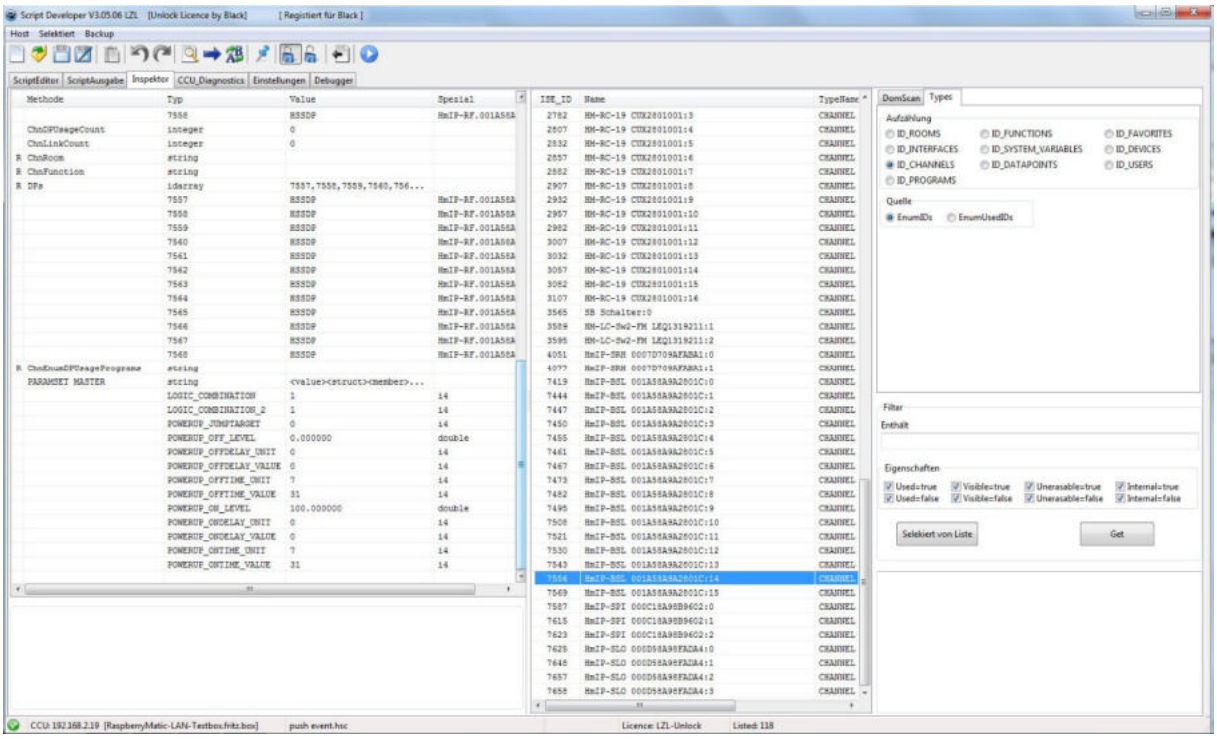
Rückfrage mit Ja bestätigen und die Kanäle werden so benannt:

Device: DeviceName

Kanal0 : DeviceName:0
Kanal1 : Devicename:1
Etc…

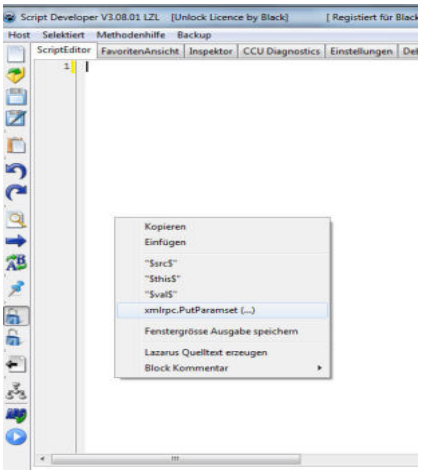
7.4 Paramset Master

Bei Device, Kanälen, die einen Paramset Master haben, wird dieser mit angezeigt (wenn in der Sicht konfiguriert und mind. Level 5)

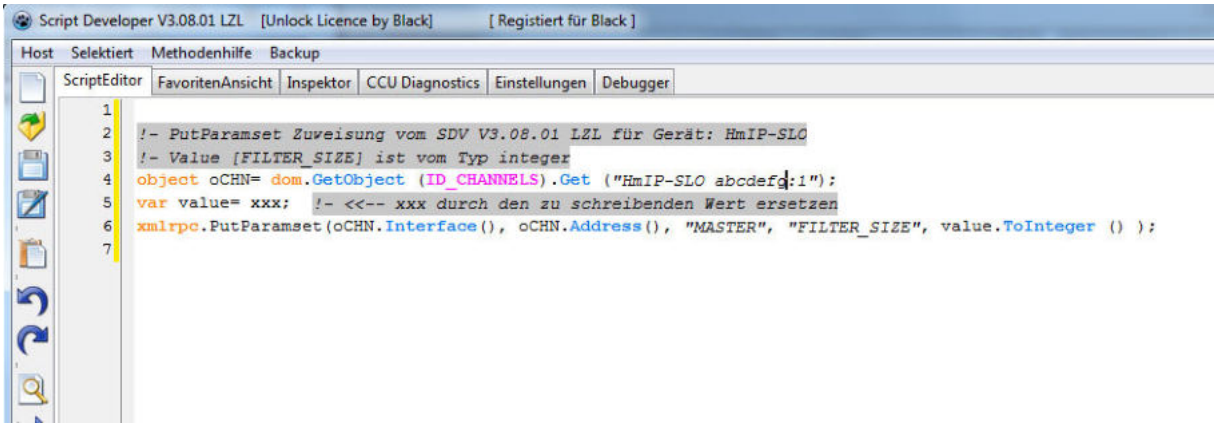


Die Werte lassen sich dann über PutParamset verändern in einem Skript

Ab Version V3.08.01. Einzelne Parameter des Mastersatzes lassen sich nun auch markieren in der Detailansicht. Im Editor lässt sich nun Automatisiert der Code für die Manipulation dieses Masterparameters erzeugen.

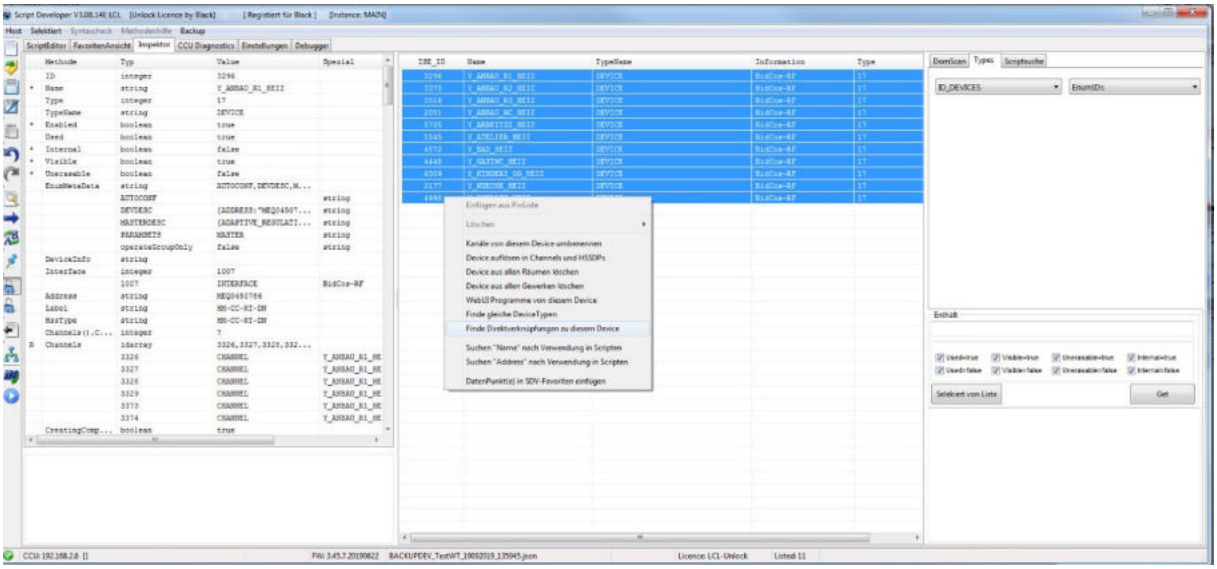


Dies erzeugt nun automatisiert den richtigen Code, in Abhängigkeit von: welches Gerät ist es (HMIP, HM, CUXD o.ä) und dem erwarteten Datentyp.

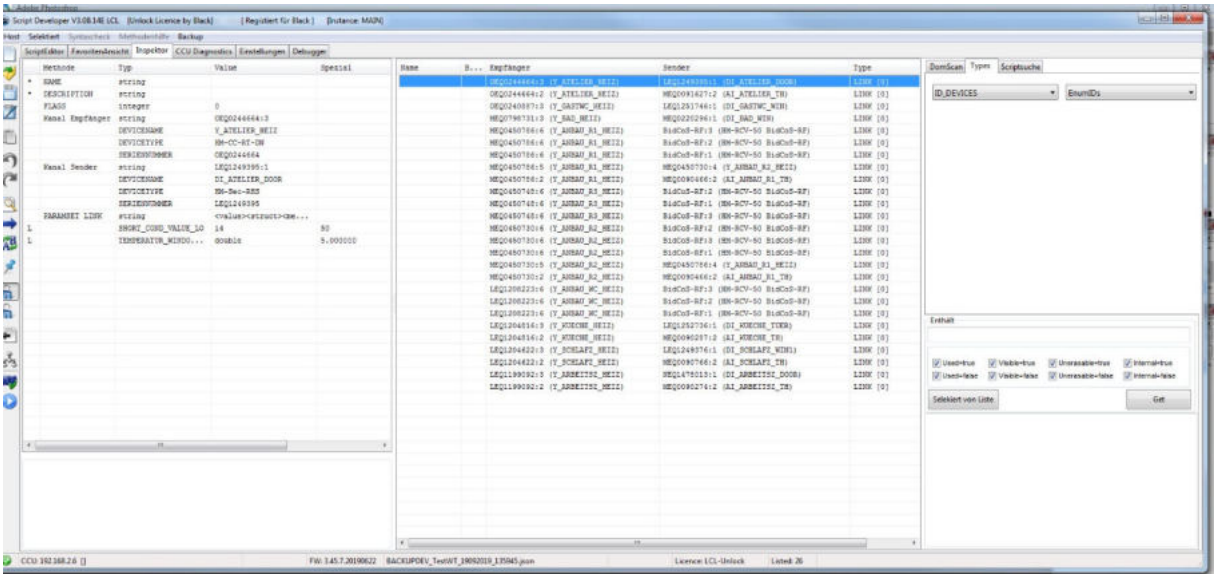


7.5 Linkset eines Gerätes

Ist ein Device markiert, so liegt auf der rechten Maustaste in der Listenübersicht der Menüpunkt „Finde Direktverknüpfungen zu diesem Device“



Die markierte Liste wird abgearbeitet, ist das markierte Element ein Device, so wird dazu versucht, die zugehörigen Linksets aufzulösen.



Die Linksets können angewählt werden, Sortiert werden nach Name, Beschreibung, Empfänger und Sender. Gleichzeitig werden die Expertenparameter, so vorhanden unter Paramset Link dargestellt.

Diese können auch markiert werden und im Editor kann für diese Parameter automatisch der Code zum Beschreiben generiert werden.

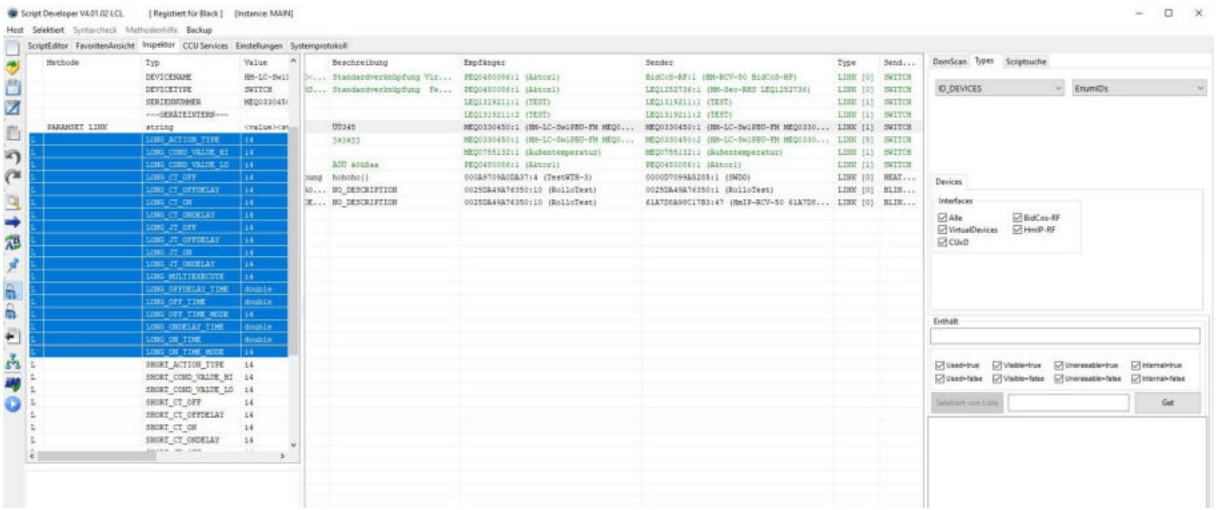
7.5.1 Kopieren von Teilen eines Linksets einer DV in eine andere DV

Über die Möglichkeiten der WebUI was das Handling von DV angeht sowie dem Datenaustausch zwischen DVs muss ich ja nicht viel erzählen, ist ein Trauerspiel. Aus dem grunde wollte ich mich da auch immer mal drangesetzt haben, wenn man in DV1 etwas geändert hat, dass dieses schnell und ohne clickiclackiOrgien in gleichartigen anderen DVs zu ändern ist.

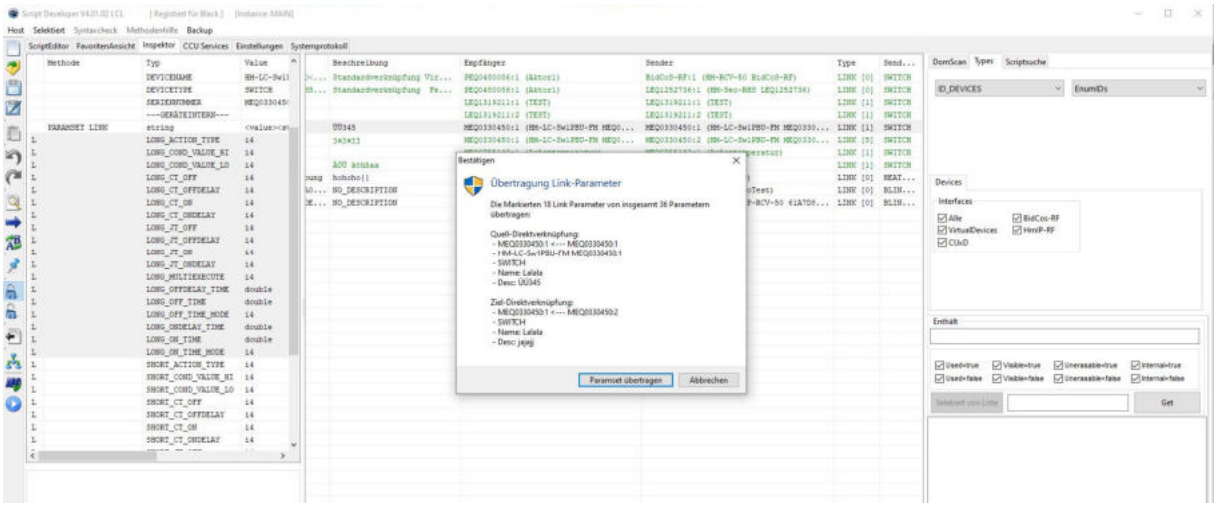
Das nötige Handling von Direktverknüpfungsdaten passte nicht so richtig in die Struktur der Gerätekopien unter CCU Services.
AUs diesem Grunde das Handling bei Direktverknüpfungen im Inspektor um die Drag-Drop Events erweitert.
Damit ist es nun möglich, sich eine bestimmte Direktverknüpfung in der Detailansicht darstellen zu lassen, dann markiert man z.B. alle Parameter, die mit SHORT_ beginnen (alles für den kurzen Tastendruck) und zieht, wie in Windows üblich, dieses in die Listenansicht. Mit Beginn des Ziehens werden die anderen Direktverknüpfungen, deren Zielkanaleigenschaften zu dieser an der Maus hängenden Direktverknüpfung passen, grün dargestellt. Auf diesen kann dann die Maus gestellt werden und wie üblich gedropt werden. Es öffnet sich dann ein Abfragemenü, ob wirklich die Parameter von der Quelle Q ins Ziel Z kopiert werden soll, und bei Bestätigung nach Ausführung noch ein Statusdialog.

Beispielhaftes Vorgehen:

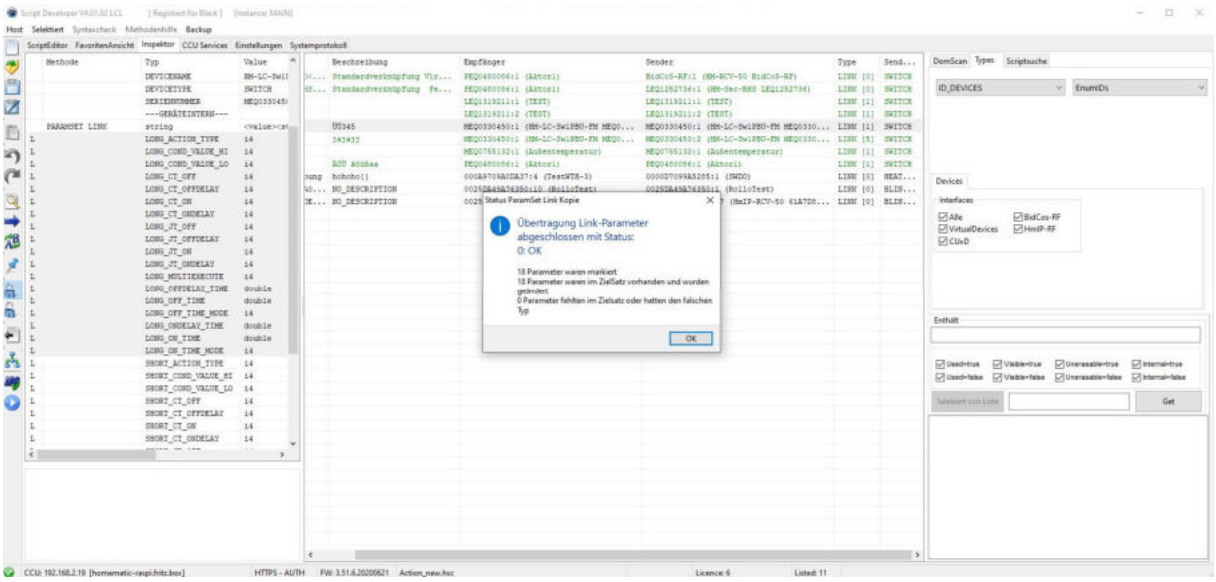
Auswahl einer DV und markieren eines Teiles des Parameter (LONG)



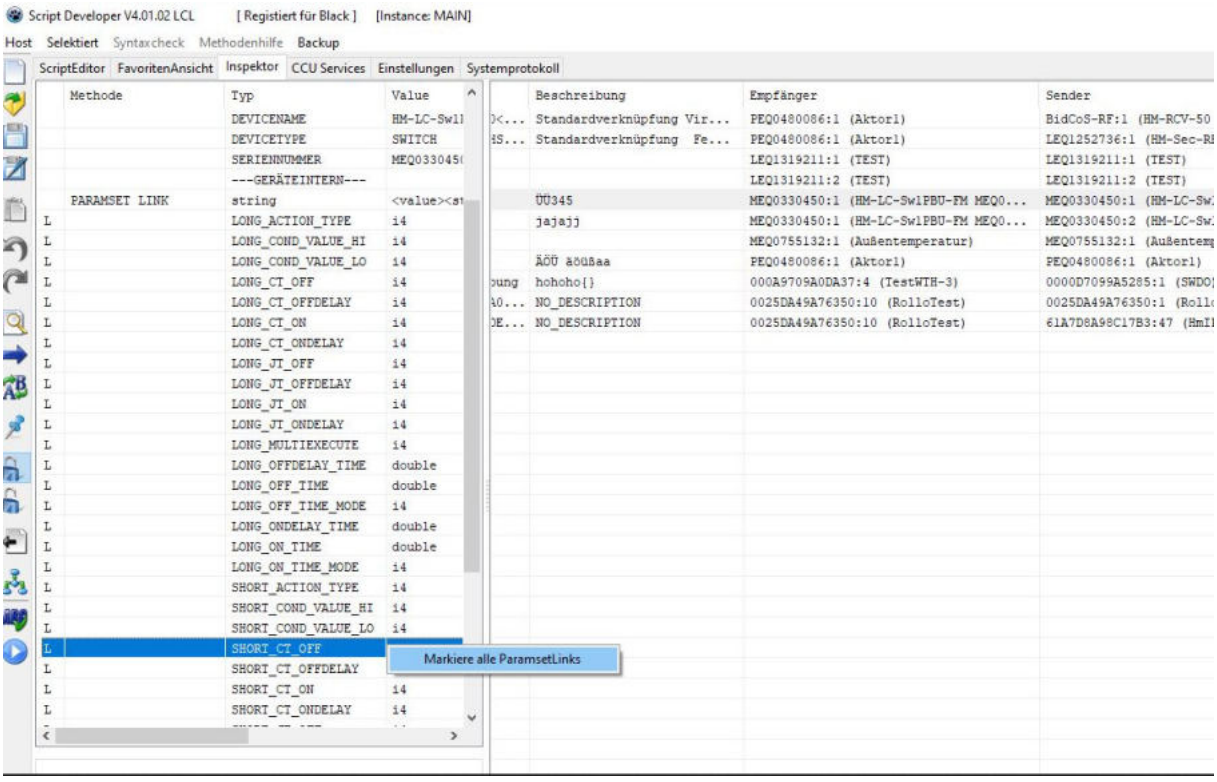
Ziehen von der Detailansicht in die Listenansicht. die möglichen ZielDVs sind nun grün geworden (Die nicht grün Eingefärbte mit der Switch Elgenschaft ist die QuellDV selber, nur ich steh nicht so auf buntes Farbenspiel, drum habe ich diese schwarz gelassen)



Auswahl einer ZielDV und Drop



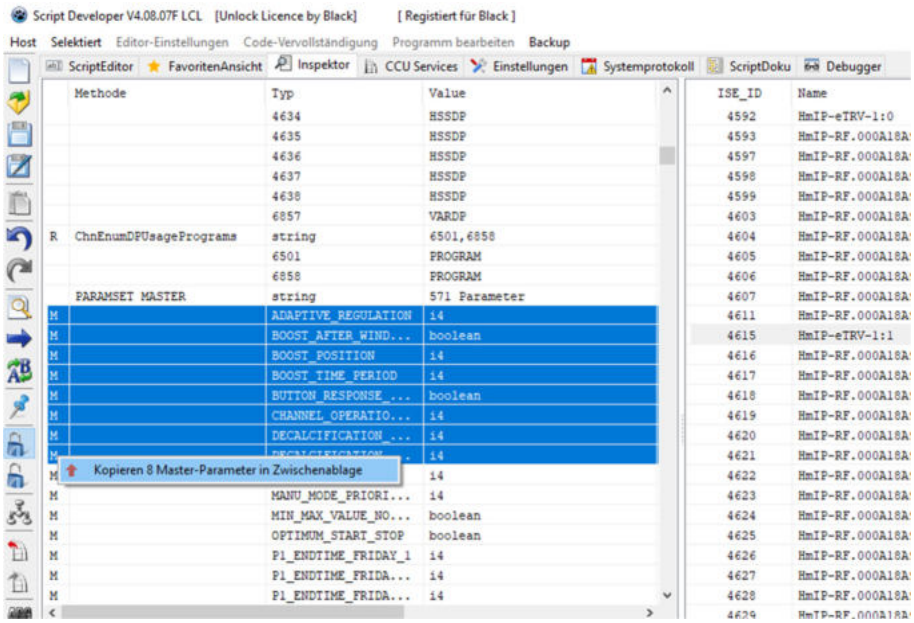
Nun nochmal kontrolle und bestätigen für Übertragen oder Abbruch.
Statusmessage nach dem Übertragen



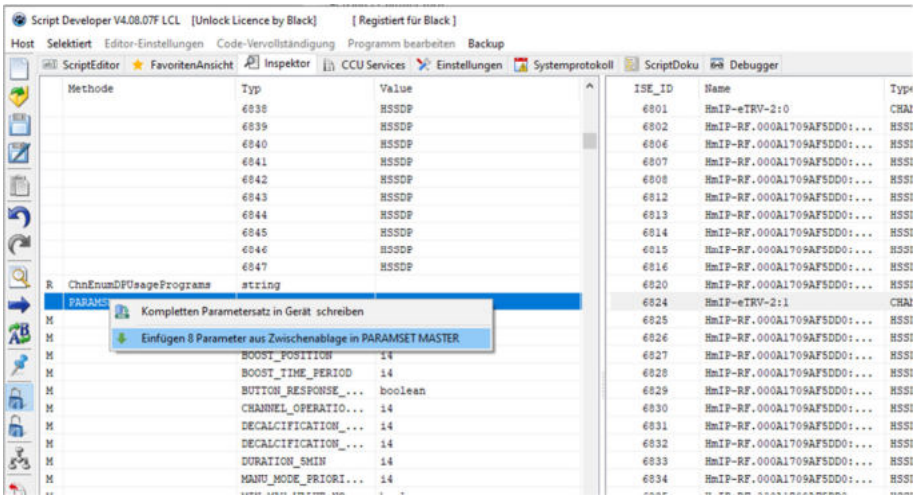
Sollen Alle DV Parameter ausgewählt werden, rechte Maustaste im Detailsview und Markiere alle ParamsetLinks

7.5.2 Einzelne Parameter markieren und in Link oder Masterparameter einfügen

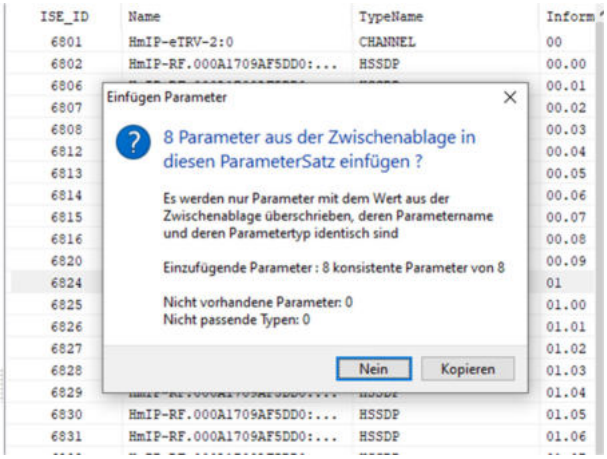
Vorgehensweise: Paramater (link oder Master markieren, rechte Maustaste und



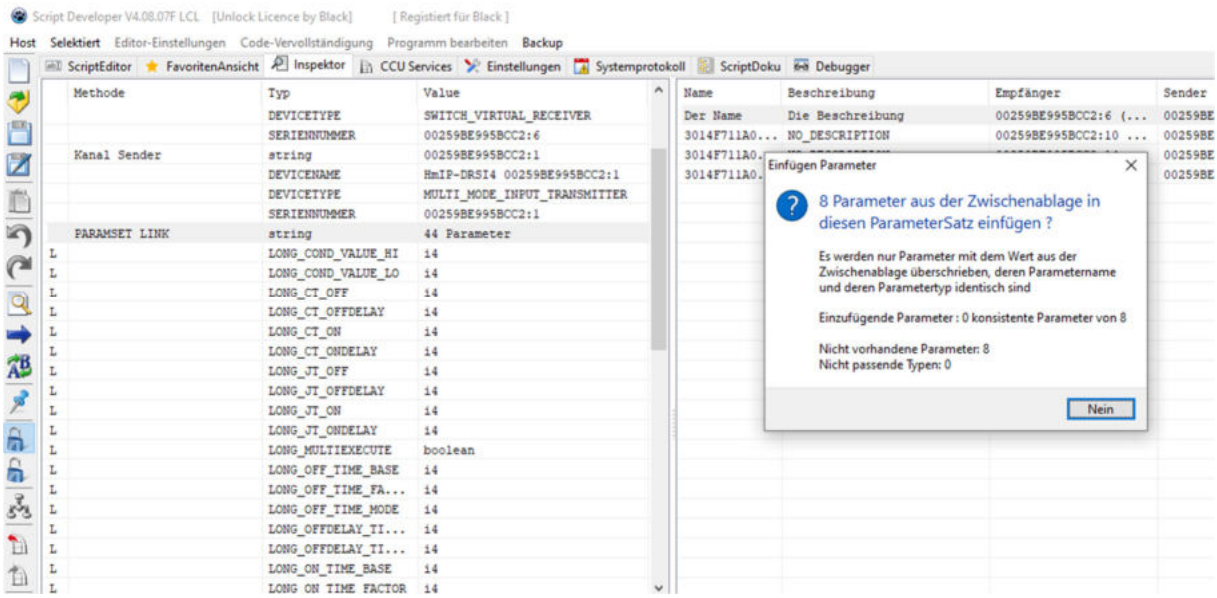
dann das Zieldevice oder Zielchannel auswählen (oder auch Direktverknüpfung) und dort rechte Maustaste auf Paramset Master bzw Paramset Link:



Es erfolgt noch eine SIcherheitsabfrage, wo auch drauf hingewiesen wird, welche Parameter im Ziel nicht vorhanden sind oder welche vorhanden sind aber mit abweichendem Typ: diese werden natürlich nicht mitkopiert.

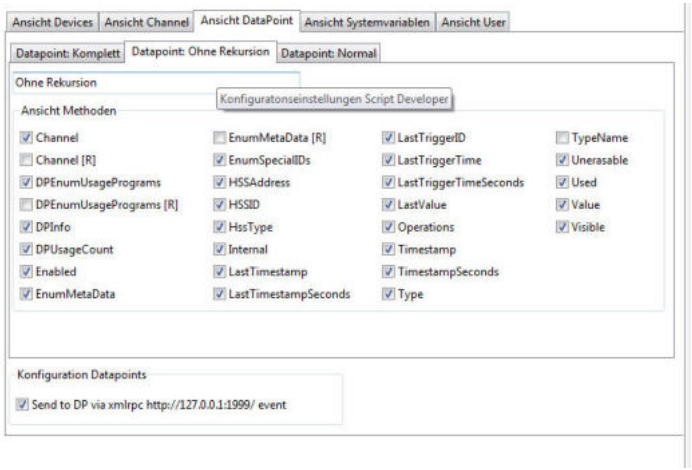


gibt es keine stimmige Übereinstimmung (in dem Beispiel markierte Masterparameter und versuch diese in eine DV einzufügen) und es keine Übereinstimmung gibt, so wird die Kopeiren Option auch gar nicht angeboten.



7.6 Rega Push auf Datenpunkte via Rega event

Rega Push: Damit lassen sich Datenpunkte innerhalb der Rega verändern, die sich normalerweise nicht verändern lassen. Z.B. Batteriefehler eines Netzaktors ^^. Setzt voraus, dass dieser Haken im der Konfiguration Datenpunkte gesetzt wurde:



Des weiteren braucht es in der Sicht Freigabe auf TypeName (SDV prüft auf HSSDP) und anklicken von Value. (Level 5 braucht es dafür auch)

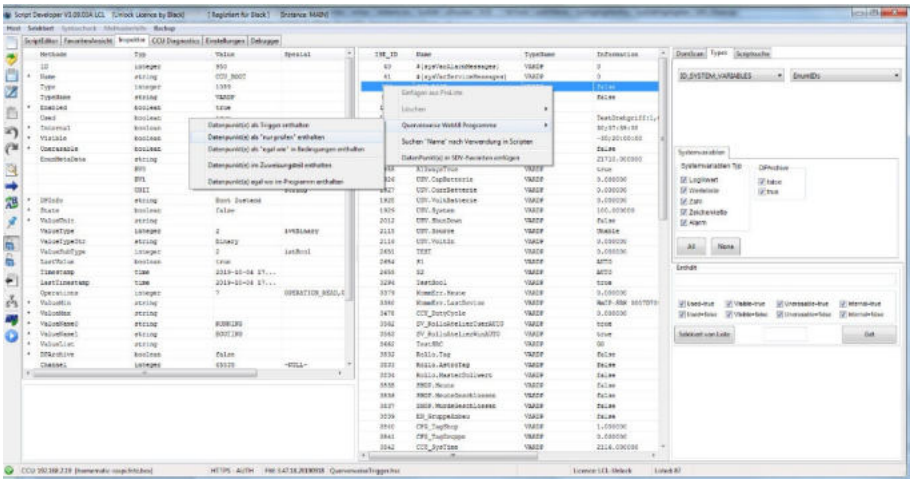
Hat der DP die Eigenschaft Write, wird ein ganz normales State (xxx) ausgeführt. Hat er diese nicht, macht der SDV bei Wertänderung von sich aus ein Rega Event auf den DP.

(Wenn die Freigabebedingungen passen)

Dieses Feature ist extrem nützlich aus eigener Erfahrung zum Testen von Programmen, wenn man z.b. Testtrigger und ähnliches von Sensoren prüfen will. Man muss hier nicht z.B mit dem Fön auf einen Thermostaten blasen , damit dieser über 35 Grad geht, einfach den DP anklicken, Value auswählen und die Zahl ändern. Wert übernehmen. (s.a. Kapitel 7.1)

7.7 Querverweise zu Datenpunkten in WebUI-Programmen

Es ist möglich, zu HSSDP, VarDP und AlarmDPs selektiv die Verwendung in WEBUI Programmen zu suchen.



Als Trigger: der Datenpunkt muss im Bedingungsteil als „bei Änderung“ oder „bei Aktualisierung“ verwendet werden.

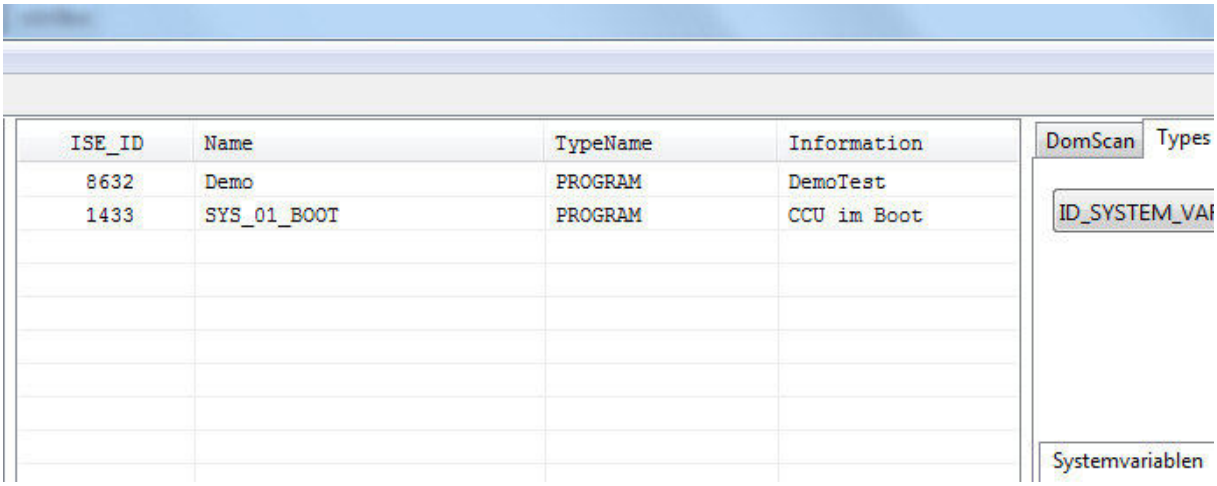
Nur prüfen: der Datenpunkt muss im Bedingungsteil als „nur prüfen“ verwendet werden

Als Egal wie in Bedingungen: der Datenpunkt muss im Bedingungsteil vorkommen

Im Zuweisungsteil: Der Datenpunkt wird im Dann oder Sonst Teil verwendet

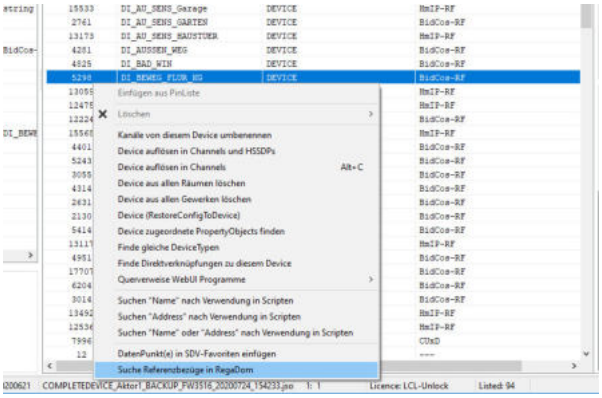
Egal wo: Der Datenpunkt muss nur irgendwo im WebUI Programm vorkommen

Als Ergebnis erscheinen in der Listenauswahl die Programme, wo der selektierte Datenpunkt unter der angewählten Bedingung vorkommt.

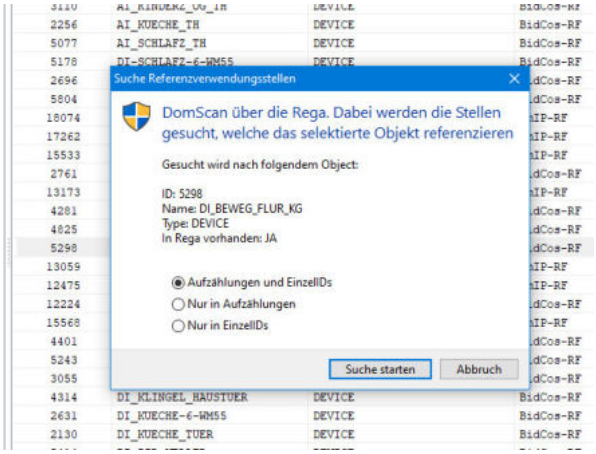


7.8 Suche Referenzbezüge in Regadom

Ein Typischer usecase bei Geisterobjekten oder falschen Bezügen ist Möglichkeit, alle Objekte finden zu können, die irgendwie auf dieses Objekt referenzieren.
Beispielsweise:
In welchen Objekten wird überall auf ein bestimmtes Objekt bezogen, hier mal in dem Beispiel, wo überall wird auf das selektierte Device referenziert.

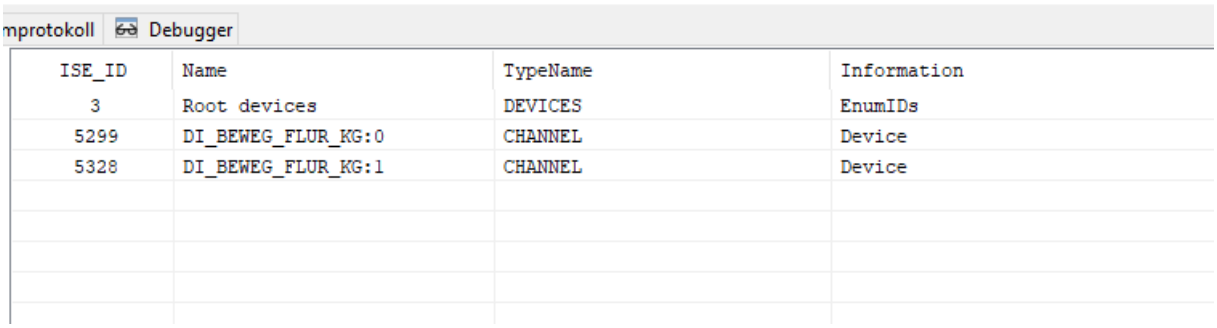


Dann erfolgt Rückfrage, wo überall gesucht werden soll:



Aufzählungen: Es wird nur in den Typischen Aufzählungen gesucht.
EinzelIDs: Es wird nur in referenzen gesucht, zb.b LeftVal in Singleconditions oder Channel in Datenpunkten
Aufzählungen und EinzelIDs: Es wird versucht die komplette rega durchzuscannen nach irgendwelchen Referenzen auf diese ID (hier 5298, das gewählte Device)

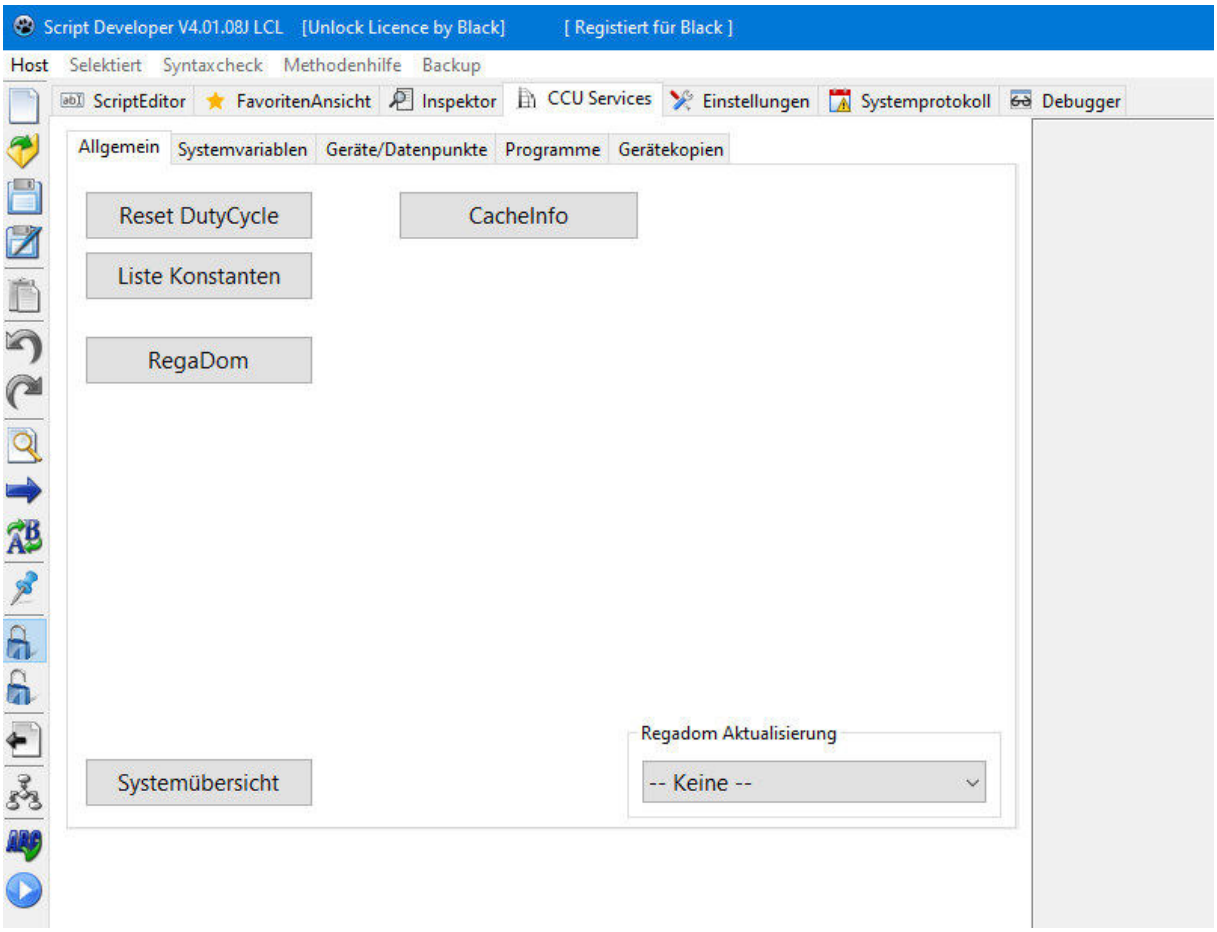
Als Ergebnis bekommt man in der Listenansicht dann eine Liste aller Datenpunkte, die irgendwie auf diese ID referenzieren.


mprotokoll Debugger

ISE_ID	Name	TypeName	Information
3	Root devices	DEVICES	EnumIDs
5299	DI_BEWEG_FLUR_KG:0	CHANNEL	Device
5328	DI_BEWEG_FLUR_KG:1	CHANNEL	Device

Hier in dem Fall ist alles normal, bei einem Geister Objekt muss nun alles Gefundene durchgegangen und mit menschlicher Intelligenz analysiert werden.

8 Diagnosebild



Bei den meisten Korrekturläufen werden folgende Aktionen vorgeschlagen:
Nur prüfen: Der SDV macht nichts ausser einer Prüfung und Ausgabe der Ergebnisse im Ausgabefeld.

Prüfen und Ergebnis in den Inspektor: Auch hier nur eine Prüfung, aber gefundene Objekte werden zur weiteren Untersuchung in die Detailansicht des Inspektors geladen.

Prüfen und korrigieren: Bei Fehlern, die der SDV auch korrigieren kann, werden diese wie unter nur prüfen im Ausgabefeld dargestellt, gleichzeitig versucht der SDV eine Reparatur

8.1 Allgemein

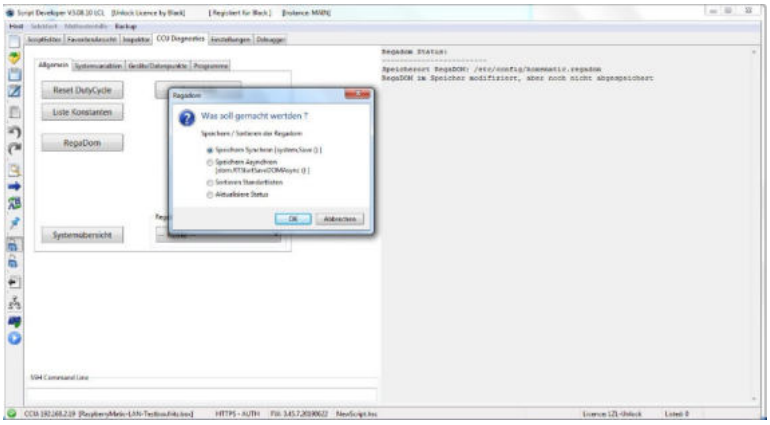
Reiter für Allgemeine Einstellungen

8.1.1 Liste Konstanten

Alle Systemkonstanten der Rega werden dargestellt

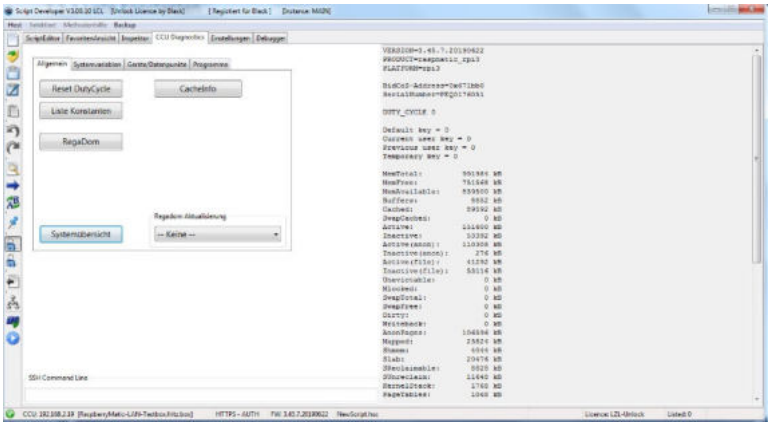
8.1.2 Regadom

Status der Rega abrufen und Speichern anstossen



8.1.3 Systemübersicht

Allgemeine Information über die CCU

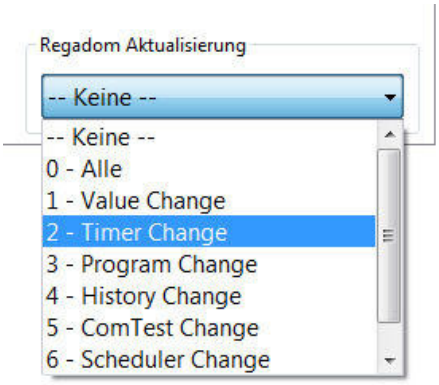


8.1.4 CacheInfo

Darstellung des CacheInhaltes der CCU

8.1.5 Regadom Aktualisierung

Anstossen von dom.RTUpdate (). Auswahl für die ComboBox



Hier auch Schnellstatus Programme mit gesetzter Copy ID. Ist nach Neustart zentrale immer noch der Wert <>0… Geisterprogramm

8.1 SSH Funktionalität

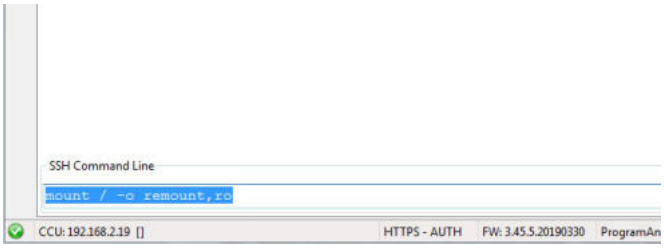
Wenn auf der CCU / Raspberrymatik SSH freigegeben ist, ist es möglich, ab Level 5 eine rudimentäre SSH Funktionalität zu nutzen.

Was ist dazu nötig ?
In der SDV Ini müssen die nötigen Schlüssel eingegeben sein.

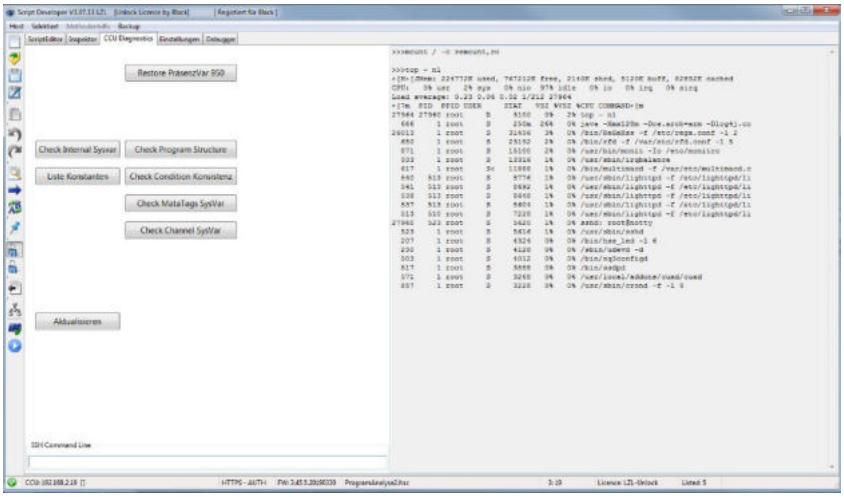
```
[CCU1]
IP=192.168.2.xx
USERNAME=xxxx
PASSWORD=yyyy
USEHTTPS=true
SSHUSERNAME=root       Bestandsnutzer: diese schlüssel hinzufügen
SSHPW=sshpasswort

[CCU2]
IP=192.168.2.xx
USERNAME=
PASSWORD=
USEHTTPS=false
SSHUSERNAME=root
SSHPW=

[HOSTCCU]
IP=192.168.2.xx
USERNAME=
PASSWORD=
USEHTTPS=False
SSHUSERNAME=root
SSHPW=
```

Ab der 3.08.xx kommt der SDV wieder ohne das PLink aus, hier wird dies nun über CUxD realisiert.

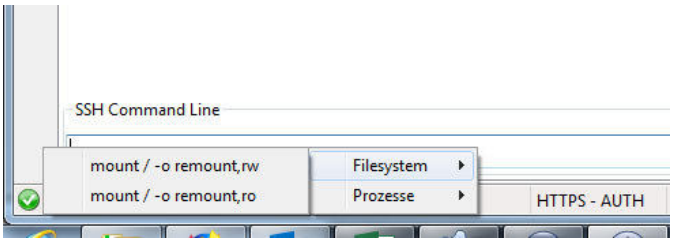
In dieser Zeile lassen sich nun einfache SSH Kommandos ausführen.



Dies soll und ist aber kein Ersatz für eine putty Konsole, eher gedacht für ich muss mal eben schnell einen Befehl ausführen oder man hat nicht die Befehlsfolgen im Kopf sondern benutzt die Hilfe. Programme dort zu starten oder z.b. top zusehen geht nicht und ist auch nicht geplant, das kann putty ganz klar besser. zweimal Rad erfinden ist auch nicht nötig.



Einige Funktionen sind als Hilfe mit rechter Maustaste auf die Kommandozeile programmiert, wird bei Bedarf von mir noch erweitert, Vorschläge dazu sind auch willkommen.



Programme dort zu starten oder z.b. top zusehen geht nicht und ist auch nicht geplant, das kann putty ganz klar besser. zweimal Rad erfinden ist auch nicht nötig. Also eine Ausführung von top rennt nach 10 Sekunden in den Thread Timeout Kill.

```
>>>top
ThreadKilled
```

8.1.1 SSH Realisierung (CUxD ab 2.3.1 oder plink.exe)

INI Schlüssel SSHPLINK=false

bei false versucht er den Zugriff über die CUXD Seite. Wenn der Zugriff nicht Freigeschaltet ist über

USERACCESS=1+
USERLOGIN=

Wichtig ist das + !!!!!!!!!!

rennt er dabei gegen Access denied.

Das Kleingedruckte: USERACCESS=1+ ist Sicherheit wie alte Version, also keine.

Wer das nicht möchte, kann den damals beschriebenen alten Weg gehen, putty suite runterladen, plink exe ins SDV Verzeichnis kopieren, cmd unter Windows öffnen und einmal im SDV Verzeichnis ausführen:

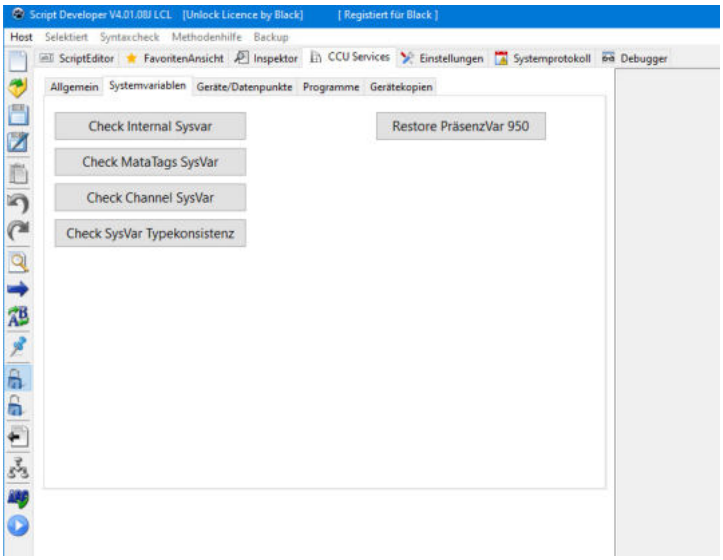
plink -ssh root@IPderCCU -pw SSHPASSWORT top - n1

Wenn es der erste Aufruf war, die Frage nach dem Zertifikat bejahen. danach geht der Aufruf auch headless mit dem SDV.

damit ist aber wieder Windows im Spiel.

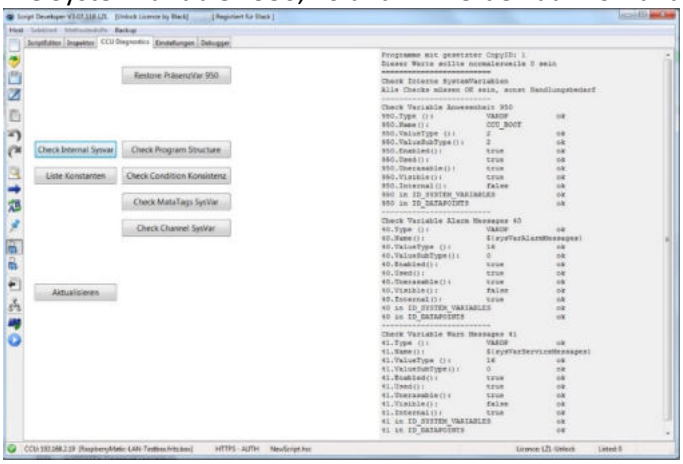
Wer weder CuXD den Access setzen möchte noch plink benutzen will, der kann den SDV nutzen bis auf die SSH Fnktionalität.

8.2 Diagnosen Systemvariablen



8.2.2 Check Internal Sysvar

Die Systemvariablen 950, 40 und 41 werden auf Vorhandensein und richtige Parameter überprüft.

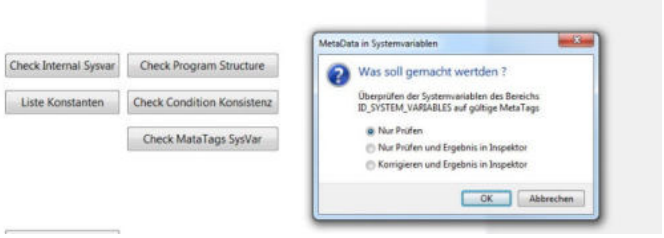


8.2.3 Check metaTags Sysvar

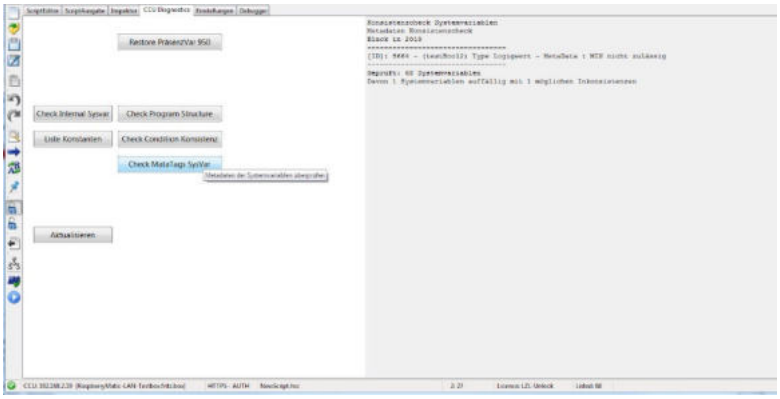
Systemvariablen werden auf korrekte Metadaten hin überprüft. Der SDV nimmt von der Prüfung aber selbstdefinierte Metatags, welche mit _sdv_ beginnen, aus.

Dieser Menüpunkt überprüft die Systemdaten auf Konsistenz von dem Typ, welcher durch ValueType und ValueSubtype spezifiziert ist und den Einträgen unter EnumMetadata.

Druck erzeugt folgendes Menü:

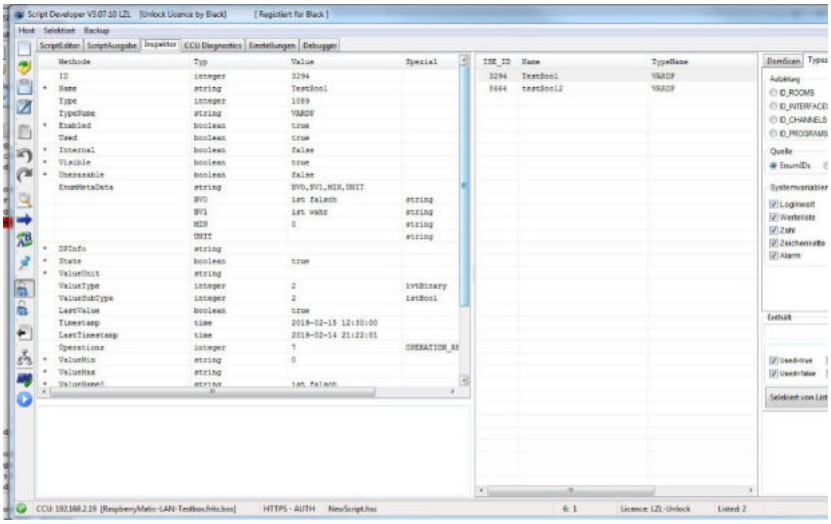


Punkt 1: nur Prüfen
Bringt eine Ausgabe in der Form, wie das Skript, welches ich im Forum Online gestellt hatte. Mehr nicht. Er tut selber nix.



Punkt 2: Nur Prüfen und Ergebnis in Explorer

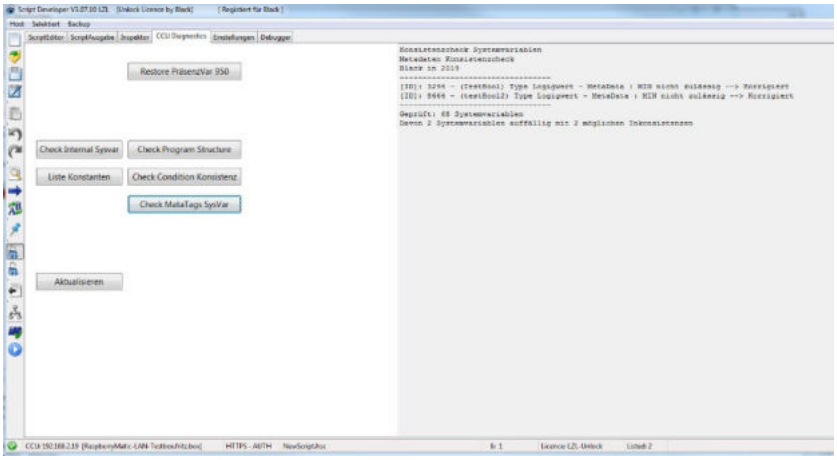
Genau wie Punkt 1, es wird eine Überprüfung gemacht, zusätzlich wird das Ergebnis noch in die Auswahlliste des Inspektors geladen. Dort kann überprüft und gegenbenenfalls auch manuell korrigiert werden.



Punkt 3: Korrigieren und Ergebnis in den Inspektor

Hier wird automatisch korrigiert und das Ergebnis dann in den Inspektor geladen.

VORHER AUF JEDENFALL EIN BACKUP MACHEN. Es gibt kein Undo hierbei.

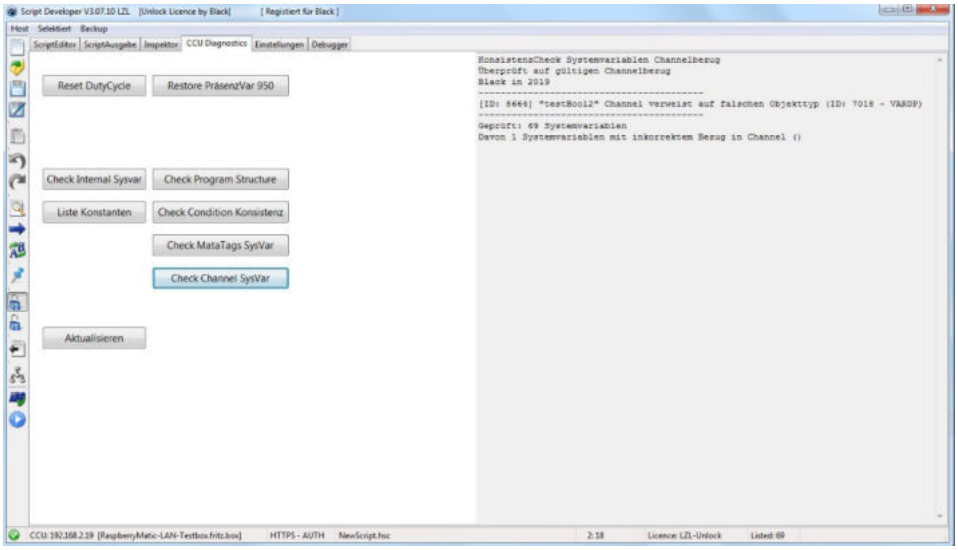


Ein nochmaliger Durchlauf bringt dann die Fehlerfrei-Nachricht.

```
Konsistenzcheck Systemvariablen
Metadaten Konsistenzcheck
Black in 2019
---------------------------------
---------------------------------
Geprüft: 68 Systemvariablen
Keine Auffälligkeiten festgestellt
```

8.2.3 Check Channel Sysvar

Prüf und oprionaler Korrekturlauf ob eine Systemvariable mit einem Eintrag Channel auf wirklich unter DPs() des jeweiligen Channels eingehangen ist.



8.2.4 Check Sysvar Typkonsistenz

Es wird überprüft, ob Systemvariablen einen der erlaubten Typen haben.
Überprüft die Typekonsistenz von Systemvariablen:

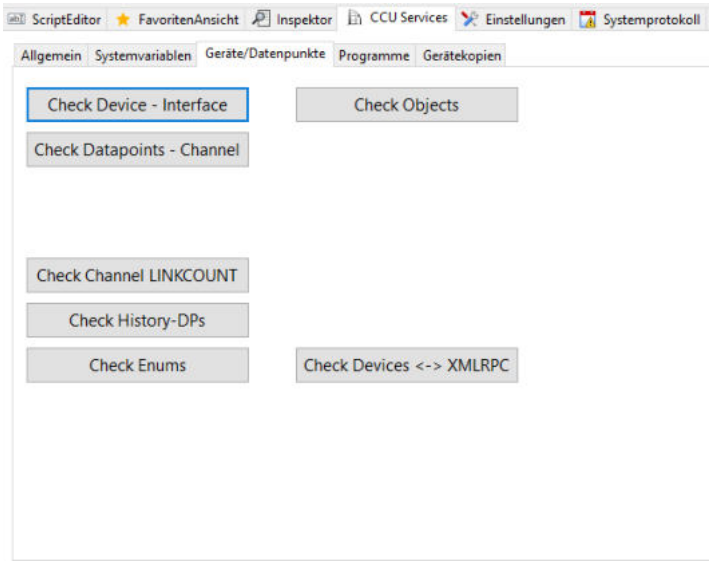
OT_ VARDP: Logikwert: Type() und ValueSubType
OT_ VARDP: Zahl: Type() und ValueSubType
OT_ VARDP: String: Type() und ValueSubType
OT_ VARDP: WerteListe: Type() und ValueSubType
OT_ VARDP: WerteListe: Vorhandensein von ValueList

OT_ALARMDP: Type() und ValueSubType

8.2.5 Restore Präsenzvar 950

Kann eine Abhanden gekommene Präsenz Systemvariable wieder rekonstruieren

8.3 Geräte Datenpunkte



8.3.1 Check Device-Interface

Analyse Interface Eintrag in Devices
Überprüfung aller Devices auf konsistenten Eintrag von der Interface ID in der Methode .Interface ().
Da Interface() readonly ist, kann der SDV einen eventuellen Fehler nicht reparieren.

8.3.2 Check Datapoints Channel

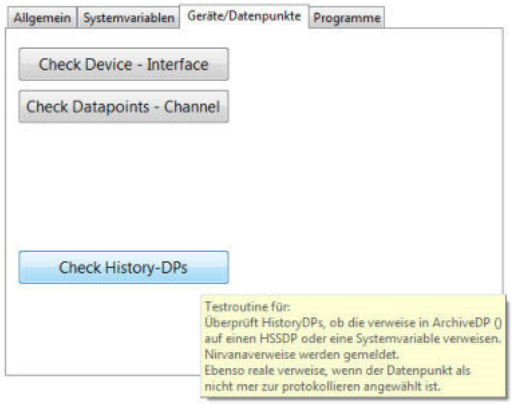
Testroutine für:
Überprüft HSSDPs, ob diese korrekt in datapoints
eingehangen sind, ob die Channelbezüge korrekt sind
und die HssAdress passt

8.3.3 Check Channel Linkcounts

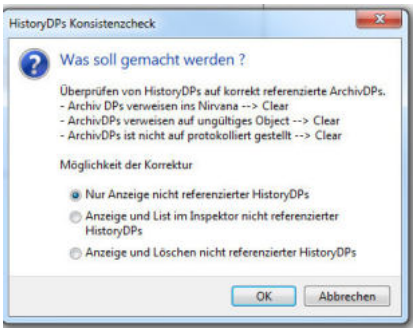
LINKCOUNT enthält die Anzahl an Direktverknüpfungen eines Kanales. Mit der Zeit kann dieser Wert inkonsistent werden. Diese Routine prüft und korrigiert auf Wunsch den LinkCount Eintrag in den Metadaten der Channels

8.3.4 Check History DPs

HistoryDPs sind eine wunderbare Gelegenheit um immensen Datenmüll in der RegaDOM zu hinterlassen. So wird beispielsweise für bei Anwahl eines Channels auf protokolliert für jeden einzelnen Datenpunkt des Channels ein HistoryDP angelegt. Wird das gerät irgendwann mal gelöscht oder die Aufzeichnung deaktivier, so bleibt – richtig gedacht – der HistoryDP erhalten.
Kann man so machen, muss man aber nicht.
Mit dem Punkt



Lassen sich verwaiste oder auch falsche HistoryDPs aufspüren. Die Art der Bearbeitung wird beim Anklicken festgelegt.

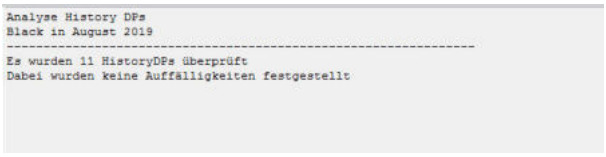


Nur Anzeige:
Listet die gefundenen „Inkonsistenzen“ rechts im Ausgabefenster auf. Es wird aber noch nix korrigiert oder verändert

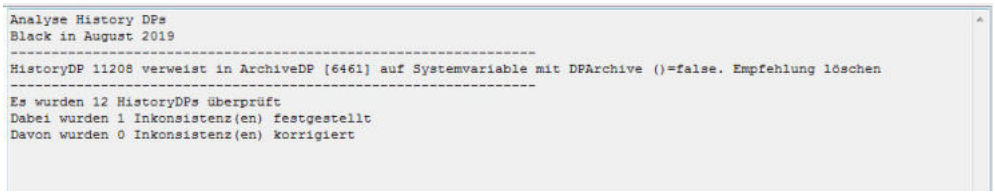
Anzeige und List im Inspektor
Listet die gefundenen „Inkonsistenzen“ rechts im Ausgabefenster auf. Zusätzlich werden die gefundenen IDs in Listauswahl des Inspektors geladen und können dort mit dem Inspektor untersucht werden. Es wird aber noch nix korrigiert oder verändert.

Anzeige und Löschen
Wie unter Punkt 1 werden die gefundenen Inkonsistenzen rechts im Ausgabefenster gelistet, gleichzeitig wird korrigiert (nicht referenzierte DPs werden gelöscht).

Lauf alles in Ordnung:



Ein HistoryDP verweist auf eine Systemvariable, bei der Protokollierung allerdings abgewählt wurde:



Wenn Korrektur angewählt würde, würde der HistoryDP gelöscht werden vom SDV.

8.3.5 Check Enums

Testroutine für:
Überprüft Enums auf folgende Konsistenz:
a:) EnumTypes ()
1. ID_ROOMS müssen Type etRoom
2. ID_FUNCTIONS müssen Type etFunction
3. ID_FAVORITE müssen Type etFavorite
haben. Korrektur ist möglich

b:) DomScan nach Enums gefundene Enums
1. etRoom müssen in ID_ROOMS
2. etFunction müssen in ID_FUNCTIONS
3. etFavorite müssen in ID_FAVORITE
gelistet sein

gleichzeitig wird noch nach etUnknown gesucht
Die Liste hierbei gefundener Inkonsistenter Enums wird in das Listenfeld des Inspektors geladen

8.3.6 Check Objects

Komplexe Testroutine für:
Überprüfung richtige Objekttypen in Aufzählungen
Gegenprüfung von beispielsweise Channel - Datenpunkt auf korrekten Verweis
Überprüfung richtiger Objekttyp von referenzieren Objekten
Überprüfung korrekter Address und HSSAddress bei Geräten

Es wird immer mit Test und Gegentest gearbeitet

Beispielhafter Testlauf:

```
Analyse Aufzählungstypen
Erstellt vom V4.01.08J LCL
Black in 2020
---------------------------------
Überprüfung 23 Objekte in Aufzählung : ID_DEVICES
Überprüfung 4 Objekte in Aufzählung : ID_INTERFACES
Überprüfung 16 Objekte in Aufzählung : ID_ROOMS
Überprüfung 10 Objekte in Aufzählung : ID_FUNCTIONS
Überprüfung 15 Objekte in Aufzählung : ID_FAVORITES
Überprüfung 247 Objekte in Aufzählung : ID_CHANNELS
Überprüfung 98 Objekte in Aufzählung : ID_SYSTEM_VARIABLES
Überprüfung 1615 Objekte in Aufzählung : ID_DATAPOINTS
Überprüfung 19 Objekte in Aufzählung : ID_PROGRAMS
Überprüfung 12 Objekte in Aufzählung : ID_CALENDARDPS
Überprüfung 98 Objekte in Aufzählung : ID_SERVICES
Überprüfung 10 Objekte in Aufzählung : ID_RULES
Überprüfung 38 Objekte in Aufzählung : ID_CONDITIONS
Überprüfung 125 Objekte in Aufzählung : ID_SCONDITIONS
Überprüfung 0 Objekte in Aufzählung : ID_DESTINATIONS
Überprüfung 27 Objekte in Aufzählung : ID_SDESTINATIONS
Überprüfung 8 Objekte in Aufzählung : ID_USERS
-----------------------------------------------
Ausgeführte Prüfungen: 12491
-----------------------------------------------
Step 2 - DomScan über die Rega mit Abgleich: Objekte in Listen
Scanberech RegaDOm IseID (1) bis IseID (65535)
         Object [19580,SDV_FAV] vom Type ENUM ist vom Typ etUnknown
         Object [19581,SDV_CHN] vom Type ENUM ist vom Typ etUnknown
         Object [19582,SDV_DP] vom Type ENUM ist vom Typ etUnknown
         Object [19658,SDV_FAV] vom Type ENUM ist vom Typ etUnknown
         Object [19659,SDV_CHN] vom Type ENUM ist vom Typ etUnknown
         Object [19660,SDV_DP] vom Type ENUM ist vom Typ etUnknown
         Object [19661,SDV_FAV] vom Type ENUM ist vom Typ etUnknown
         Object [19662,SDV_CHN] vom Type ENUM ist vom Typ etUnknown
         Object [19663,SDV_DP] vom Type ENUM ist vom Typ etUnknown
         Object [19728,SDV_FAV] vom Type ENUM ist vom Typ etUnknown
         Object [19729,SDV_CHN] vom Type ENUM ist vom Typ etUnknown
         Object [19730,SDV_DP] vom Type ENUM ist vom Typ etUnknown
         Object [19731,SDV_FAV] vom Type ENUM ist vom Typ etUnknown
         Object [19732,SDV_CHN] vom Type ENUM ist vom Typ etUnknown
         Object [19733,SDV_DP] vom Type ENUM ist vom Typ etUnknown
         Object [19734,SDV_FAV] vom Type ENUM ist vom Typ etUnknown
         Object [19735,SDV_CHN] vom Type ENUM ist vom Typ etUnknown
         Object [19736,SDV_DP] vom Type ENUM ist vom Typ etUnknown
-----------------------------------------------
In der RegaDom überprüfte Objekte  : 2532
Höchste verwendete ID in RegaDom   : 20270
GeisterObjekte ohne Listenzuordnung: 18
```

Hier finden sich Geister Testobjecte des Types OT_ENUM (ist das testsystem)

8.3.7 Check Devices < -- > XMLRPC

Folgendes wird überprüft:
Alle Geräte mit gültigem Interface müssen auch in ID-Devices gelistet sein.
Jedes dieser Geräte muss sich auch im Schnittstellenprozess wiederfinden.

Aufgelistet werden:
- Geräte, die Im Schnittstellenprozess vorhanden sind aber nicht in der Rega
- Geräte, die in der Rega vorhanden sind aber nicht im Schnittstellenprozess
- Geräte die nicht in ID-Devices auftauchen

Check Devices <--> xml neu eingeführt. Dieser Lauf führt einen Vergleich der rega gegen die Schnittstellenprozesse durch. Damit lässt dich auch ein gerät finden, dass nicht mehr in der rega ist aber noch in den Schnittstellenprozessen hängt. Eine Reparatur wird nicht automatisch gemacht, aber der weg vorgeschlagen (der beginnt immer mit einem VollBackup). Das vor einiger Zeit mal CuxD GeisterDevice liess sich damit auch finden.

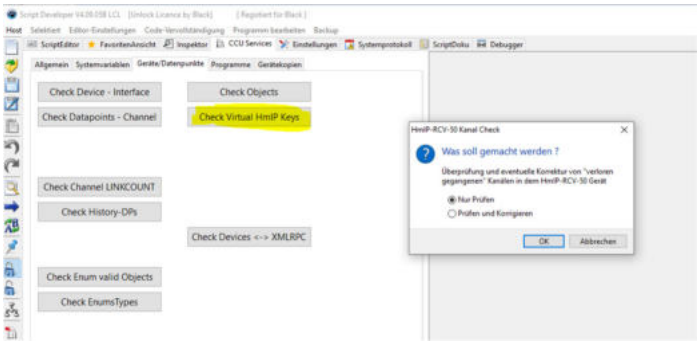
Fehlerfrei sollte so ein Ergebnis kommen:

```
Folgende Geräte wurden an den Schnittstellenprozessen gefunden:
        CUxD          : 2
        BidCos-RF     : 9
        BidCos-Wired  : 0
        HmIP-RF       : 9
        VirtualDevices: 2
Abfrage der Geräte aus der RegaHSS
RegaScan über Ise-Bereich von ID (1)  bis ID (65535)
Folgende Geräte wurden in der Rega gefunden:
        CUxD          : 2
        BidCos-RF     : 9
        BidCos-Wired  : 0
        HmIP-RF       : 9
        VirtualDevices: 2
Alle OT_DEVICE sind in ID_DEVICES gelistet
------------------------------------------------------------------
Es gibt keine Inkonsistenz zwischen RegaDom und den Schnittstellenprozessen
```

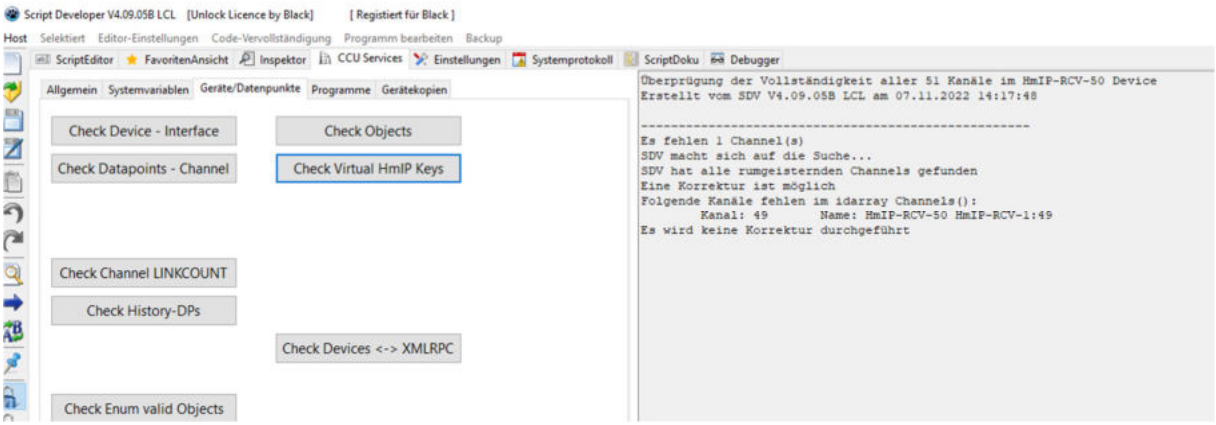
8.3.8 Check Virtual HmIP Keys.

Da ja "magisch verschwundene" Kanäle aus der virtuellen HmIP Fernbedienung kein Einzelfall sind, habe ich im SDV eine Möglichkeit programmiert, sas Gerät auf diesen Effekt zu prüfen und wenn das HmIP-RCV-50 wirklich einen Teil seiner Kanäle vergessen hat, dieses automatisiert wieder zu korrigieren.
Trotz allem natürlich: VORHER BACKUP MACHEN

Es besteht die Möglichkeit nur prüfen oder prüfen und korrigieren

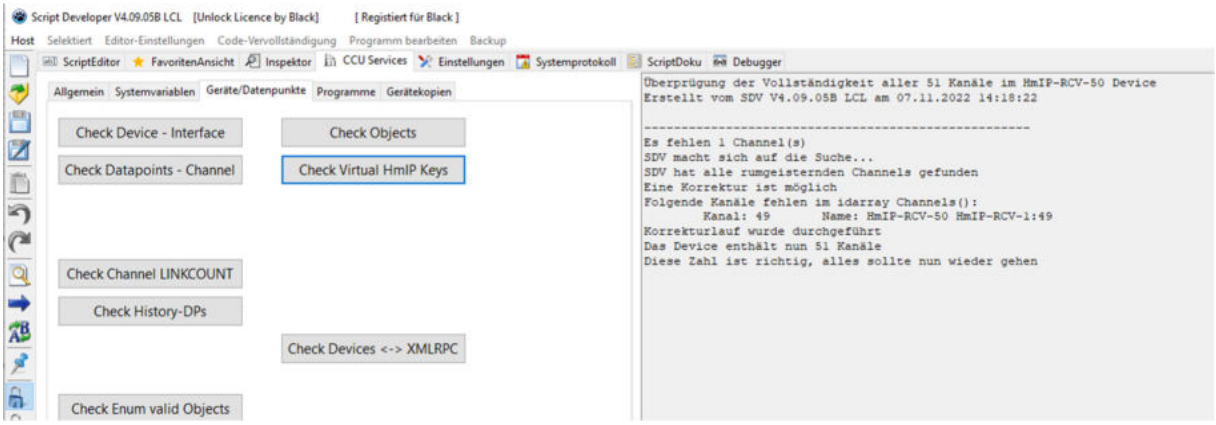


Nur Prüfen checkt nur, listet eventuell Fehler auf, korrigiert aber nichts
Gibt es einen Fehler, erscheint dieses:

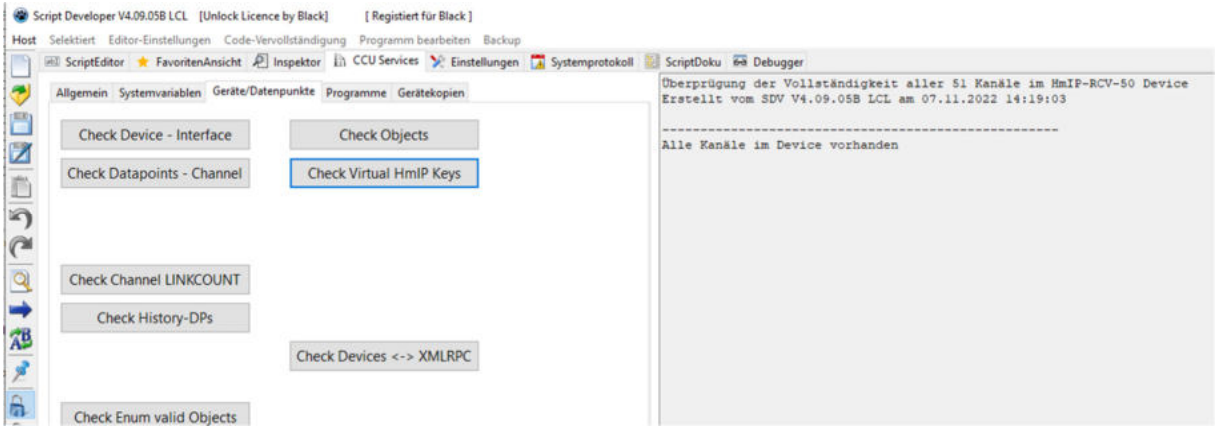


Der SDV prüft, ob alle benötigten Kanäle (er muss 51 finden) noch in der Rega rumgeistern, Reparaturen werden nur gemacht, wenn die Geister und Realobjekte vorhanden sind)

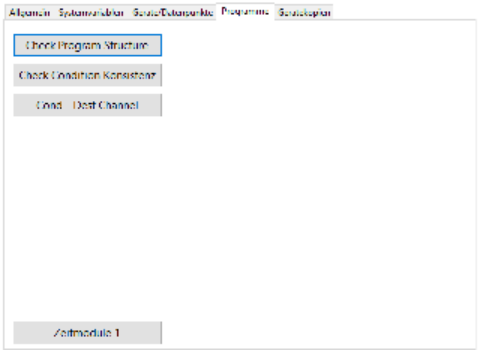
Eine Reparatur bastelt den verschwundenen Kanal wieder an die richtige Stelle zurück



Ein nochmaliger Lauf bringt dann die Meldung, keine Fehler mehr da

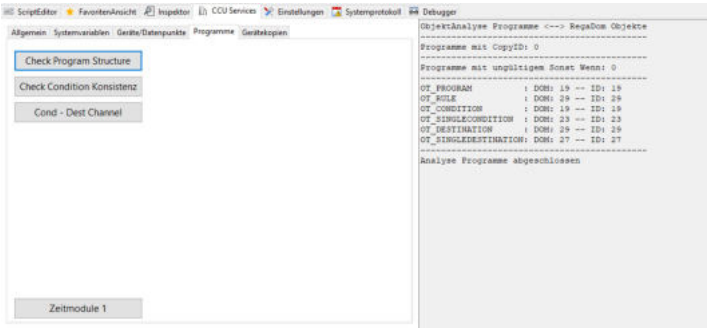


8.4 Programme



8.4.1 Check Program Structure

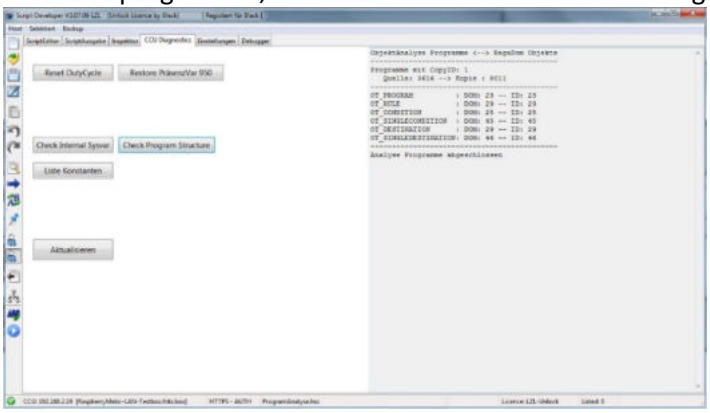
Mittels der Programmanalyse kann die Konsistenz der Programmobjekte überprüft werden.
So sieht der Aufruf auf, wenn die Prüfroutine erfolgreich über die Programme gelaufen ist.



Keine Einträge in ProgrammCopyID, die weiteren Tests ergeben eine Übereinstimmung zwischen dem DomScan und der Analyser der Programme

Mögliche Fehlerbilder:

1. Geisterprogramme, Leichen welche beim Editieren übrig geblieben sind.



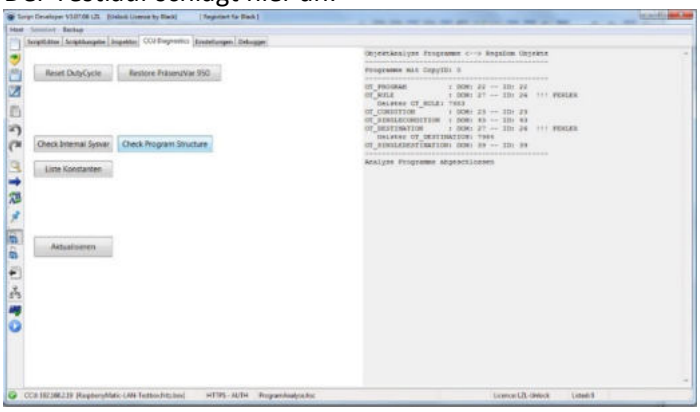
Hier wird das ursprüngliche Programm und die (Geisterkopie) daraus angezeigt. Der erste Versuche wäre: einen Reboot zu versuchen und zu überprüfen, ist der Eintrag des Geisterprogrammes weg oder nicht. wenn nicht, kann manuell versucht werden, die Objecte der Kopie (nicht des Quellprogrammes) zu löschen

2. Programmfragmentreste

In der Regadom tummeln sich Reste ehemaliger Programme, welche aber nicht mehr in der Struktur unter ID_PROGRAMS auftauchen.

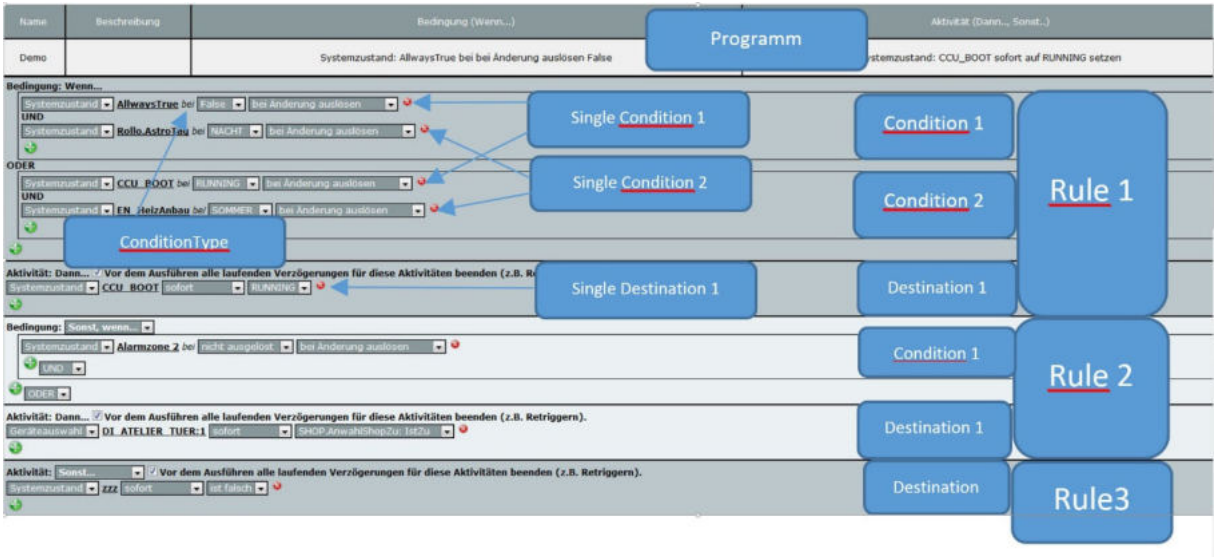
Provoziert habe ich es hier in dem Testfall, in dem ich ein RULE Object mal händisch angelegt habe.

Der Testlauf schlägt hier an:

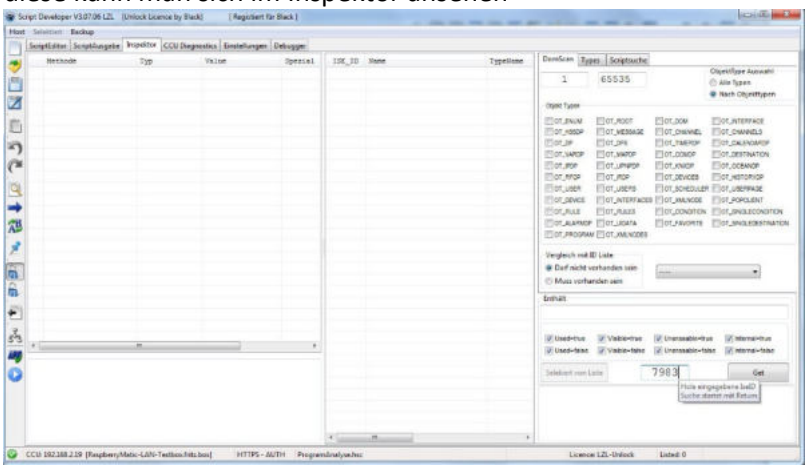


Es wird unter den Objecten OT_RULE und OT_DESTINATION angezeigt, dass sich in der Rega 2 Objekte tummeln, welche keinen Bezug zu dem Inhalt eines Programmobjektes haben. (2 deshalb, weil ein dom.CreateObject (OT_RULE) auch gleichzeitig das zu der Rule gehörende OT_DESTINATION Object erzeugt)

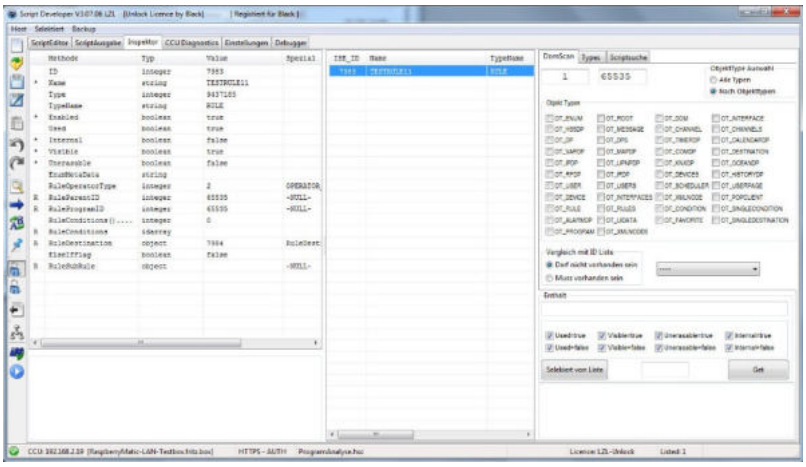
Ein Programm ist auf der CCU so Aufgebaut:



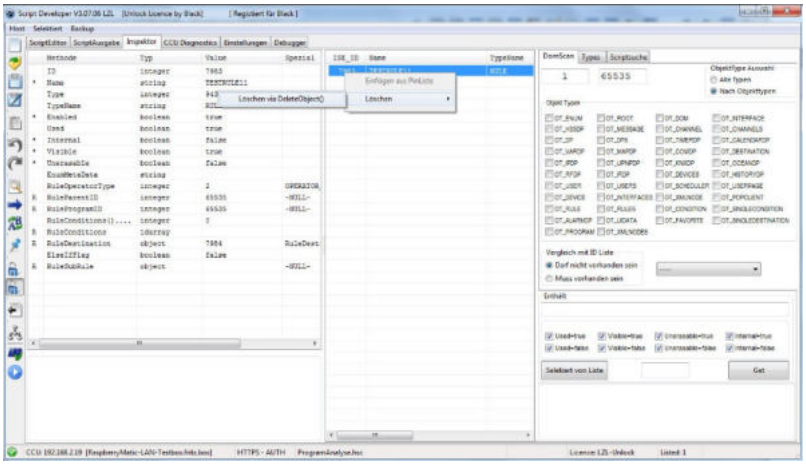
Die iseID des gefundenen Objektes wird ausgegeben
diese kann man sich im Inspektor ansehen



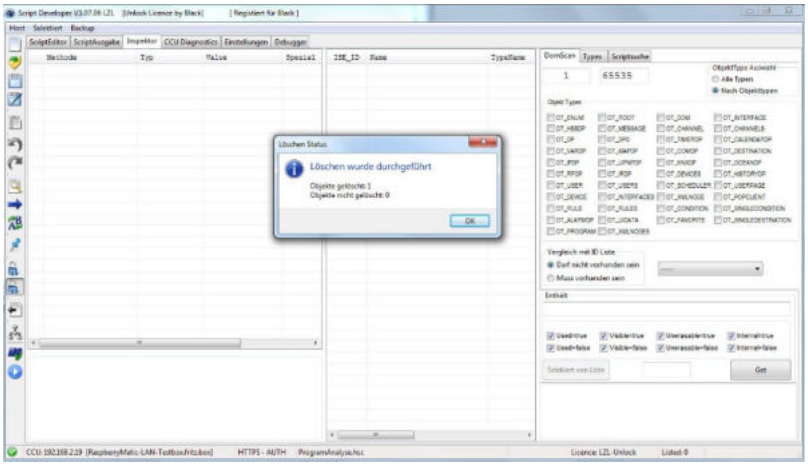
Dazu die gefundene ID in die Suchmaskte eintragen und Enter drücken



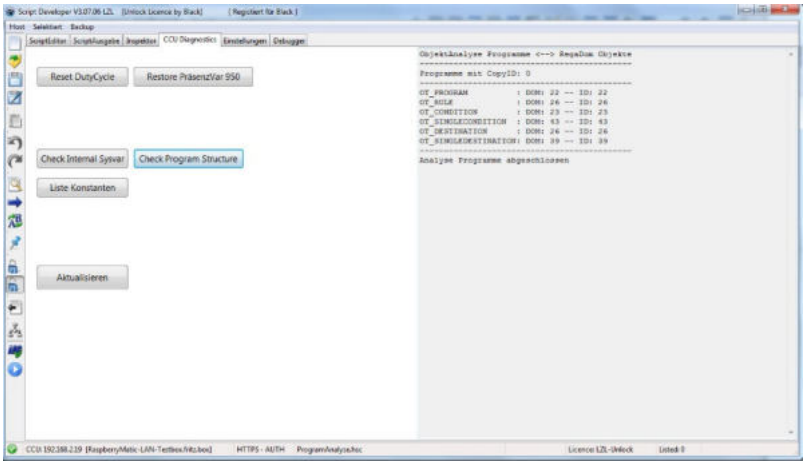
Man sieht, ein Object ohne Bezüge. es kann also entfernt werden. Löschen Freigeben, Programmunterobjekte müssen auch freigeben sein zum löschen, dann



es wurde gekillt



Die abschliessende Kontrolle zeigt:



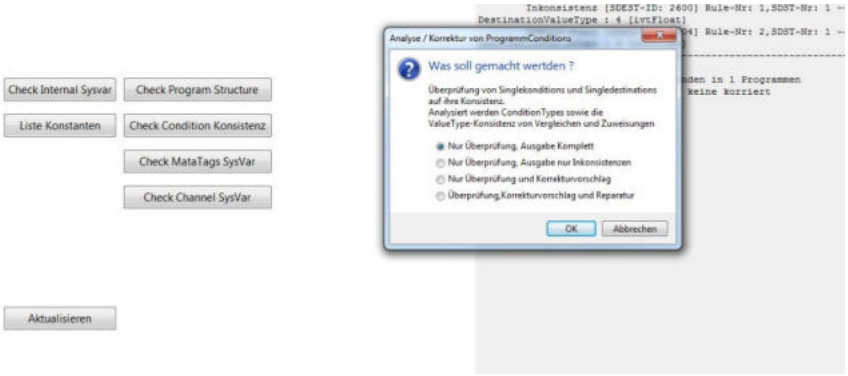
Es ist weg.

Und immer die Erinnerung: vor Löschen immer ein Backup machen, bei löschen gibt es kein Undo.

8.4.1 Check Condition Konsistenz.

Durch häufiges Ändern, oder auch internen Fehlern in der WEB-UI kann es vorkommen, dass eigentlich in sich richtig ausschauende Programme in Ihren Objektstrukturen logische Fehler aufweisen und nicht mehr triggern oder ansonsten unlogisches Verhalten aufweisen. Der bekannteste Fehler z.B. ist ein Programm mit einem Stringvergleich welches aber reproduzierbar nicht triggern wird, weil die WebUi aus dem bei Strings eigentlich vorgesehenem ConditionType 1 (bei) eine 6 (ist kleiner bis gleich) macht.

Als Vorstufe zu der Erweiterung Program backups kann der SDV nun mir bekannte logische Inkosistenzen erkennen, aufzeigen, und wenn möglich, einen Korrekturvorschlag nennen und diesen auf Anweisung auch ausführen.



Auswahl ist selbsterklärend.

Stufe 4 (mit Reparatur sollte immer erst nach Stufe 3 (Überprüfung mit Korrekturvorschlag) benutzt werden. Hier werden die Fehler aufgeführt und ein Lösungsvorschlag aufgezeigt. Wenn dieser schlüssig ist, gut, kann man den SDV drauf loslassen, wenns arg schräg aussieht kann oder muss man die Korrektur manuell machen.

```
ProgrammAnalyse Konsistenz SingleConditions & Singledestinations
Black in April 2019
------------------------------------------------------------
Analyse Program: Alarmtest [ID: 8528]
        Inkonsistenz [SCND-ID: 8567] Rule-Nr: 1,CND-Nr: 1, SCND-Nr: 1 -- LeftVal: 2 [ivtBinary], RightVal1: 4 [ivtFloat]
                Ursache Boolvergleich ivtBinary Links mit nicht ivtFloat Wert rechts
                Umrechnen von rechts Wert 15.000000 [ivtFloat] in true [ivtBinary]
                Kann vom SDV korrigiert werden
------------------------------------------------------------
Analyse Program: Stringtest [ID: 8622]
        Inkonsistenz [SCND-ID: 9258] Rule-Nr: 1,CND-Nr: 1, SCND-Nr: 1 -- LeftVal: 20 [ivtString], RightVal1: 16 [ivtInteger]
                Ursache Stringvergleich ivtString Links mit nicht ivtInteger Wert rechts
                Umrechnen von rechts Wert 25 [ivtInteger] in 25 [ivtString]
                Kann vom SDV korrigiert werden
------------------------------------------------------------
Analyse Program: USV_01_SHUTDOWN [ID: 2003]
        Inkonsistenz [SDEST-ID: 2600] Rule-Nr: 1,SDST-Nr: 1 -- DestinationDP : 2 [ivtBinary], DestinationValueType : 4 [ivtFloat]
        Inkonsistenz [SDEST-ID: 2604] Rule-Nr: 2,SDST-Nr: 1 -- DestinationDP : 2 [ivtBinary], DestinationValueType : 4 [ivtFloat]
------------------------------------------------------------
Geprüft: 37 Programme
Dabei wurden 4 Inkonsistenzen gefunden in 3 Programmen
Von diesen 4 Inkonsistenzen wurden keine korriert
```

In diesem Beispiel werden gefundene Inkonsistenzen aufgezeigt, die Ursache dargelegt und die Lösung und ob der SDV diese, wenn er mit Reparatur gestartet wird, diese korrigieren kann.

Wenn ja, wird im in einem Korrekturlauf diese Inkonsistenzen beseitigt.

Der SDV verändert dabei Programmobjekte..
AUF JEDEN FALL VORHER EIN BACKUP MACHEN !

Es gibt kein UNDO, das geht nur über Restore

```
ProgrammAnalyse Konsistenz SingleConditions & Singledestinations
Black in April 2019
-----------------------------------------------------------------
Analyse Program: Alarmtest [ID: 8528]
        Inkonsistenz [SCND-ID: 8567] Rule-Nr: 1,CND-Nr: 1, SCND-Nr: 1 -- LeftVal: 2 [ivtBinary], RightVal1: 4 [ivtFloat]
                Ursache Boolvergleich ivtBinary Links mit nicht ivtFloat Wert rechts
                Umrechnen von rechts Wert 15.000000 [ivtFloat] in true [ivtBinary]
                Kann vom SDV korrigiert werden
                Korrigiert durch RightVal1ValType (ivtBinary)
-----------------------------------------------------------------
Analyse Program: Stringtest [ID: 8622]
        Inkonsistenz [SCND-ID: 9258] Rule-Nr: 1,CND-Nr: 1, SCND-Nr: 1 -- LeftVal: 20 [ivtString], RightVal1: 16 [ivtInteger]
                Ursache Stringvergleich ivtString Links mit nicht ivtInteger Wert rechts
                Umrechnen von rechts Wert 25 [ivtInteger] in 25 [ivtString]
                Kann vom SDV korrigiert werden
                Korrigiert durch RightVal1ValType (ivtString)
-----------------------------------------------------------------
Analyse Program: USV_01_SHUTDOWN [ID: 2003]
        Inkonsistenz [SDEST-ID: 2600] Rule-Nr: 1,SDST-Nr: 1 -- DestinationDP : 2 [ivtBinary], DestinationValueType : 4 [ivtFloat]
        Inkonsistenz [SDEST-ID: 2604] Rule-Nr: 2,SDST-Nr: 1 -- DestinationDP : 2 [ivtBinary], DestinationValueType : 4 [ivtFloat]
-----------------------------------------------------------------
Geprüft: 37 Programme
Dabei wurden 4 Inkonsistenzen gefunden in 3 Programmen
Von diesen 4 Inkonsistenzen wurden 2 korrigiert
```

Ein nochmaliger Lauf zeigt diese 2 korrigierbaren nun nicht mehr als inkonsistent an.

```
ProgrammAnalyse Konsistenz SingleConditions & Singledestinations
Black in April 2019
-----------------------------------------------------------------
Analyse Program: USV_01_SHUTDOWN [ID: 2003]
        Inkonsistenz [SDEST-ID: 2600] Rule-Nr: 1,SDST-Nr: 1 -- DestinationDP : 2 [ivtBinary], DestinationValueType : 4 [ivtFloat]
        Inkonsistenz [SDEST-ID: 2604] Rule-Nr: 2,SDST-Nr: 1 -- DestinationDP : 2 [ivtBinary], DestinationValueType : 4 [ivtFloat]
-----------------------------------------------------------------
Geprüft: 37 Programme
Dabei wurden 2 Inkonsistenzen gefunden in 1 Programmen
Von diesen 2 Inkonsistenzen wurden keine korriert
```

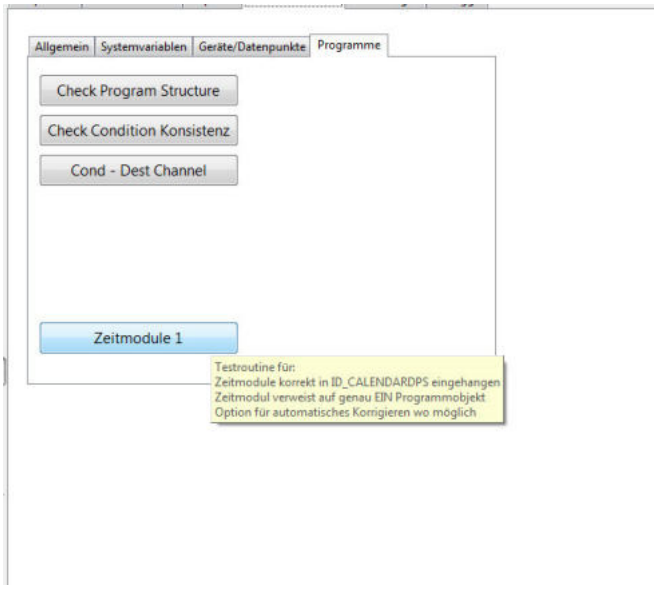
8.4.3 Cond- Destination Channel

Testroutine für:

ConditionChannel und Destinationchannel korrekter Verweis auf das referenzierte Objekt bzw ID_ERROR sowie korrekte Verwendung von ivtSystemID bzw ivtObjectId.

Hintergrund: Eine Systemvariable steht in SCNDs und SDSTs normalerweise als ivtSystemID vermerkt, HSSDPs als ivtObjectID. Systemvariablen, die Kanälen zugewiesen werden, werden auch als ivtObjectID geführt. Der Testlauf überprüft auf die korrekte Anwendung des Objectbezeichners und kann dieses auch korrigieren.

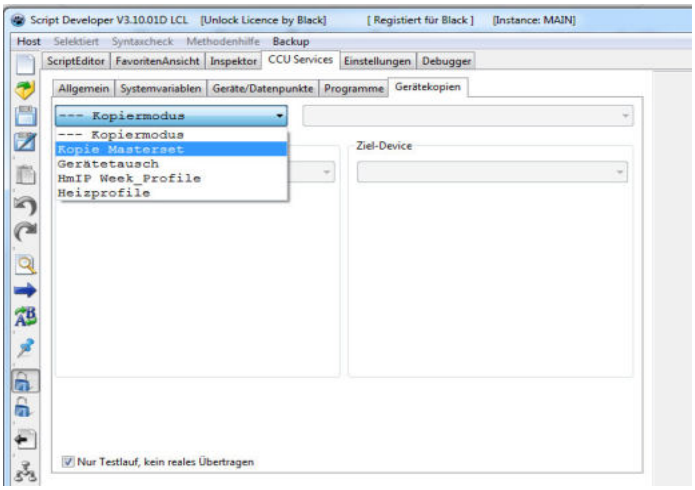
8.4.4 Test und Korrektur Zeitmodule



Seit Version 3.08.10 ist das Menü CCU Diagnostics in Reitern strukuriert. Unter Programme findet sich nun die Testoption Zeitmodule 1 . Hierbei wird geprüft auf korrektes Einhängen des Zeitmodules in ID_CALENDARDPS, ob jedem Zeitmodul genau ein Programmobjekt zugeordnet ist. Korrektur kann da erfolgen wo möglich. Auswahl erfolgt nach anklicken des Buttons „Zeitmodul 1“

9 Gerätekopieren

Unter diesem Reiter in der Hauptkategorie CCU Services verbirgt sich ein mächtiges Tool, um Geräteeinstellungen von einem Gerät in ein anderes zu kopieren bzw Geräte auszutauschen. Im gegensatz zur originalen WebUI arbeitet der SDV dabei auch mit ähnlichen (nearEqual) Geräten und auch mit HmIP Geräten.



1. Kopie Masterset: kopier die Geräteeinstellungen eines Gerätes in ein Baugleiches oder ähnliches Gerät
2. Gerätetausch: noch nicht implementiert in 3.10.02, kommt noch
3. HmIP Week_Profile kopiert die Wochenprofile aus dem entsprechenden Kanal eines HmIP Aktors in einen anderen HmIP Aktor
4. Heizprofile: kopiert die Heizprofile von Hand / Heizkörperthermostaten bzw Heizgruppen selektiv untereinander, auch zwischen HMclassic und HmIP Geräten

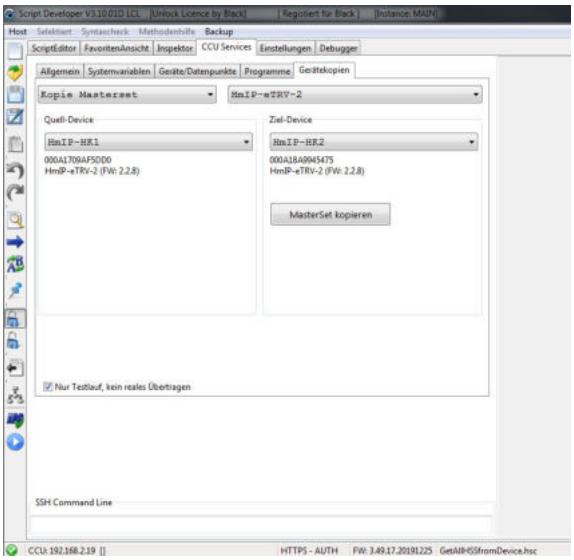
Ähnliche Geräte sind im Source Code Hardcoded, momentan sind folgende Gruppen angelegt:

```
NearEqualMASTER:= tNearEqualDevice.Create ();
NearEqualMASTER.Add ('HM-LC-Bl1-FM'#9'HM-LC-Bl1PBU-FM'#9'HM-LC-Bl1-SM');
NearEqualMASTER.Add ('HM-LC-Sw1PBU-FM'#9'HM-LC-Sw1-FM'#9'HM-LC-Sw1-DR');
NearEqualMASTER.Add ('HM-Sec-RHS'#9'HM-Sec-RHS-2');
NearEqualMASTER.Add ('HM-Sen-MDIR-O-3'#9'HM-Sen-MDIR-O-2'#9'HM-Sen-MDIR-O'#9'HM-Sen-MDIR-SM');
```

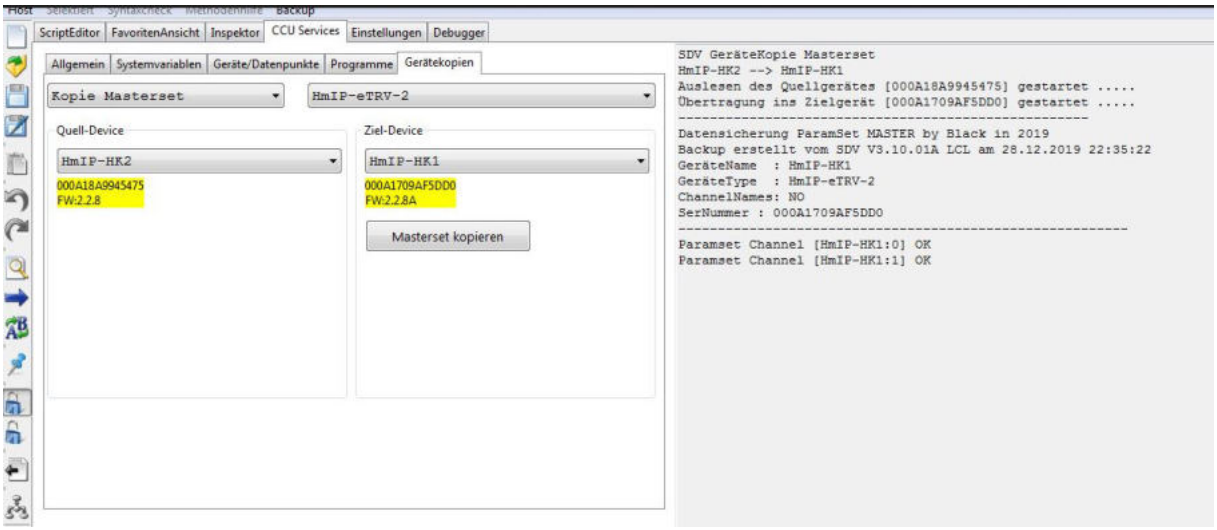
9.1 Kopieren Masterset

Als Erstes unter CCU Services - Gerätekopien den Kopiermodus einstellen Dann im nächsten Fenster die Geräteart auswählen, unter der kopiert werden soll. Es wird vom SDV eine Auswahlliste generiert, wobei immer mindestens 2 gleichartige Geräte vorhanden sein müssen. (sonst wäre es auch schlecht mit kopieren). Dann das Quell und das Zielgerät selektieren. und Masterset Kopieren drücken. daraufhin werden die Einstellungen von dem Quellgerät in das Zielgerät kopiert.

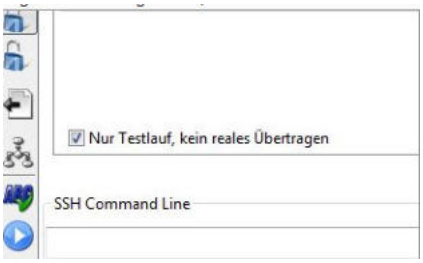
Unter Gerätetype werden die Gerätetypen vorgeschlagen, die mindestens 2 mal vorhanden sind, dann daraufhin das Ziem und das Quelldevice auswählen.



Sollten beide Geräte nicht identisch sein oder bei identischen Geräten eine unterschiedliche Firmware vorliegen, so wird das Infofeld gelb hinterlegt. Letzte Entscheidungsinstanz ist dann der Mensch



Angehakt gibt es nur Testlauf, ohne Haken wird ausgeführt... Standartmäßig ist bei Programmstart der haken gesetzt. Es werden bei der Generierung Quertests auf gegenseitige Existenz und Typgleichheit der einzelnen Keys laufen lassen. Existiert ein Key des Zieldevices nicht im Quelldevice oder ist ein Typ nicht gleich, so wird der Key samt Value des Zieldevices beibehalten und nicht verändert. Es erfolgt dann aber auch eine Meldung. ebenso wenn die Kanalnumerierung in beiden Geräten nicht passig ist. (sollte aber passen normalerweise, weil Geräte mit unterschieden lasse ich nicht als Paar zu , hardcoded)



Mögliche Fehlerbilder:
Hier existiert ein Key im Zieldevice, aber nicht im Quelldevice: Meldung

```
SDV GeräteKopie Masterset
HmIP-HK1 --> HmIP-HK2
Übernehmen 21 Values aus Quelldevice in HmIP-HK2:0
   Key ARR_TIMEOUT existiert nicht
Übernehmen 571 Values aus Quelldevice in HMIP-HK2:1
```

Ein Channel lässt sich nicht zuordnen:

```
SDV GeräteKopie Masterset
HmIP-HK1 --> HmIP-HK2
ChannelNummer zu (HmIP-HK2:0) existiert nicht im QuellDevice
ChannelNummer zu (HMIP-HK2:1) existiert nicht im QuellDevice
```

9.2 Gerätetausch

noch nicht implementiert

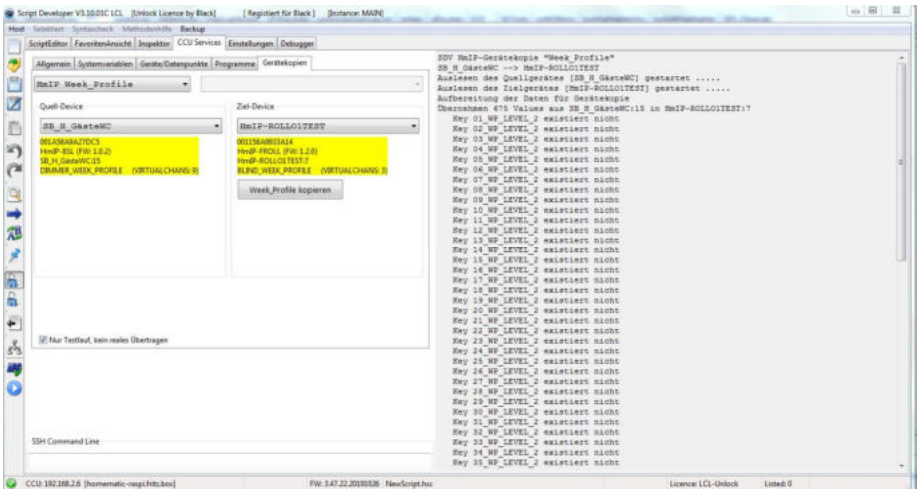
9.3 Kopieren Week Profiles

Im Menü IP WeekProfile stehen alle IP Geräte mit einem WEEK_Profile Channel zur Auswahl, hier lassen sich nun Quell und Zieldevice auswählen.
Sind beide Typ mäßig nicht gleich oder unterscheiden sich in der Kanalnummer der WEEK_Profile Channel oder haben unterschiedliche Anzahlen virtueller Kanäle so werden die Infofelder gelb hinterlegt.. Letzte Instanz Mensch.

Wird trotzdem übertragen gedrückt läuft folgende, erweiterte Logig:
prüfen ob es den Key in dem Quelldevice gibt: wenn nein, Meldung und der alte Wert im Zieldevice bleibt erhalten. Hier in Bild zu sehen, ein BSL hat im Gegensatz zu einem FROLL keinen LEVEL_2 drum die Menge an Meldungen.
dann Level: Ein Switch kennt nur 0 oder 1 ein Dimmer 0 bis 1. ist das Ziel ein Switch, werden die Werte uminterpretiert: alles außer 0 wird im Switch zu einer 1. also Dimmer etwas an bedeutet beim Überspielen in einen Switch: Switch an.

Bei den Targetkanälen wird ausmaskiert : hier auch zu sehen, der BSL hätte 9 virtuelle Kanäle, der FROLL 3. es wird beim Übertragen runtermaskiert auf die Kanäle des Zieldevices..

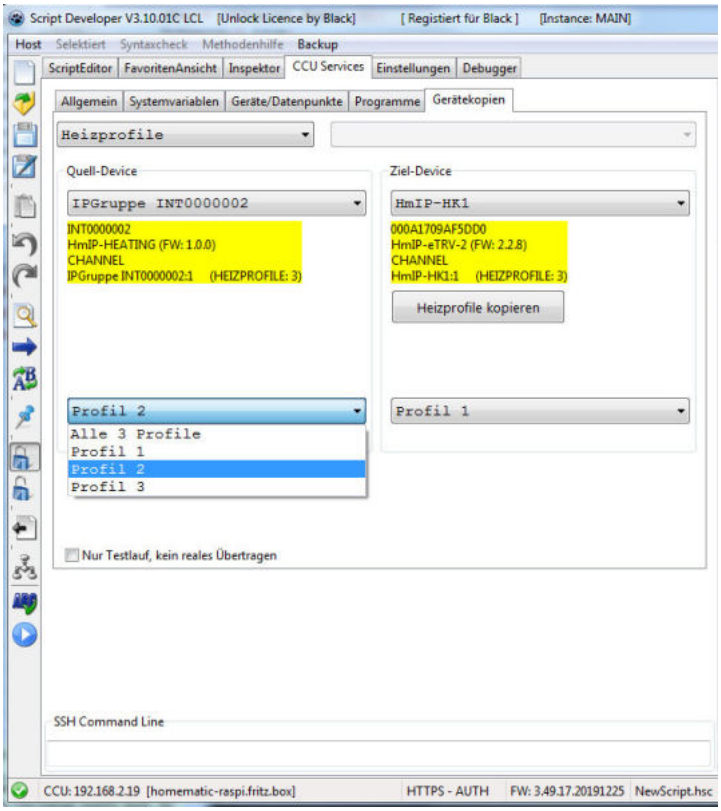
Der SDV findet die Geräte selbstständig und schlägt diese dann im Auswahlmenü vor. ebenso wird der WEEK Profile Kanal automatisch gefunden sowie die Anzahl der VirtualChans für die mask ermittelt.
Es können dabei die Wochenprofile (mit den o.g. Einschränkungen) auch unter unterschiedlichen IP Geräten kopiert werden.



9.4 Kopieren Heizprofile

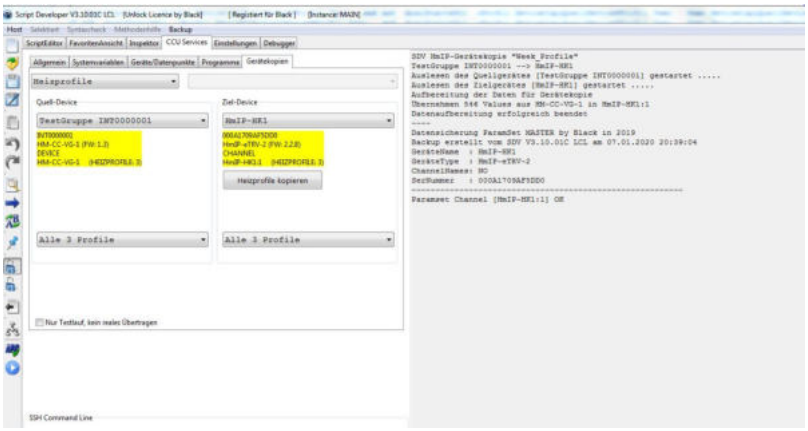
Es können Profile selektiv auch zwischen verschiedenen Gerätefamilien kopieren werden, also z.b. von IPgruppen in IP Thermostate, oder von IP Thermoste in Classic Thermostate, oder von einer Classic Gruppe in einen IP Thermostaten. Bei mehreren Profilen in einem gerät können alle Profile oder auch selektiv kopiert werden, es geht also auch: von Quellgerät Gruppe 1 in Zielgerät Gruppe 2

Bei Auswahl der Geräte wird ermittelt, welche und wieviele Profile vorhanden sind und diese dann in der Auswahl angeboten

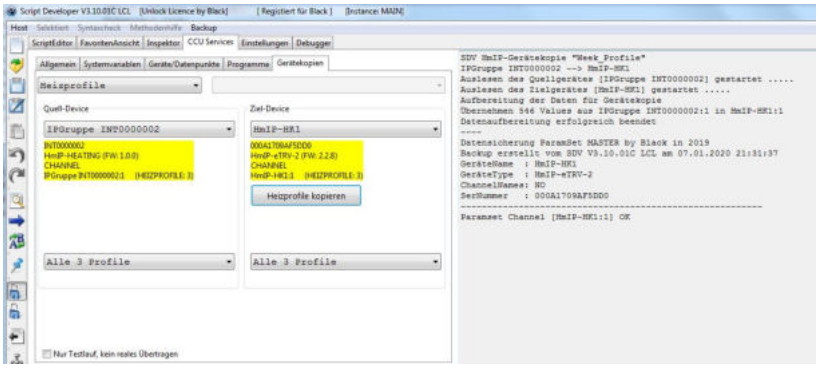


Zwischen allen Geräten classic HK Thermostat, classic WT Thermostat, classic Heizgruppe sowie zwischen IP HK Thermostaten verschiedener Ausführungen, dem IP Wandthermostaten und der IP Heizgruppe können nun wild die Heizprofile gerätegruppenübergreifend hin und her kopiert werden.

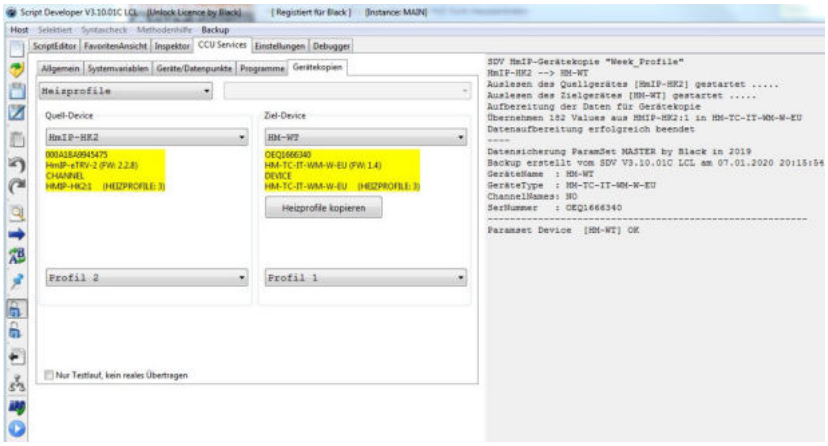
Beispiel: Kopieren aller 3 Profile einer classic Heizgruppe in einen IP HK-Thermostat



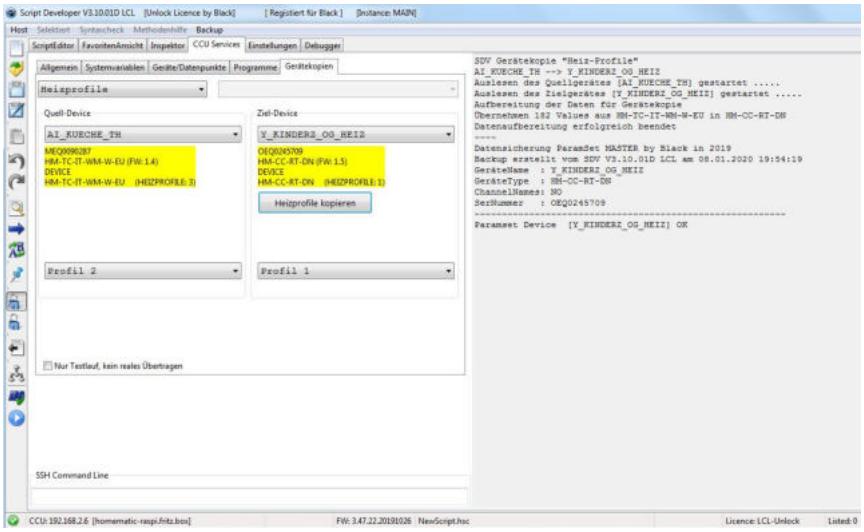
Beispiel: Kopieren alle Profile einer IP Heizgruppe in einen IP Wandthermostaten



Beispiel: Kopie Profil IP-HK Thermostat Profil 2 in classic Wandthermostat Profil 1



Beispiel: kopieren aus einem WT classic in deinen HK classic, der ja nur ein Profil hat

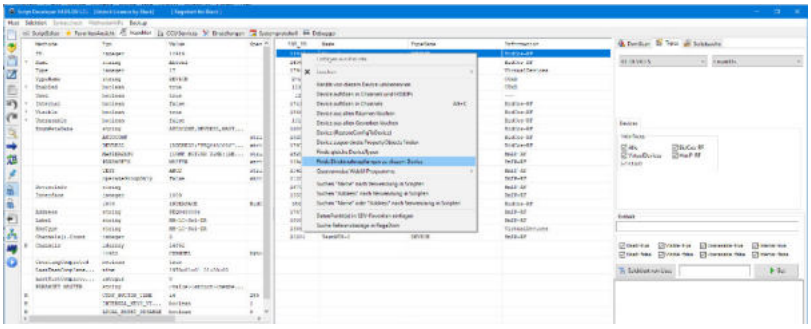


10 Direkte Verknüpfungen DVs

Der SDV unterstützt bei der Untersuchung und Bearbeitung von Direktverknüpfungen. Dies soll nicht die WEB-UI ersetzen, sondern mit Funktionen und Filtern unterstützen, welche die WebUI nicht hat.

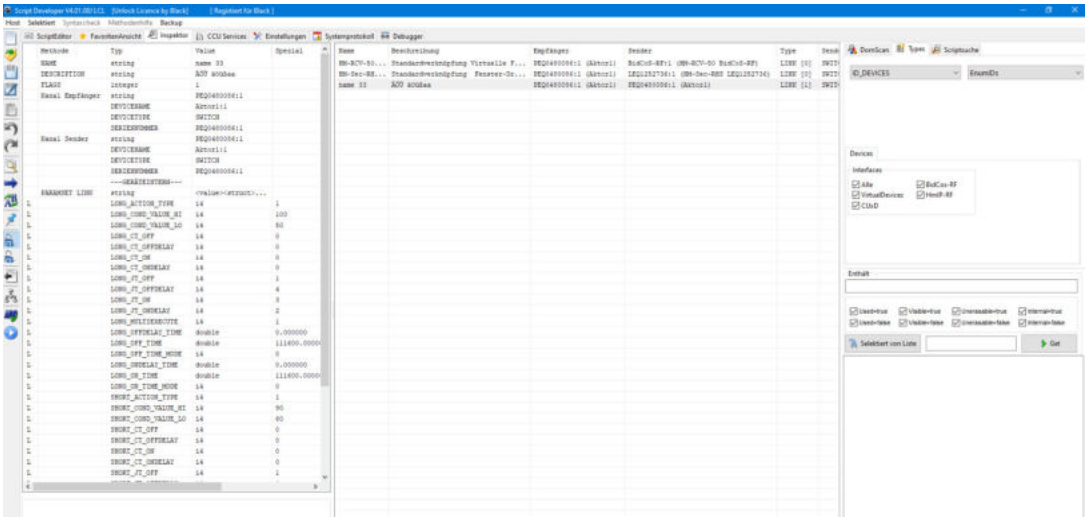
Es gibt keinen Direkten Menüpunkt für Direkte Verknüpfungen, Die Arbeitsoberfläche ist der Inspektor.

Beispiel:
Darstellen aller Direktverknüpfungen eines Gerätes



Gerät im Inspektor auswählen, rechte Maustaste und in dem Menü dann den Unterpunkt: Finde Direktverknüpfungen zu diesem Device:

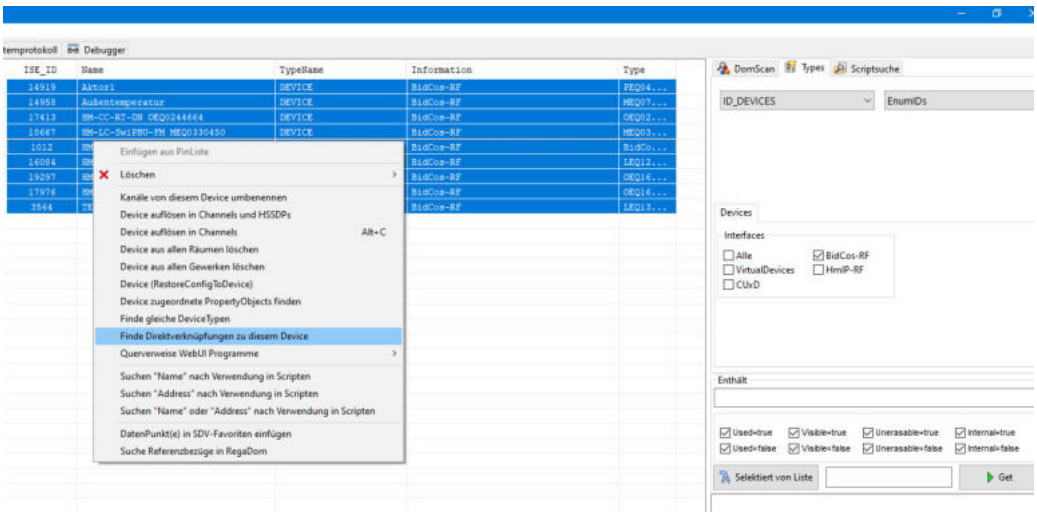
Hat das Device keine Direktverknüpfungen, so erscheint ein leeres Listenfeld, dieses hier hat aber DVs, diese werden nun gelistet:



Der SDV listet hierbei wirklich alles auf, auch Geräteinterne Direktverknüpfungen, welche in der WEbUI nicht unter DVs erscheinen.
Diese können nach allen Spaltenkriterien sortiert werden (Auf Empfänger, Sender etc klicken). Klick auf die DV in der Listenansicht öffnet in der Detailansicht alle Daten, die mit einem * gekennzeichneten Parameter können verändert werden. Aus Sicherheitsgründen ist die Namen und Beschreibung Bearbeitung bei DVs mit einem Flag<>0 gesperrt. Wie schon von den Mastersets bekannt und in der Editorbeschreibung nachzulesen ist es hier auch möglich, z.B. einen oder mehrere L Parameter zu markieren und dann im Editor automatisch den korrekten Code für PutParamset bzw getparamset erzeugen zu lassen. Der SDV parametriert, da er da Sender und Empfänger kennt, direkt den kompletten Aufruf richtig.

Natürlich kann auch direkt von mehreren oder von allen Devices die Direktverknüpfungen aufgelöst werden. Dazu in der Listenansicht mehrere oder alle Geräte anwählen, recht Maustaste und auflösen.

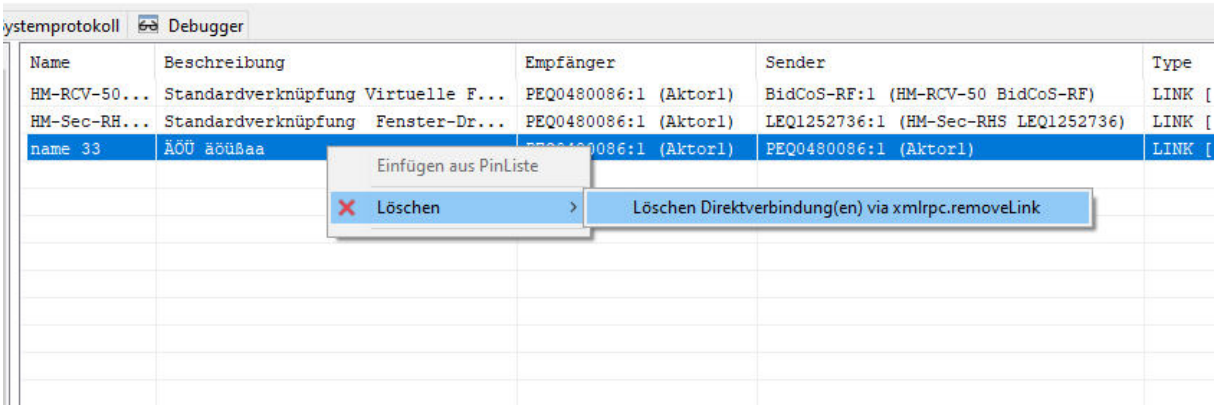
Sinnig bei der Vorauswahl ist noch: alles Geräte mit einem Interface, z.b. alle Bidcos geräte. Dazu in Device vorselektion BidCosRF anwählen, Deviceliste generieren, Ctrl-A in der Listenansicht, recht Maustaste, DVs auflösen und schon hat man alle DVs von BidCos Geräten.



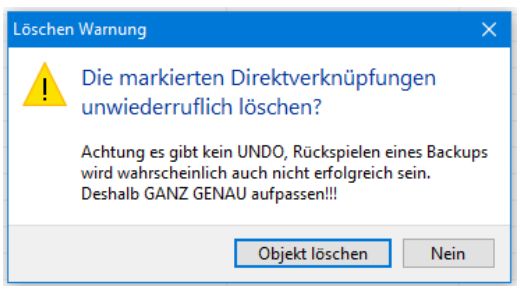
Ergibt dann aufgelöst:

10.1 Löschen von Direkten Verknüpfungen

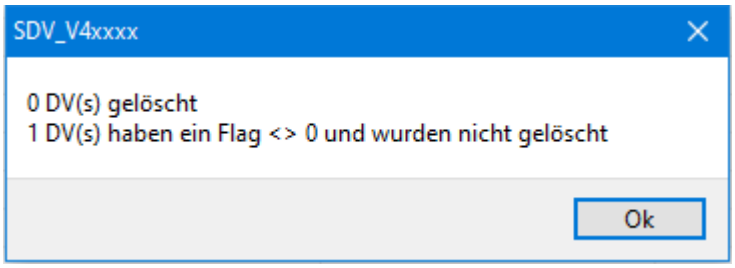
Eine oder auch mehrere Direkte Verknüpfungen lassen sich markieren und löschen. Dazu muss das Schloss offen sein und gemäß Kapitel 4.6 das löschen von DVs erlaubt sein.



Vorsichtshalber muss das nochmals bestätigt werden



Hier in diesem Fall griff dann noch der besondere Schutzmechanismus, diese DV hat ein Flag <>0 und wird deshalb, obwohl angewählt und bestätigt, nicht gelöscht.

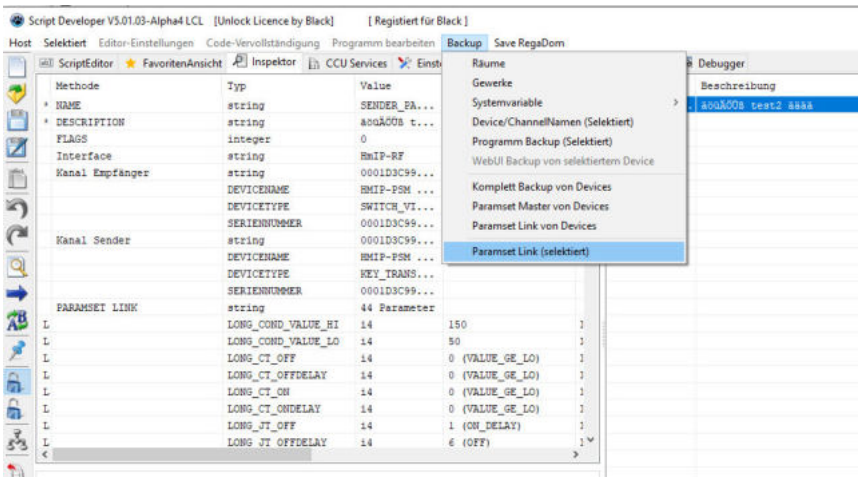


10.2 Sichern von Direktverknüpfungen

Wie im Kapitel 7.2.7 (Sichern komplettes Device) bzw 7.2.9 beschrieben können die Direkten Verknüpfungen zu einem oder auch mehreren selektierten Geräten im JSON Format gesichert werden. Dazu wie oben beschrieben Geräte wählen und dann Hauptmenü Backup.

Edit: Ab der letzten 4.x Version können auch einzelne Direktverknüpfungen exportiert werden

Dazu wie üblich DVs auflösen, DV anwählen und mit diesem Menüpunkt wird die einzelne DV exportiert



11. SDV Programmeditor

Der Programmeditor des SDV wurde von mir entwickelt, um die krampfhaften Begrenzungen und Einschränkungen der WebUI zu umgehen. Hinzu kommt der verschwenderische Umgang der WebUI mit IseIDs bei der Programmbearbeitung. Ziel sollte ein Editor sein, der nicht stylisch bunt sein musste, sondern der Möglichkeiten bietet, die in der WebUI fehlen und sich aufgrund der internen Arbeitsweise, wie in der WebUI Programme editiert werden, wohl auch nicht implementieren lassen. Unter diesem Aspekt entstand 2022 im Grundzug und 2023 dann letztlich ausprogrammiert dieser Editor.
In dem Zusammenhang einen ausdrücklichen Dank an @Nimmnenkeks und @RolandT, die in der Alpha Test und Entwicklungsphase mit Ideen und Ausprobieren immer an der bloody edge eines Alpha Systems zu dem jetzigen Editor beigetragen haben.
Ich denke auch, das ist jetzt erst der erste Wurf, es wird anhand der Möglichkeiten nun weitere Ideen und Erweiterungen geben mit der Zeit.
Der Editor hier ist nicht unbedingt gedacht für einen absolut unbedarften Anfänger, - für jemand mit bisschen Verständnis über seine CCU ist er aber ein exzellentes Hilfsmittel
Solange noch raw-beta Phase ist der Editor an Level6 gebunden, wird später auch noch Standart-5 werden. Die Anzahl an Programmzeilen ist mittlerweile auch gut im fünfstelligen Bereich (ohne Bibliothekskomponenten oder Classes aus dem SDV selber, nur für den Editor programmierte Zeilen)

11.1 Speichermanagement

Das Speichermanagement beschreibt, wie die Rega und der SDV-Programmeditor sowohl ihre Daten halten als auch mit dem Verbrauch von IseIDs umgehen.

11.1.1 Allgemeine Definitionen

Um gewisse Eigenschaften zu Umschreiben wird folgendes Wording benutzt:

11.1.1.1 Strukturverändernde Operationen

Strukturveränderung bedeutet, es wurden Änderungen im Programmeditor durchgeführt, die sich nicht in die bestehende Struktur des Ursprungsprogrammes schreiben lassen. (Beispiel: Hinzufügen / Löschen von Objekten, Verschieben von Objekten außerhalb ihres ursprünglichen Geltungsbereiches.

11.1.1.2 Werteverändernde Operationen

Eine Werteverändernde Operation verändert wie der Name schon sagt nur Werte innerhalb eines Objektes, verändert aber die Struktur eines Programmes nicht. Beispiel wäre das Ändern eines Vergleichswertes oder das Ändern einer Verzögerung etc.

11.1.1.3 Virtuelles Programm

Ein virtuelles Programm ist ein Programm, welches als neues Programm im SDV-Programmeditor erzeugt wurde. Solange dieses noch nicht durch Hochladen als reales Programm in der Rega angelegt wurde, existiert dieses nur im Speicherbaum des SDV-Programmeditors.

11.1.1.4 Childs und Parents

Ein Childobjekt ist ein Objekttyp, der unterhalb seines Parent im Baum angelegt ist. In einem Destination Objekt z.B. ist das Child die SingleDestination.

11.1.1.5 Abkürzungen und Objekthierachien

-GRP: Programmgruppe (Kommt noch, siehe Issue im SDV Thread)

- PRG: Program - Programmobjekt, die Objekte, die unter Programme in der WebUI gelistet werden

-MAIN- Condition -Maincondition – Bedingungsblock, spezielle Eigenschaften unter Main Condition

-RULE – Regelobjekt, die Wenn, SonstWenn, Sonst Blöcke der WebUI, eine Rule kann mehrere Conditions, muss aber genau eine Destination enthalten

-CND – Condition - Bedingungsblock , beinhaltet verschiedene Bedingungen
-SCND – Singlecondition - ist eine Einzelne logische Bedingung
-DST – Destination – Anweisungsblock – Eine Regel enthält immer genau einen Anweisungsblock
-SDST- SingleDestination – einzelne Anweisung, z.b. ein Skript oder eine Zuweisung

PRG
- MAIN (gibt es genau einmal bei einem PRG)
  + SCND (optional)
  + SCND (optional)
- RULE (1. Rule = WENN)
  + CND (optional, sollte aber keine leeren Bedingungen enthalten)
    . SCND
    . SCND (optional)
  + CND (n)
    . SCND (n)
    . SCND (n)
  + DST
    . SDST (sollte keine leeren Anweisungen enthalten
    . SDST (n) optional)
- RULE (2. Rule = 1. SONST WENN , oder auch Sonst)
  … (CND + DST)
- RULE (x)

Parent		Objekt		Child
		PRG	→	RULE
PRG	←	RULE	→	CND
PRG	←	RULE	→	DST
RULE	←	CND	→	SCND
RULE	←	DST	→	SDST
		PRG	→	MAIN
PRG	←	MAIN	→	SCND

11.1.2 Speichermanagement der WebUI/ der Rega

Die WebUI wirft bei jeder kleinsten Programmänderung mit dem Verbrauch von IseIDs wild um sich
Dazu ein kleines Beispiel
Gegeben sei folgendes WebUI Programm
Legende:
P: Programmkopfobjekt
T: Der Programmrumpf (dies können mehrere hundert verschiedene Objekte sein)
_: Lücken in der Rega

PT

Nun wird dieses Programm editiert
PTP_ET_E

Und das editierte Programm wird übernommen
P___T

Nun stellt man fest, eine Zahl war falsch, editieren:
$P\_\_\_TP_ET_E$

Und wieder übernommen
P_____T

Durch die interne Arbeitsweise (Programmkopie in der Rega selber) und eine höchst ineffektive Zusammenführung des geänderten Programmes wachsen die Lücken in der Rega unaufhörlich.
Vergleichbar ist es mit der Expansion des Universums. Die Materie (die Objekte) bleiben mengenmäßig gleich, der Raum (die Regalücken) wachsen unaufhörlich. Mir sind Systeme bekannt, deren IDs im 1000000 Zahlenraum angekommen sind und hauptsächlich aus Lücken bestehen.

11.1.3 Speichermanagement des SDV Programmeditors

Im Gegensatz zur Rega/WebUI arbeitet der SDV-Programmeditor komplett RAM orientiert. Von dem/den Programmen werden Tree-structure Abbilder mit allen benötigten Elementen im Arbeitsspeicher des Computers angelegt.
Damit sind alle Änderungen Rega unabhängig, es können alle erdenklichen Operation, die mit dem WebUI Programmeditor nicht gingen, durchgeführt werden. Dazu zählt:

- Reihenfolge von Objekten frei verschieben (auch Regel Objekte)
- Kopieren von Objekten und Einfügen an beliebiger Stelle (Auch aus anderen Programmen)
- Löschen bzw Einfügen von Objekten an beliebigen (aber erlaubten) Stellen
- Direktes Bearbeiten von Programmstrukturelementen
- Möglichkeiten erweiterter Eigenschaften die die WebUI nicht kennt (Main-Conditions, Logikerweiterung)
-

Dank RolandT (der mich mit seinem Undo Redo Wunsch bis kurz vor den Wahnsinn getrieben hat) verfügt der SDV Programmeditor über eine (momentan 100 stufige) Undo/Redo Funktionalität, jede Form von Änderungen können zurückgenommen werden.

Das geänderte Programm muss natürlich dann am Ende entweder verworfen (Die Änderungen finden sich nicht in der Rega wieder oder gespeichert werden.

11.1.4 Speicherstrategie des SDV-Programmeditors

Ein Programm kann vom SDV-Programmeditor mit mehreren möglichen Strategien gespeichert werden. Der SDV-Programmeditor räumt dabei jeweils seinen Bereich der Rega auf, um möglichst wenige, im Idealfall keine Lücken zu hinterlassen.

11.1.4.1 Speichern (Komplett Anlage)

Dieser Punkt kann bei jeder Änderung durchgeführt werden, Strukturverändernde Operationen lassen sich allerdings nur so in der Rega speichern:
Strategie: Das Komplette Programm wird aus der Rega gelöscht, anschließend wird das Programm komplett neu geschrieben.
xxPT
xxPT (Änderung im Ram)
xx (Löschen des Programms)
xxPT (nach Neuschreiben)

Wenn auch mehrere strukturelle Änderungen am Programm hintereinander gemacht werden, entstehen keine Lücken, da der SDV-Programmeditor vor dem Speichern alles wegräumt und dann neu sauber schreibt.

Für den Hinterkopf: Hier wird alles neu geschrieben, auch die ID des Programmobjektes kann sich hierbei ändern. Dies mögen manche Middlewares nicht, die dann ein Neueinlesen der Rega erwarten. Dafür gibt es den nächsten Punkt ^^

11.1.4.2 Speichern Komplettanlage (ID von Programm bleibt)

Dies entspricht dem Punkt 11.1.4.1, nur das das Programmobjekt beim Neuschreiben seine ursprüngliche ID behält, der Rest wird aber weggeräumt und wie unter 11.1.4.1 neu geschrieben

xxPT
xxPT (Änderung im Ram)
xx (Löschen des Programms)
xxP (Neuanlage des Programmobjektes unter alter alten ID)
XXPT (nach Neuschreiben)

Diese Form ist Middleware freundlich

11.1.4.3 Speichern Override

Wurden nur Werteverändernde Operationen gemacht, die Struktur ist aber identisch, so kann via Override das im SDV Programmeditor geänderte Programm direkt in die Struktur des Urspungsprogrammes geschrieben werden. Dies braucht überhaupt keine neuen IDs.

Keine Sorge, man muss dieses nicht alles im Kopf behalten, der SDV-Programmeditor prüft anhand eines Hashes, ob Strukturelle oder nur Werteändernde Operationen vorlagen und graut entsprechende Menüpunkte aus. Ebenso gibt's eine voreinstellbare Speicherstrategie, welche dies automatisiert nach den Prioritäten abhandelt, so dass nur der der Speicher Button von Relevanz ist. Trotzdem ist es hilfreich, die interne Arbeitsweise der Rega als auch des SDV-Programmeditors zu kennen und bewerten zu können

xxPT
xxPT (Änderung im Ram)
xxPT (nach überschreiben)

11.1.4.5 Virtuelles Programm real in der CCU speichern (Neuanlage)

Legt ein virtuelles Programm in der Rega der CCU an, dadurch wird ein virtuelles Programm zu einem normalen, realen Programm.

11.1.4.6 Speichern unter neuem Namen (Neuanlage)

Erstellt eine Kopie des bearbeiteten Programmes, in der Rega, wobei dem Namen bei Speichern der Suffix „_SDVKopie" hinzugefügt wird. Entspricht funktional 11.1.4.1

11.1.4.7 Reload von CCU Programm (Programm nach Programmnamen)

Sinnig für den Fall, dass eine Änderung im SDV-Programmeditor sich während der Arbeit als falscher Weg herausstellt, der Punkt verwirft alle Änderungen und liest das Rega Programm neu in den Speicher.

11.1.4.4 Speichern Override (Extend)

Noch nicht freigegeben, soll eine Mischung aus 11.1.4.3 und 11.1.4.2 werden

11.2 Der Programmeditor

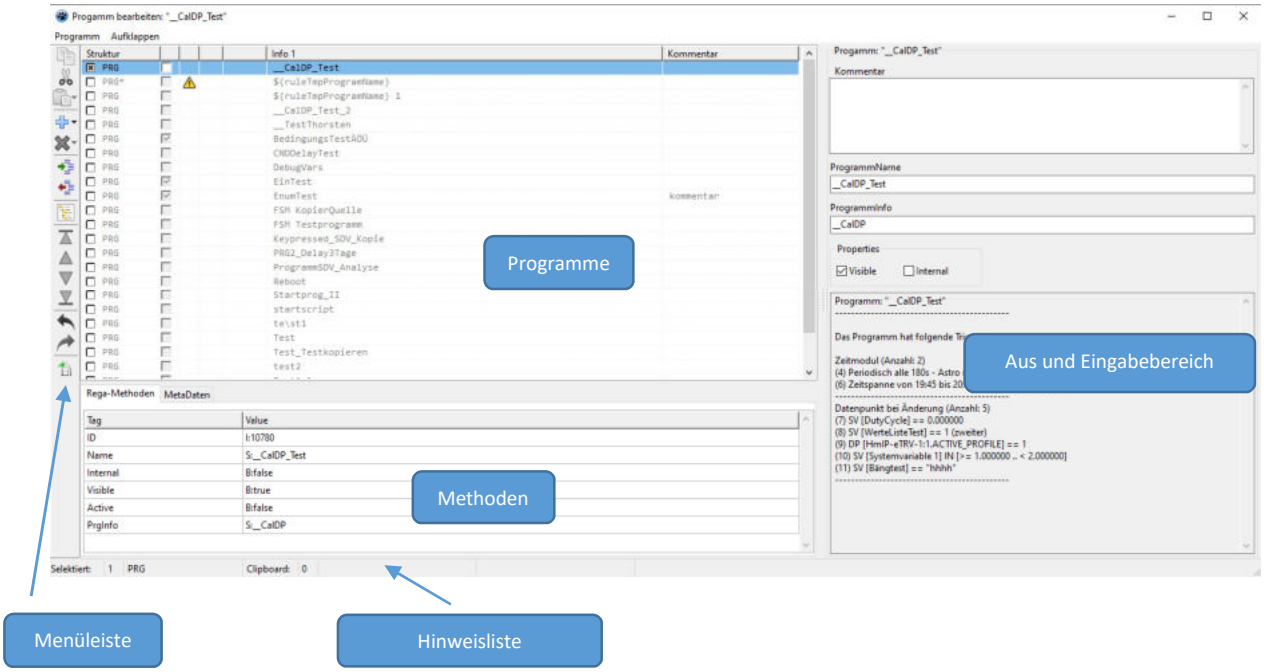
Der Programmeditor wird gestartet, indem der Menüpunkt des Hauptmenüs (Programme bearbeiten) angeklickt wird. Ist im Inspektor in der Listenansicht kein Programm selektiert, so startet der SDV-Programmeditor ohne aktives Arbeitsprogramm. Ist im Inspektor ein Programm selektiert so startet der Editor mit diesem selektierten Programm als aktives Arbeitsprogramm.

Erstmal ein kleines Vorwort zu dem Programmeditor. Im Programmeditor kann immer ein Programm gleichzeitig bearbeitet werden. Dieses ist das Arbeitsprogramm. Das Arbeitsprogramm kann auch gewechselt werden. Alle anderen Programme können dargestellt und auch in Einzelheiten betrachtet werden, aber aktive Änderungen gehen nur im Arbeitsprogramm.

Im SDV-Programmeditor wird immer das Arbeitsprogramm und wahlweise alle oder die ebenfalls zur Anzeige selektierten Programme dargestellt. Dies ermöglicht eine Anpassung der Anzeige und Erhöhung der Übersichtlichkeit. Soll nur in einem Programm schnell ein Wert geändert werden so ist es sinnvoll, nur das Arbeitsprogramm darzustellen. Soll aus einem bestimmten Programm Teile in das Arbeitsprogramm kopiert werden, so wählt man nur dieses Programm zusätzlich an und blendet alle anderen aus.

Das Arbeitsprogramm

- steht im SDV-Programmeditor immer ganz oben.
- wird in schwarzer Schrift dargestellt



Welches Objekt selektiert ist (blaue Zeile) wird hier angezeigt:

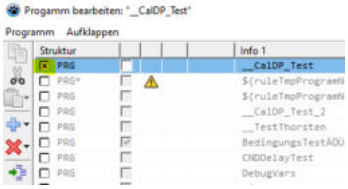
Selektiert:	1	PRG

Welches Objekt im Clipboard liegt, wird hier angezeigt:

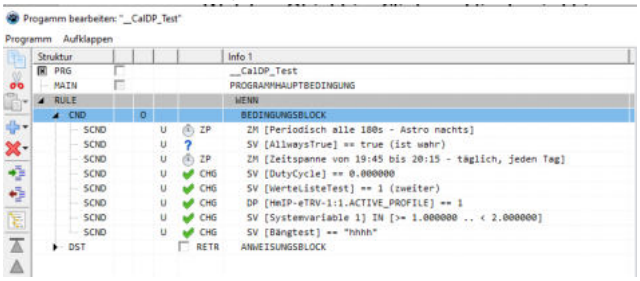
Clipboard:	1	SCND

Das Arbeitsprogramm steht immer an 1. Stelle:

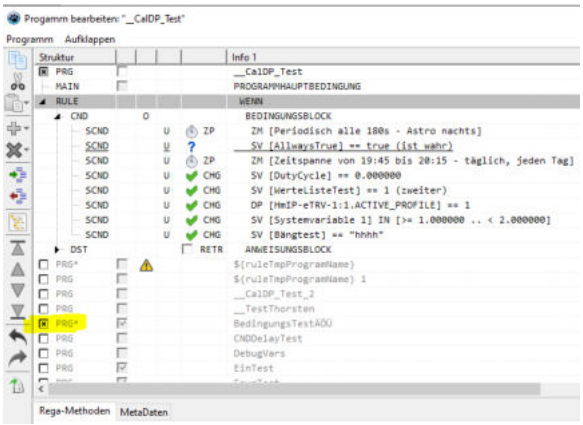
Das gelb markierte, bei einem Arbeitsprogramm ausgegraute Kreuz gibt an, das Programm ist immer eingeblendet



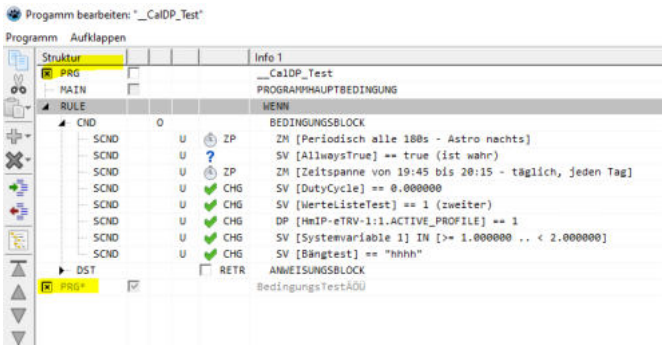
Soll nur in dem Arbeitsprogramm etwas gemacht werden, lassen sich alle anderen Programme schnell ausblenden, bei Bedarf mit einem Button natürlich auch wieder einblenden.



Soll noch ein oder auch mehrere weitere Programme sichtbar sein, der Rest aus Übersichtlichkeitsgründen aber ausgeblendet werden, so lassen sich in den nicht Arbeitsprogrammen die Kreuz Markierungen setzen und dann lassen sich alle Programme ohne Markierungen ausblenden oder aber wieder einblenden

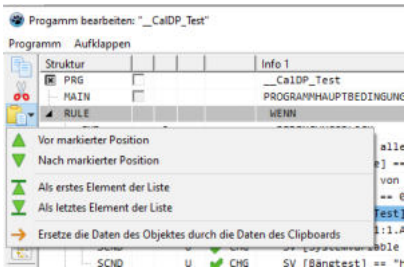


Teile dieses/dieser Programme können dann z.b. als Kopierquelle genutzt werden



Doppelklick auf eine Zeile öffnet vorhandene Unterobjekte (Vorhandensein von Unterobjekten zeigt das schwarze Dreieck)

11.2.1 Positionsangaben beim Einfügen / Generieren



11.2.1.1 Vor markierter Position

Das in der Zwischenablage liegende Objekte / das neu erzeugte Objekt wird vor der markierten Position im Editor eingefügt

11.2.1.2 Nach markierter Position

Das in der Zwischenablage liegende Objekte / das neu erzeugte Objekt wird nach der markierten Position im Editor eingefügt

11.2.1.3 Am Anfang der Liste

Das in der Zwischenablage liegende Objekte / das neu erzeugte Objekt wird als erstes Element in der Liste dieser Objekte dargestellt

11.2.1.4 Am Ende der Liste

Das in der Zwischenablage liegende Objekte / das neu erzeugte Objekt wird als letztes Element in der Liste dieser Objekte dargestellt

11.2.1.5 Ersetze Daten des Objektes durch Daten des Clipboards

Dies kopiert nicht das Objekt selber , sondern nur den Inhalt des Objektes in ein Objekt des gleichen Types

11.2.1.6 Als erstes Element der Ziel-ChildListe

Beispielhaft: im Clipboard liegt ein SCND Objekt, Angeklickt ist ein CND Objekt, so kann das SCND Objekt des Clipboards am Anfang der SCND Liste des CND Objekte eingefügt werden

11.2.1.7 Als letztes Element der Ziel-ChildListe

Beispielhaft: im Clipboard liegt ein SCND Objekt, Angeklickt ist ein CND Objekt, so kann das SCND Objekt des Clipboards am Ende der SCND Liste des CND Objekte eingefügt werden

11.2.2 Menüleiste

Die Buttons der Menüleiste haben alle die Hintfunktionalitäten hinterlegt. Tasten, die im Augenblick nicht freigegeben sind, werden ausgegraut dargestellt

11.2.2.1 Kopieren

Das selektierte Objekt (samt seinen untergeordneten Child-Objekten) wird auf das Editoreigene Clipboard kopiert. (Tastenkürzel CTRL+C)

11.2.2.2 Ausschneiden

Nur im Arbeitsprogramm möglich. Das selektierte Objekt (samt seinen untergeordneten Child-Objekten) wird an der aktuellen Stelle ausgeschnitten und auf das Editoreigene Clipboard kopiert. (Tastenkürzel CTRL+X)

11.2.2.3 Einfügen

Nur im Arbeitsprogramm möglich. Das Objekt auf dem Clipboard wird an der markierten Stelle im Programm eingefügt. Der SDV Editor prüft dabei intern, ob ein Einfügen des Objektes des Clipboards an der Stelle möglich ist. Welches Objekt im Clipboard liegt, lässt sich in der Statuszeile sehen. 

Tastenkürzel CTRL+V

11.2.2.4 Erzeugen

Erzeugt den angegebenen Objekttyp an der im aufklappenden Untermenü ausgewählten Stelle. Es ist hier auch möglich, mitten in einem Programm ein zusätzliches Regelobjekt zu erzeugen. Wenn ein Programmobjekt angeklickt ist, so enthält das Auswahlmenü zusätzlich auch die Möglichkeit ein zusätzliches, neues Programm zu erzeugen. Ein neues Programm hat erst einmal immer den Namen: „Neues Programm (UTC des Erzeugens)“ Tastenkürzel Ctrl + Plustaste

11.2.2.5 Löschen

Löscht das angewählte Objekt. Ein Programm kann auch gelöscht werden, dazu muss es aber das Arbeitsprogramm sein. Es erfolgen auch entsprechende Sicherheitsabfragen.

Bis auf das Löschen von Programmen kann jede Aktion durch Undo/Redo wieder rückgängig gemacht werden

11.2.2.6 Alle Programme darstellen

Alle Programme, auch die die nicht angekreuzt sind, werden dargestellt

11.2.2.7 Nur angekreuzte Programme darstellen

Stellt nur noch die angekreuzten Programme dar, alle anderen werden ausgeblendet

11.2.2.8 Alle untergelagerten Objekte (Childs) aufklappen

Alle von dem angewählten Objekt vorhandenen untergeordneten Objekte (Childs) werden aufgeklappt. Wird benutzt zum kompletten Aufklappen eines Programmes oder eines anderen, noch eingeklappten Unterpunktes.

11.2.2.9 Verschieben an Anfang

Das angewählte Objekt wird an die erste mögliche Position in seinem Parent verschoben

11.2.2.10 Verschieben um eine Position nach oben

Das angewählte Objekt wird um eine Position nach oben in seinem Parent verschoben

11.2.2.11 Verschieben um eine Position nach unten

Das angewählte Objekt wird um eine Position nach unten in seinem Parent verschoben

11.2.2.12 Verschieben ans Ende

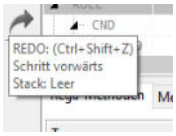
Das angewählte Objekt wird an die letzte mögliche Position in seinem Parent verschoben

11.2.2.13 UNDO

Einen Bearbeitungsschritt rückgängig. Jede Aktion, sei es Einfügen, Verschieben, Werte ändern kann mit Undo Rückgängig gemacht werden. Es ist nicht so wie in der WebUI, eine Anweisung löschen und weg ist sie. Nur das Löschen von ganzen Programmen kann so nicht mehr rückgängig gemacht werden. Tastenkürzel Ctrl-Z. Der Hint zeigt die aktuelle Tiefe des Undo Stacks:

11.2.2.14 REDO

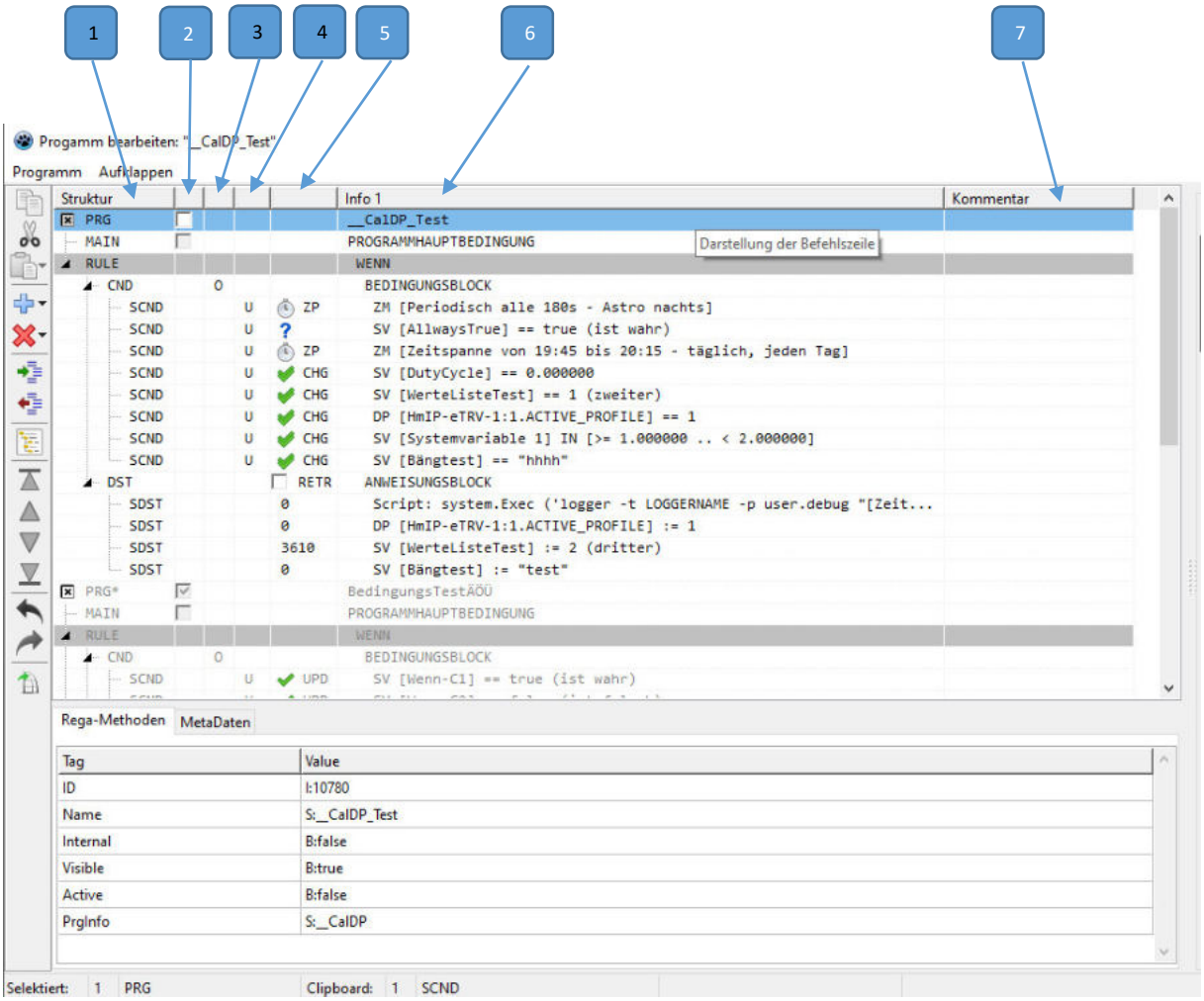
Ein oder mehrere REDO Schritt wieder rückgängig machen. Jede Aktion, sei es Einfügen, Verschieben, Werte ändern kann mit Undo Rückgängig gemacht werden. Es ist nicht so wie in der WebUI, eine Anweisung löschen und weg ist sie. Nur das Löschen von ganzen Programmen kann so nicht mehr rückgängig gemacht werden.
Tastenkürzel Ctrl-Shift-Z. Der Hint zeigt die aktuelle Tiefe des Redo Stacks:



11.2.2.15 Upload in CCU

Das Arbeitsprogramm wird gemäß eingestellter Strategie in die CCU geladen. Ein Virtuelles Programm wird in der CCU neu erzeugt. Ein Upload löscht den Undo/Redo Stack.

11.2.3 Spalten des Programmeditors



1. Information über die Objektart, das Markieren durch Ankreuzen und die unterlagerte Struktur
2. Aktiv Spalte: Ein Programm oder auch eine Main Condition ist aktiviert
3. Verknüpfung Bedingungsblock in SPS Schreibweise
4. Verknüpfung Einzelne Bedingung in SPS Schreibweise
5. Aktionsspalte:

- Einzelbedingung: Art der Triggerung bzw nur Prüfen
- Anweisungsblock: Retriggern oder nicht (Checkbox)
- Einzelanweisung: Verzögerung der Ausführung in Sekunden

6. Infospalte:
   Beschreibung des Objektes

- PRG= Programmnamen
- RULE= Art und Anzahl der Regel
- CND= Bedingungssblock
- SCND= Art der Anweisung (Kurzschreibform)
  ZM: Zeitmodul
  DP: Datenpunkt eines Gerätes
  SV: Systemvariable
  AV: Alarmvariable
- DST: Anweisungsblock
- SDST: Art der Anweisung in Kurzschreibweise
- Skript: Homematic Skript
- DP: Datenpunkt eines Gerätes
- SV: Systemvariable
- AV: Alarmvariable

7. Kommentar Jedes Objekt kann mit einem Kommentar versehen werden. Dieser ist allerdings von mir begrenzt auch maximal 500 Zeichen. (Krieg und Frieden oder die Bibel sollte also nicht als Kommentar hinterlegt werden). Damit ist endlich eine Beschreibung der Aktion, des Vergleiches möglich. Die WebUI beherrscht das nicht, der Inspektor des SDV schon.

Das Methodenfeld ist in der jetzigen Version noch sichtbar. (Debugginghilfe). Hier ist hinterlegt, wie der SDV-Programmeditor das Programm der WebUI eingelesen hat bzw. wie er dieses wieder zusammenbauen würde. Im Gegensatz zum Inspektor, wo immer nur ein Schritt geändert werden kann und dies auch direkt in der Rega gemacht wird, arbeitet der SDV-Programmeditor ja komplett im Speicher des PCs.

Im Aus/Eingabebereich des SDV-Programmeditors können die Informationen zu einem Objekt in menschenlesbar freundlicher Form dargestellt werden bzw. wenn dieses ein Arbeitsprogramm ist, dort auch geändert werden.

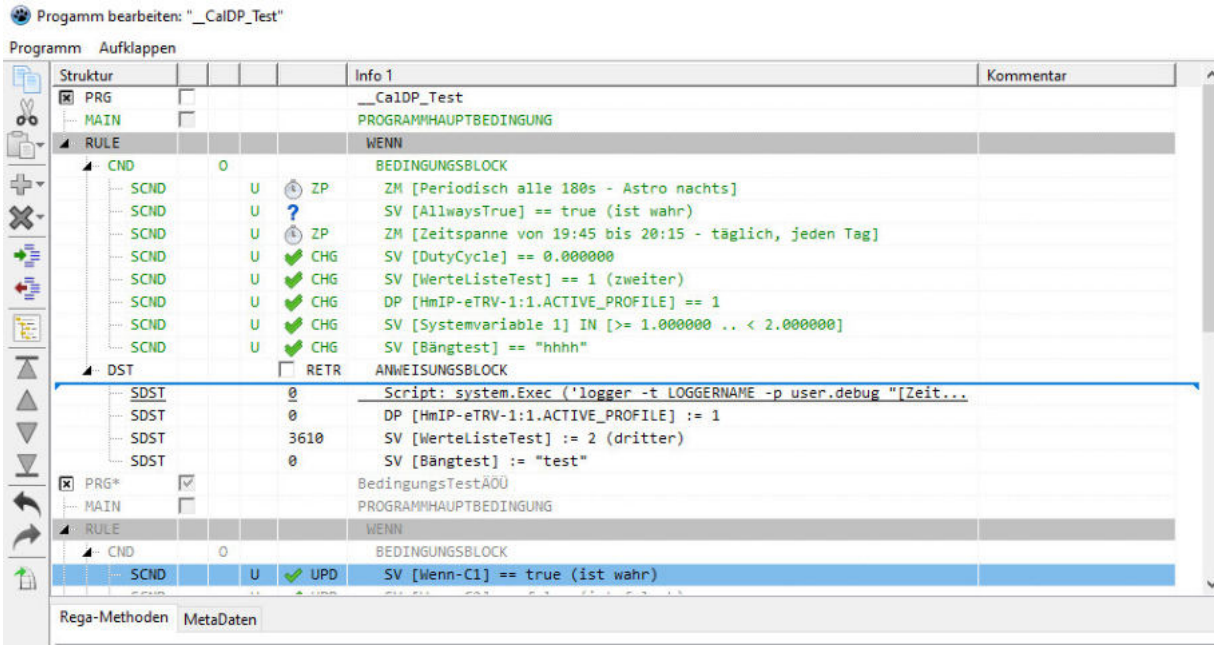
11.3 Drag – Drop

Über Drag Drop lassen sich Teile des Arbeitsprogrammes innerhalb des Arbeitsprogrammes verschieben bzw. Teile von Nicht Arbeitsprogrammen in das Arbeitsprogramm hineinkopieren.

Hier Beispiel: Dragmode Kopieren von einem Objekt aus einem Nicht-Arbeitsprogramm

Die Zeilen, wo das markierte Objekt abgelegt werden darf, werden grün eingefärbt. Bei Zeilen wo keine Ablage möglich ist, ist der Dragcursor auch ein stilisiertes Verbotsschild.

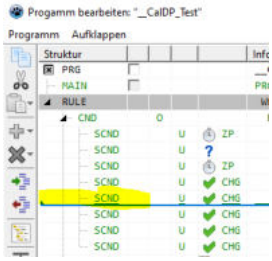
Hier im Beispiel: eine SCND ist markiert, sie darf Abgelegt werden im MAIN, in einer CND und in SCNDs.



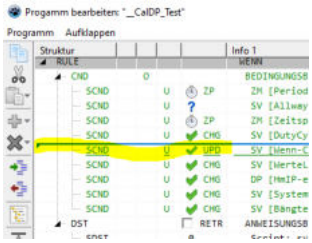
Genaues Bestimmen der Ablageposition. Am besten an diese Vorgehensweise halten, diese ist immer zielführend.
Objekt markieren mit rechter Maustaste, Maustaste aber ca 1 Sekunde gedrückt halten, bis in der Statuszeile der Dragmode erscheint. Nun ist das Control im Verschiebemodus. Abgebrochen wird dieses mit Escape oder Ablegen des Objektes in einem verbotenen Bereich.

Dann in der ersten Spalte die Maus an die Position bewegen, wo abgelegt werden soll. Man sieht dort auch schon einen blauen Teilrahmen. Im Dragmode klappen unterlagerte Objekte beim Draufziehen der Maus automatisch auf.

Das Objekt wird hinter dem mit der Mausposition definierten (hier gelb markierten) Objekt eingefügt. Wenn dies das Parent Objekt des markierten Quellobjektes ist, erfolgt das Einfügen in der Childlist an letzter Stelle.



Das Objekt wird vor dem mit der Mausposition definierten (hier gelb markierten) Objekt eingefügt. Wenn dies das Parent Objekt des markierten Quellobjektes ist, erfolgt das Einfügen in der Childlist an erster Stelle.



Loslassen der Maus führt dann zum Ablegen des Objektes samt aller seiner Untergeordneten Objekte an der definierten Stelle.

Sollte dabei ein Fehler passiert sein.. Richtig, der SDV-Programmeditor hat eine UNDO- REDO Funktion, mit der sich alle Schritte (bis auf Löschen von Programmen) zurücknehmen und auch wieder vorspulen lassen.

11.4 Undo Redo

Der SDV-Programmeditor hat einen recht tiefen Undo-Redo Stack (zum jetzigen Zeitpunkt 100 Stufen). Damit lassen sich alle Aktionen (Einfügen, Verschieben, Drag Drop, Werteänderungen, einfach alles, bis auf das komplette Löschen von Programmen) auf Knopfdruck wieder zurücknehmen. Die Idee dazu stammte von RolandT, die Möglichkeiten sind natürlich weitreichend, die Programmierung dessen hatte es aber in sich.

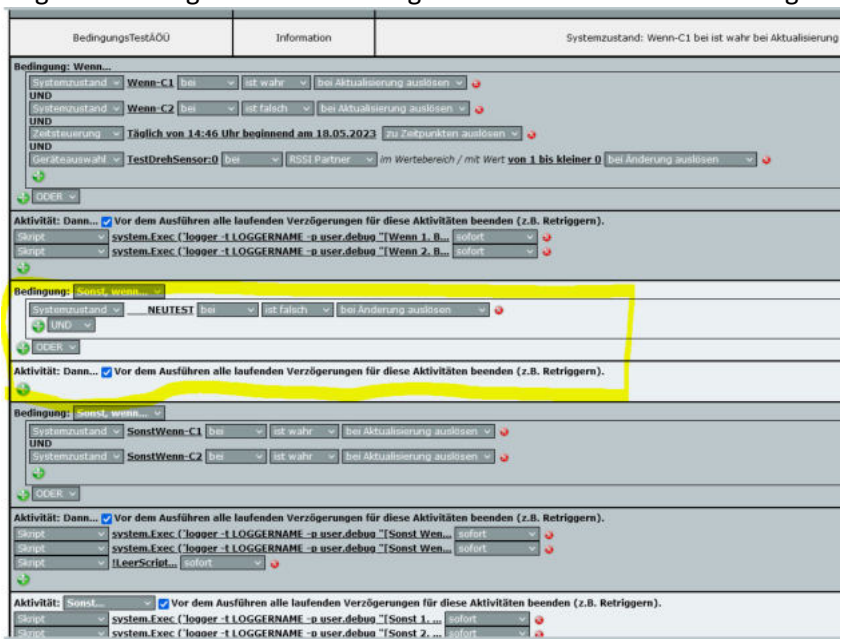
Aber. Es Geht.

11.5 Exemplarische Aufgabenstellungen

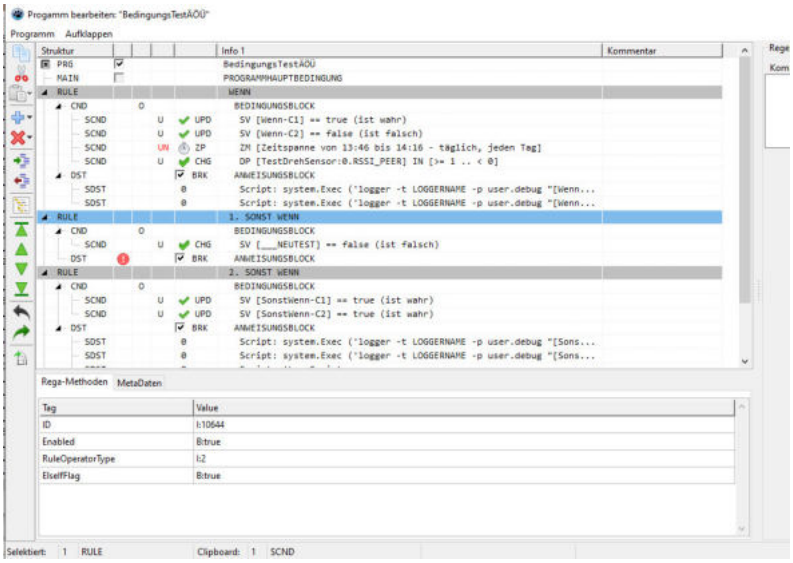
Die nun hier beschriebenen Aufgabenstellungen sind mit dem SDV-Programmeditor in wenigen Sekunden erledigt, während dies in der WebUI nur durch aufwendige und langwierige Maus-Clicki Orgien zu lösen ist.

11.5.1 Löschen einer Regel inmitten eines Programmes

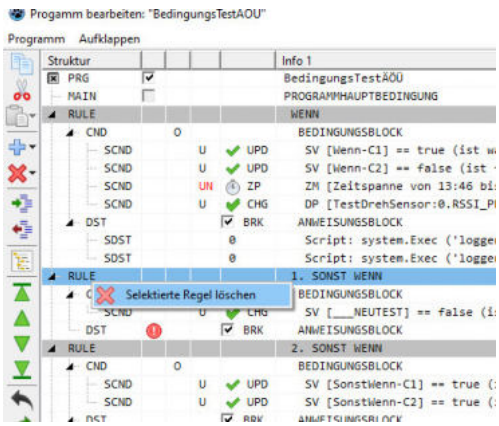
Gegeben sei folgendes WebUI Programm. Die Gelb umrandete Regel mittendrin soll weg



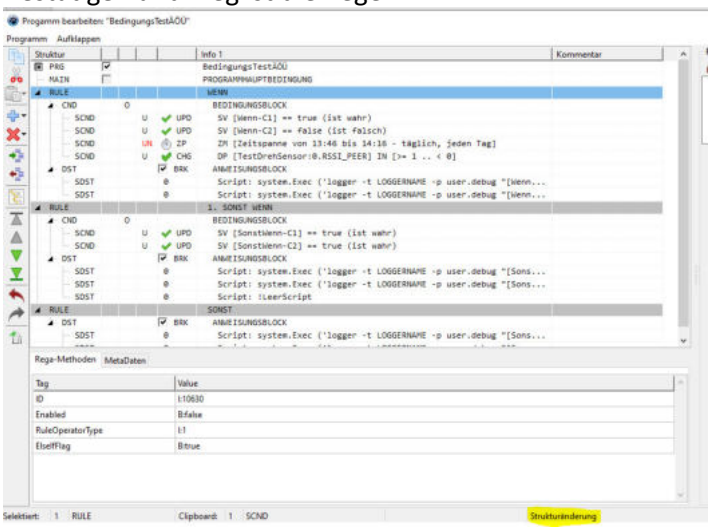
Vorgehensweise:
Das Programm im SDV als Arbeitsprogramm öffnen und die 1 -SonstWenn Bedingung selektieren (erkennt man schnell wieder an der leeren Aktivität, heisst keine Single Destinations wie in der WebUI)



Das rote Kreuz oder Ctrl-Del öffnet das Bestätigungs Menü:

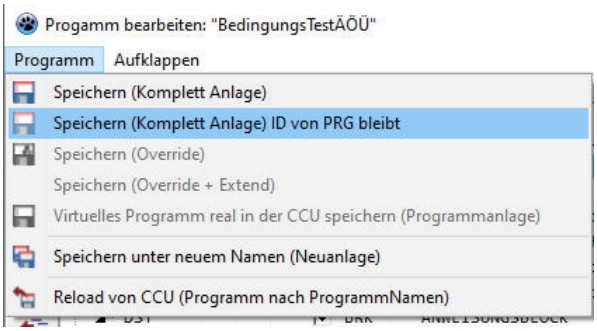


Bestätigen und weg ist die Regel:

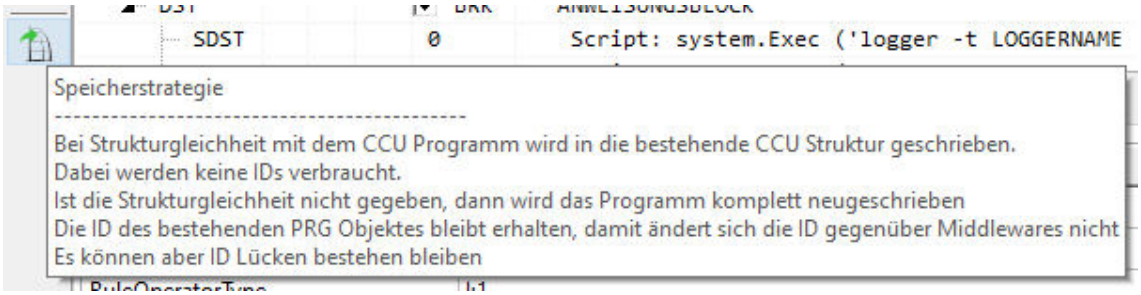


Wir sehen unten auch: es ist eine Strukturänderung. Diese Änderung könnte ich jetzt mit Undo auch wieder zurücknehmen. Wollen wir aber nicht. Wichtig: Der SDV Programmeditor arbeitet ja immer im Speicher des PC. Wir haben nun in der Baumstruktur diese Regel eliminiert, diese muss aber noch auf die CCU. Solange wir nicht auf Speichern gedrückt haben, haben wir in der CCU noch nichts geändert oder kaputt gemacht.

Also Speichern: oben auf Programm zeigt uns schon die eingeschränkten Möglichkeiten bei einer Strukturänderung, hier ist nur Neuanlage, kein Override möglich (siehe 11.1.5 Speicherstrategien)

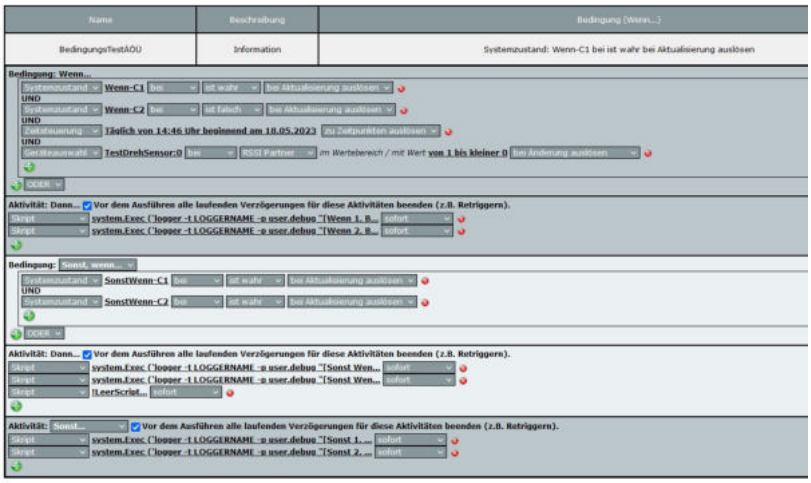


Da die Upload Taste sich im Setup des SDV mit der entsprechenden Strategie einstellen lässt (und vorab beim ersten Mal die von mir empfohlene Strategie schon voreingestellt hat, lässt ich mit der Maus via HINT kontrollieren)



Schreibt ein Druck auf die Upload Taste das Programm dann neu in die CCU. Im Gegensatz zur WebUI wird dieses aber jedes Mal sauber neu angelegt und alle alten Reste entfernt, so dass das berühmte Kaputteditieren von Programmen so nicht vorkommen kann.

Und dann nach dem Schreiben des Programmes sieht die WebUI wie gewollt so aus:



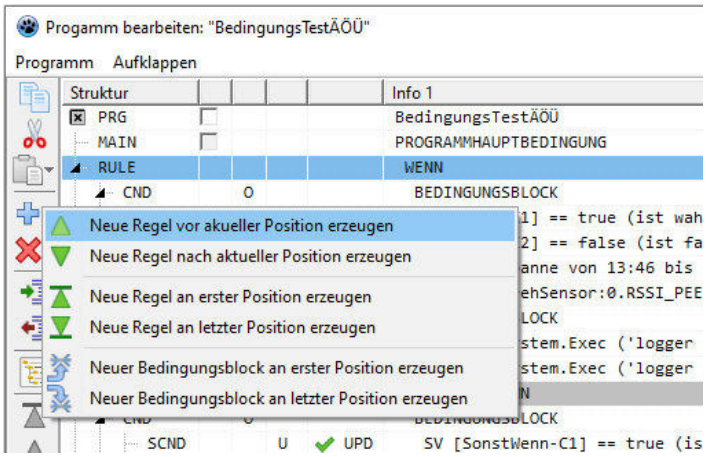
Das Ganze dauerte ein paar Sekunden, da braucht die WebUI schon länger beim Handeditieren um den Auswahldialog einer Systemvariablen aufzuklappen. Dafür würde jetzt jede weitere Bearbeitung keine IseID Lücken entstehen lassen.

11.5.2 Einfügen einer neuen Regel vor dem Wenn

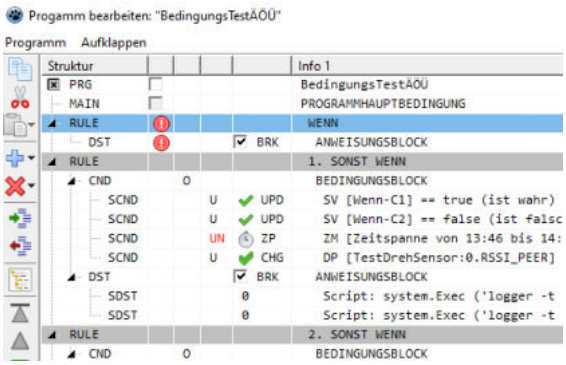
Es soll eine neue Regel vor dem Wenn eingefügt werden, aus dem bestehenden Wenn soll dann das erste Sonst Wenn werden.

Ich wünsche mit der WebUI dann schon mal viel Spass ^^

Im SDV Programmeditor Arbeitsprogramm definieren, erste Regel anklicken, dann entweder den + Button oder CTRL+PlusTaste für Erzeugen

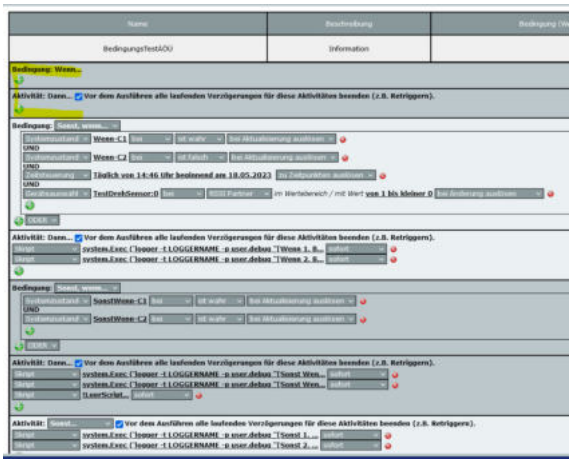


Neue Regel vor aktueller Position bzw. Neue Regel an erster Position werden eine neue erste Regel erzeugen.



Es existiert nun ein neues leeres Wenn an erster Position, das alte Wenn ist zum 1. SonstWenn geworden. Dies kann nun im SDV-Programmeditor oder in der WebUI bearbeitet werden. Wieder Upload in die CCU und wir haben ein neues, leeres Wenn. Das Ganze dauerte auch wieder nur wenige Sekunden.

Und das Ergebnis wie erwartet in der WebUI

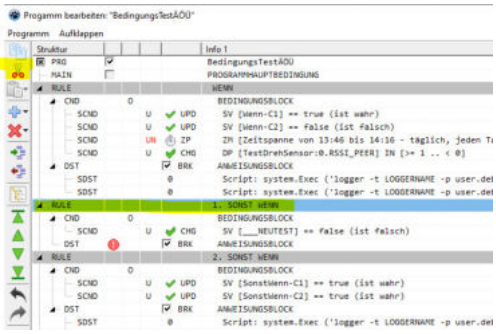


11.5.3 Vertauschen des Wenn und des 1. Sonst-Wenn Regelblocks

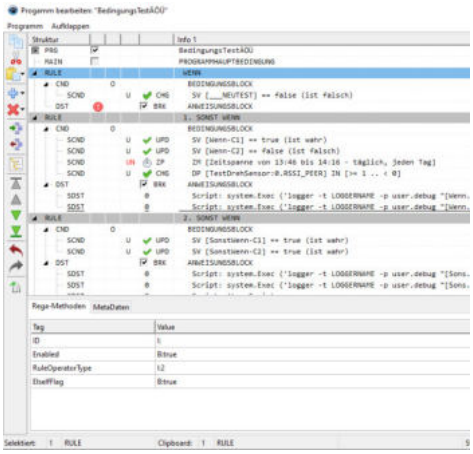
Aus dem 1. Beispiel soll die Wenn Regel mit der 1. Sonstwenn Regel vertauscht werden. Es bieten sich 2 Lösungsansätze an.

1. Ausschneiden der Kompletten 1. Sonst Wenn Regel
   Einfügen der Sonst-Wenn Regel aus der Zwischenablage vor der Wenn Regel
2. Drag/Drop der Sonstwenn Regel und Ablegen vor der Wenn Regel
3. Verschieben der Regel via Button

Los geht's. Lösung mit Ausschneiden:



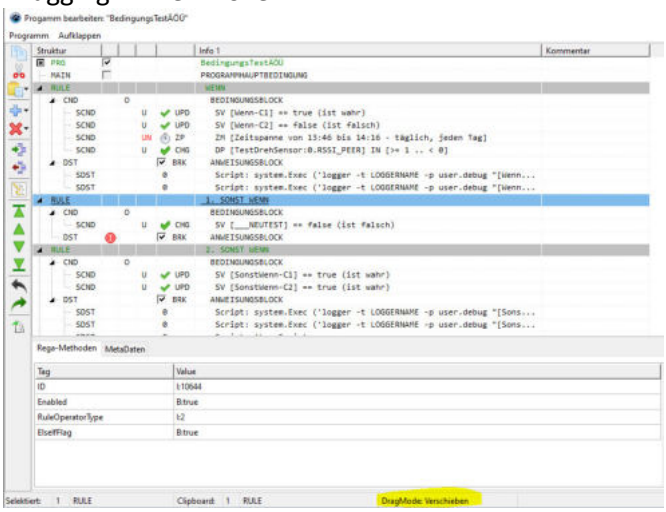
Dann die Wenn Bedingung anklicken, Einfügen anwählen vor der angewählten Position. Unten sehen wir, im Clipboard liegt die Rule, die wir eben ausgeschnitten haben. Also nun Einfügen.
Und wir sehen das Ergebnis, die Regel liegt nun vor dem alten Wenn



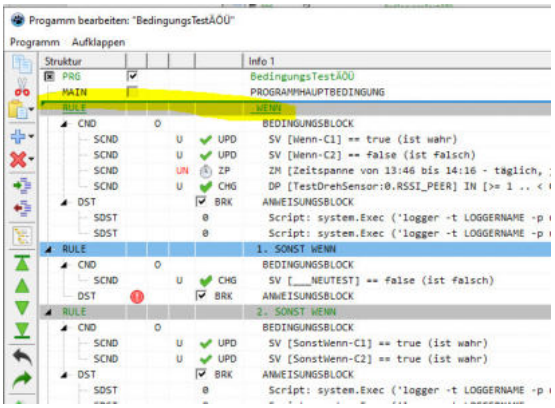
Nach dem Übertragen ist nun das geänderte Programm in der CCU.

Ansatz 2: Drag Drop

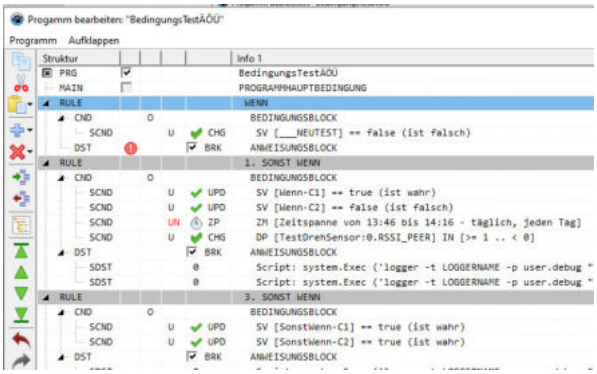
Die zu verschiebende Rule (das 1. Sonst wenn auswählen), rechte Maustaste, die berühmte Sekunde gedrückt halten, und wir sehen: DragMode: verschieben. Innerhalb des Arbeitsprogrammes ist Dragging immer Move.



Nun noch die Rule auf das erste, auch grün dargestellte Wenn schieben, dabei auf den oberen Teil des Rahmens für vor der Position achten



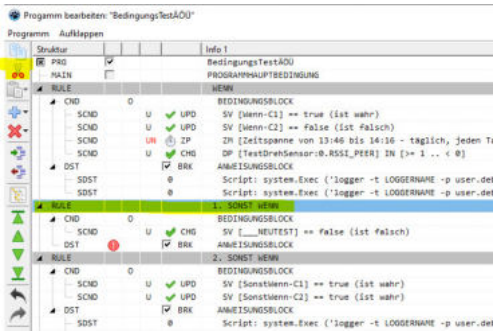
Und fertig:



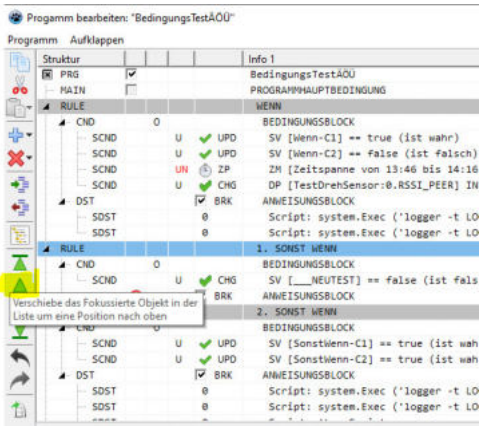
Nach dem Übertragen ist nun das geänderte Programm in der CCU.

Zum Schluss noch Verschieben über Buttons

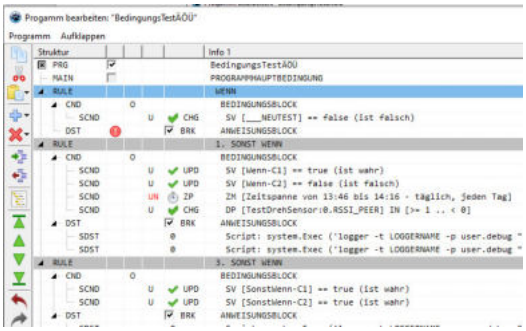
Wir wählen wieder die 2. Regel



Nun aber den Button „ein Element nach oben verschieben“



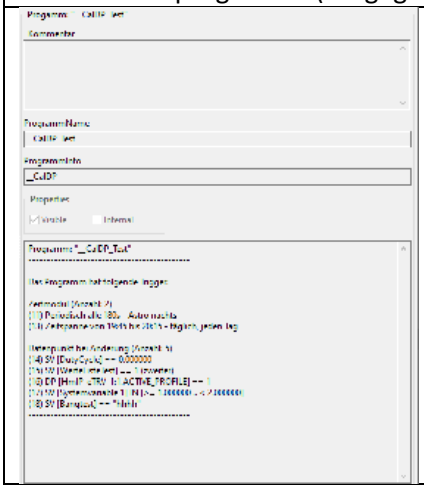
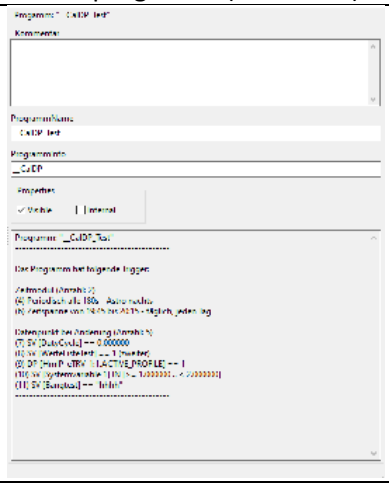
Ergibt dann durch einen Tastendruck das gewünschte Ergebnis



Nach dem Übertragen ist nun das geänderte Programm in der CCU.

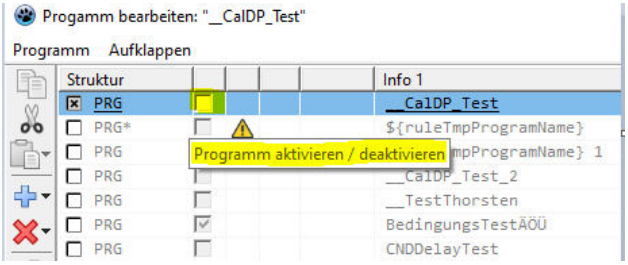
11.6 Anzeigen bzw. Ändern von Programmen oder Programmunterobjekten

In einem definierten Arbeitsprogramm können Werte geändert werden. Ist ein Programm nicht als Arbeitsprogramm definiert, so werden zwar alle Werte angezeigt, diese werden aber ausgegraut dargestellt.

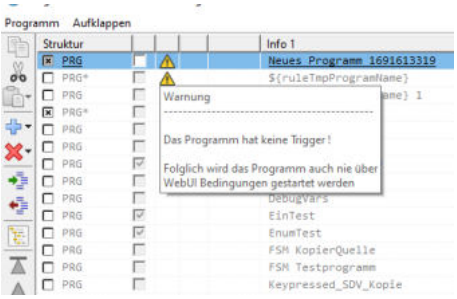
Nicht Arbeitsprogramm: (ausgegraut)	Arbeitsprogramm (editierbar)
	

11.6.1 Programmobjekt (PRG)

Veränderbar beim Programmobjekt ist der Kommentar, der Name und die ProgrammInfo sowie die Eigenschaften Visible und Internal im linken Feld. Aktive wird geändert direkt im Kasten im Programm Bereich des SDV-Programmeditors

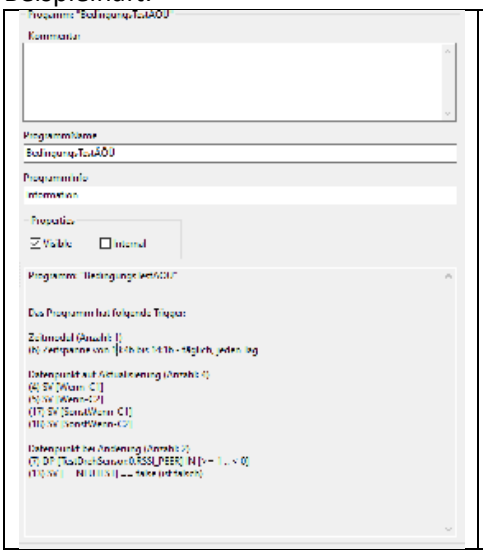


Der Editor macht auch gleichzeitig eine Analyse. Durch Darstellen des gelben oder des roten Ausrufezeichens macht der Editor auf unpassende Angaben aufmerksam.
Bei einem neuen Programm fehlen natürlich Trigger, folglich erscheint da im Hint:



Bei einem Programm wird im Ein/Ausgabefenster auch gleichzeitig dargestellt, welche Einzelnen Bedingungen das Triggern des Programmes und damit das logische Überprüfe der einzelnen Regeln auslösen.

Beispielhaft:



Hier in dem Programm die Trigger

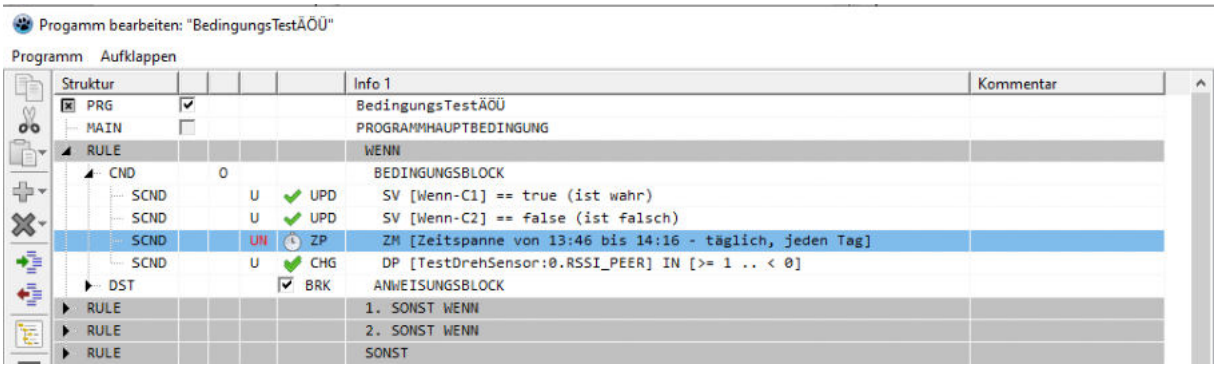
-1 Zeitmodul

-4 Datenpunkte, die auf aktualisierung Triggern

-2 Datenpunkte, die bei Änderung triggern (mit der Angabe des Bereichs

Wenn ich nun die genaue Bedingung und deren Einstellungen wissen will, kann ich im Programmteil suchen, es geht aber auch elegant schneller, Doppelklick auf die Zeile im Ein/Ausgabe Bereich und der passende Teil wird im Programm Vereich aufgeklappt und der Cursor spring an die Bedingung:

Hier exemplarisch: ich will ins Zeitmodul, also Doppelklick auf die Zeitspanne



Im Ein Ausgabebereich stehen dann die genauen Informationen zu der angezeigten Singlecondition, in diesem Fall das Zeitmodul. Dazu aber später mehr.

11.6.2 Programmhauptbedingung (Main-Condition)

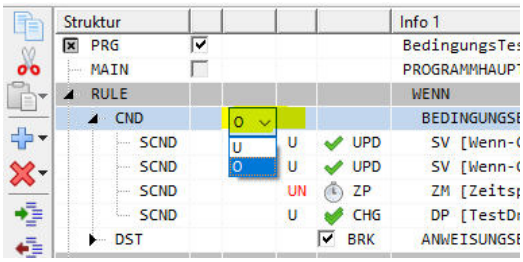
Die WebUI kann mit einer Main Condition nichts anfangen. Nichts desto trotz ist dieses ein extrem nützliches Instrument, um Programme einfacher und auch übersichtlicher zu gestalten. Der Main-Condition ist ein eigenes Kapitel gewidmet. In einer Main Condition kann nur der Kommentar geändert werden, erzeugen lassen sich nur einzelne Bedingungen (SCND). Aus Kompatibilitätsgründen lässt sich im Setup die Main-Condition komplett ausblenden.

11.6.3 Regelblock

In einer Regel kann nur der Kommentar vergeben werden. Die DST in einer Regel kann nicht gelöscht oder verschoben werden. In einem Regelblock lassen sich Bedingungsblöcke erzeugen.

11.6.4 Bedingungsblock

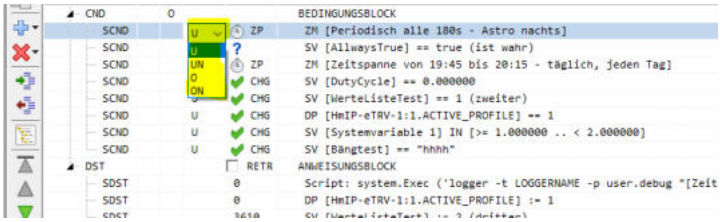
Der Bedingungsblock ist Teil seiner Regel und enthält eine oder mehrere Bedingungen (Ausnahme: die Sonst Regel enthält keine Bedingungsblöcke und auch keine Bedingungen. Von einem Bedingungsblock kann die Logische Verknüpfung eingestellt und ein Kommentar eingegeben werden.
Dazu in der Spalte mit der Logischen Verknüpfung klicken, die Spalte verbreitert sich automatisch und das Auswahlmenü für die Verknüpfung erscheint.



11.6.5 Die Einzel-Bedingung

Die Einzelbedingung ist Teil einer Bedingung. Von einer Einzelbedingung kann wie gewohnt der Kommentar geändert werden sowie die Art der Logischen Verknüpfung bzw. Die Trigger / Prüfbedingung.

11.6.5.1 Logische Verknüpfung



Die Schreibweise stammt aus der SPS-AWL Programmierung. Seit einigen Firmware-Versionen hat die WebUI leider eine „Auffälligkeit“. Ob Absichtlich oder nicht entstanden weiß ich nicht. Einer der Vorteile dieser Notation ist, dass ich die Logik frei mischen kann. Das funktioniert auch und die WebUI kann es auch richtig anzeigen. ABER. Öffne ich ein Programm mit der erweiterten Logik des SDV-Programmeditors in der WebUI, so wird diese Logik zwar richtig dargestellt, aber – beim Schreiben wird völlig unnütz eigentlich bei allen SCNDs die Verknüpfung der ersten SCND eingetragen. Halt nur das, was die „dumme“ WebUI kann. Es ist jetzt die Entscheidung des Anwenders: Ich will die WebUI jederzeit nutzen können, dann Finger weg von der freien Notation. Wenn ich allerdings den SDV-Programmeditor als Mastereditor nutze, so merkt sich dieser „seine“ Logik. Wenn die WebUI die dann beim Speichern zerstört hat, reicht es das Programm wieder im SDV-Programmeditor zu öffnen und einfach via Override zu speichern, dann wird wieder die „gemerkte“ SDV-Editorlogik in die Rega geschrieben. Solange das nicht gepatcht ist in der WebUI geht’s leider nicht anders.

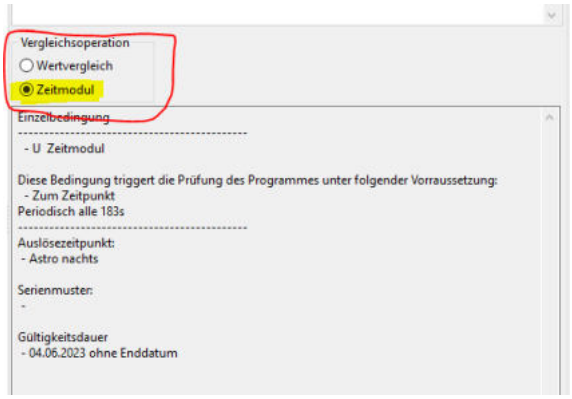
11.6.5.2 Triggerungsarten bzw. nur Prüfen

Nur prüfen:

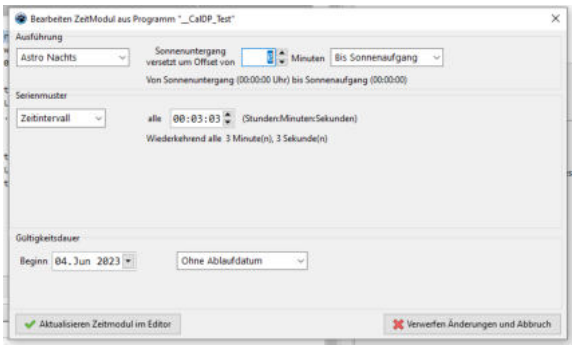
	NP	Nur Prüfen triggert kein Programm, sondern diese Bedingung wird einzig und alleine beim Logiktest geprüft.
	ZM	Zeitmodul zu Zeitpunkt (Programmauswertung wird getriggert)
	UPD	Bei Aktualisierung (Update). Ein neues Eintreffen eines Wertes führt zur Triggerung der Programmauswertung
	CHG	Bei Änderung (Change): Ein neu eingetroffener Wert muss die Logische Bedingung erfüllen, in der vorherigen Auswertung aber diese noch nicht erfüllt haben.
	IMP	Bei Tastendruck (Nur bei Longpressed oder Shortpressed)

11.6.5.3 Zeitmodul

Eine Einzelbedingung kann entweder ein Zeitmodul sein oder eine Vergleichsoperation.
Die Umschaltung wird in dem Auswahlmenü getätigt



In dem Informationsfenster werden die Daten zu dem Zeitmodul dargestellt. Geändert wird durch Doppelklick in das Zeitmodul-Informationsfenster oder aber durch Doppelklick in die Info Zelle des Zeitmoduls im Programmeditor.



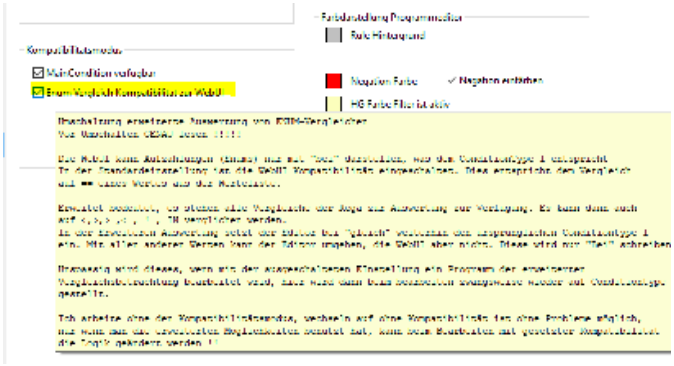
Das Menü entspricht dem Änderungsmenü aus dem Inspektor unter 4.10.1

11.6.5.4 Die normale Einzelbedingung (Variablenvergleich)

Ein Vergleich findet mit einem Gerätedatenpunkt (DP), einer Systemvariablen oder einer Alarmvariablen statt. Die Linke Seite bei dem Vergleich ist also immer eine Variable, hier abgekürzt Source genannt. Je nach Typ des Source ergeben sich folgende Vergleichsmöglichkeiten.

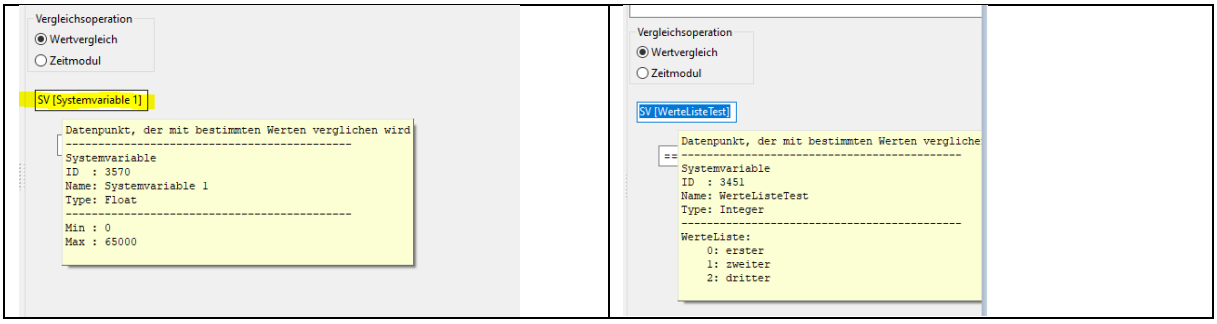
Source ändern:
Doppelklick auf die Sourcevariable öffnet das Datenpunkt-Auswahlmenüfenster. Es werden die Datenpunkte vorgeschlagen, die an dieser Stelle auch erlaubt sind (read bzw. event). Im Gegensatz zur WebUI ändert der SDV-Skripteditor, wenn nachträglich der Typ geändert wird, den Typ der Vergleichswerte automatisch richtig mit.

Tastendruck (kurz oder lang): Keine weitere Angabe nötig
String: vergleich nur auf gleich
Enum: (Ein Integer mit zugeordneter Werteliste)
Hier ist es ein bisschen komplizierter, es kann ein Kompatibilitätsmodus zur WebUI eingestellt werden, hierbei ist wie in der WebUI nur == (bei) möglich. Die Rega beherrscht allerdings den vollen integer Vergleich. Nachteil, die WebUI kann es nicht darstellen.

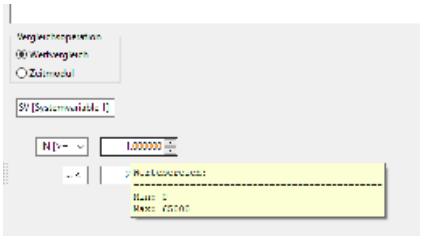


Integer/real: voller Vergleich

Der Hint auf die SourceVariable zeigt Einzelheiten zu der Variable an:



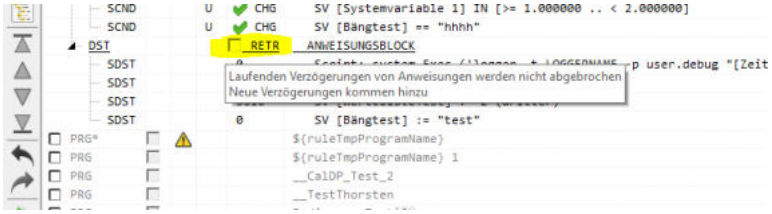
Der Hint auf den Vergleichswert zeigt den gültigen Wertebereich an



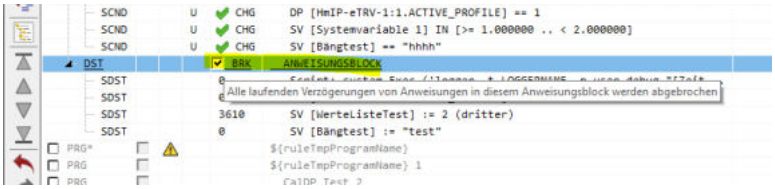
11.6.6 Anweisungsblock

Im Anweisungsblock wird festgelegt, ob die unter dem Anweisungsblock angelegten Anweisungen retriggert werden oder nicht (siehe auch Handbuch zur WebUI)

Dies wird im Editorfeld eingestellt



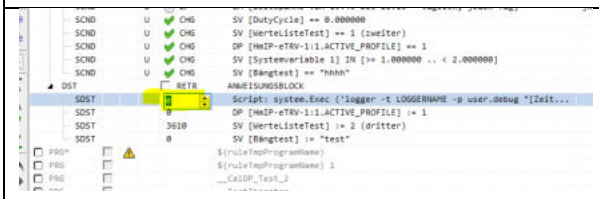
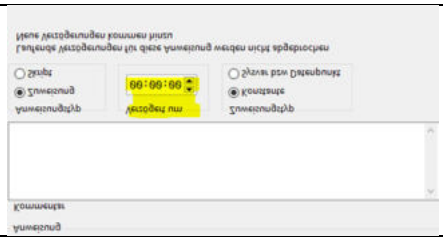
Bzw.



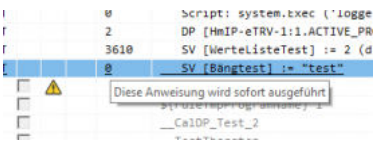
Als Gedankenstütze wird dieses im Hint von einer Einzelanweisung und auch im Eingabebereich einer Einzelanweisung dargestellt.

11.6.6 Die einzelne Anweisung

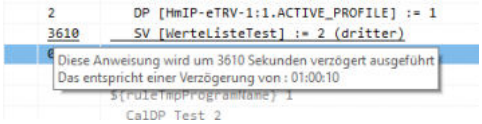
Die einzelne Anweisung (SDST für Single Destination). Eine Einzelanweisung kann ein Script sein oder eine Zuweisung. Beiden gemeinsam ist die möglichkeit, die Ausführung zu verzögern

Einstellbarkeit im Editorfeld	Einstellbarkeit im Datenfeld
	

Im Hint werden über dem Editorfeld auch die Hinweise zu der Verzögerung angezeigt:

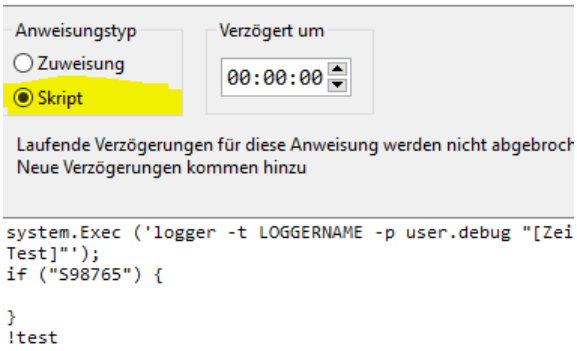


Bzw:



11.6.6.1 Script

Ein Script wird in einer Einzelanweisung hier angewählt:

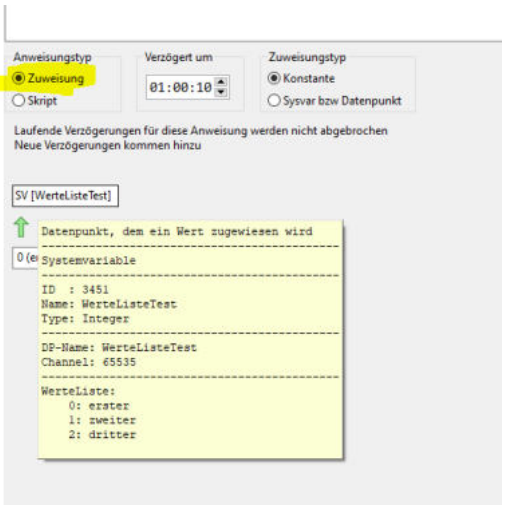


#

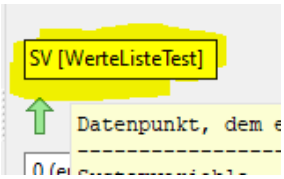
Im Anzeigefeld wird das Skript als Text dargestellt. Eine schnelle Änderung kann in dem Anzeigefeld vorgenommen werden. Hier natürlich ohne die weitreichenden Möglichkeiten des SDV-Skripteditors. Soll der Editor genutzt werden, so reicht ein Doppelklick in das Anzeigefeld, und das Skript wird im Skripteditor des SDV zur Bearbeitung bereitgestellt.
Wird der Skripteditor wieder verlassen, indem der Programmeditor geöffnet wird, so wird das Script des SDV-Programmeditors übernommen.
Soll dies nicht gewünscht sein → es gibt die Undo Taste.

11.6.6.2 Variablenzuweisung

Eine Variablenzuweisung weist, wie der Name schon sagt, einer Variablen eine Konstante oder den Wert einer anderen Variablen zu

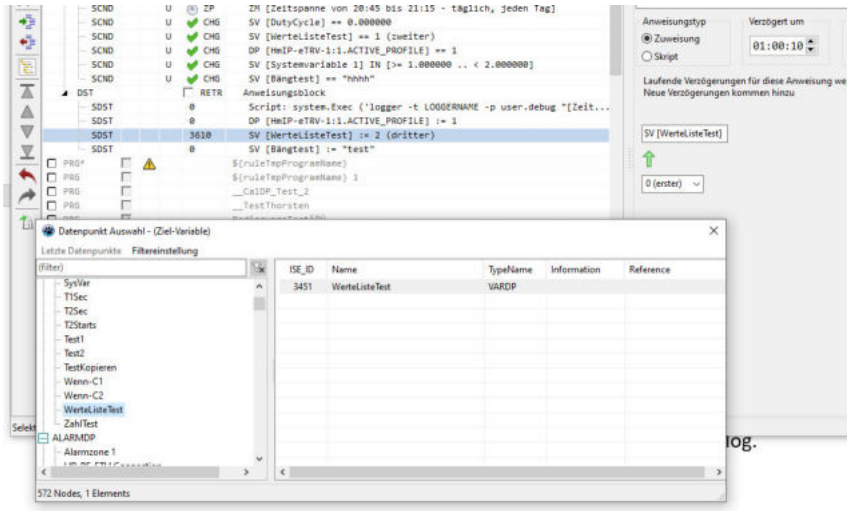


Der TargetVariablen , in dem Fall SystemVariable SV mit dem Namen WertelistenTest wird ein Wert , hier in dem Fall eine Konstante zugewiesen.

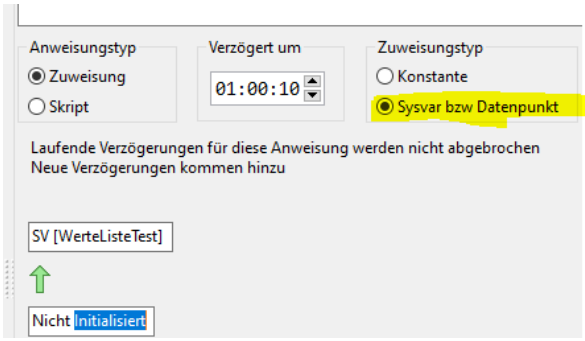


Im Hint werden Informationen über die Target Variablen angezeigt.

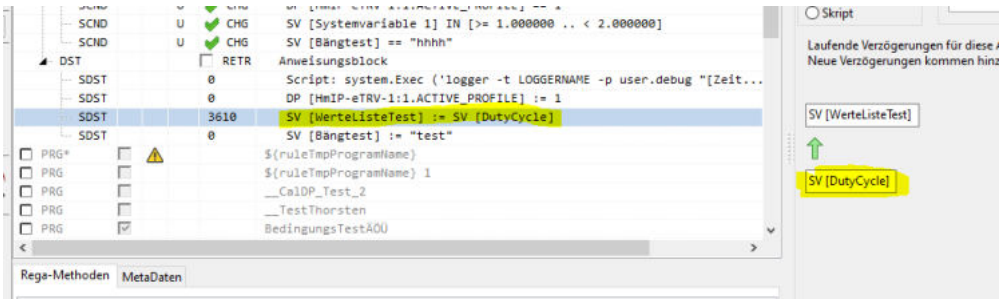
Ändern der Target Variablen (oder erste Zuweisung, wenn diese noch nicht definiert ist)
Doppelklick auf das Eingabefeld der Target Variablen öffnet den Auswahldialog.



Ebenso kann der Inhalt einer Variablen der Target Variablen zugewiesen werden



Dargestellt wird dieses so:



11.7 Die Maincondition

Die Maincondition ist eine Option in der Rega, die von der WebUI allerdings nicht genutzt wird. Der SDV konnte im Inspektor schon nach Programmen mit Mainconditions filtern, ebenso unterstützen die Backup Routinen des SDV die MainConditions (schon seit Jahren)

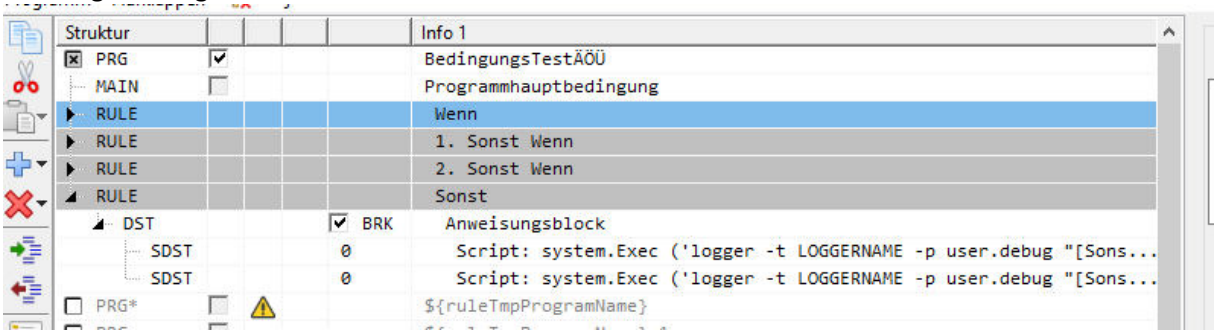
Was ist eine Maincondition ?
Eine Maincondition ist eine Art Hauptschalter, deren Bedingung erfüllt sein muss, damit das Programm überhaupt ausgeführt wird. Das bedeutet, das mit einer Maincondition auch die Ausführung des Sonst Teiles unterbunden wird. Damit ist zum Beispiel die komplette Unterdrückung der Ausführung bei Systemstart möglich.
Ebenso werden Programme kürzer und Übersichtlicher, wenn mehrere Regeln zum Einsatz kommen und immer Bedingungen geprüft werden müssen die sich so einfach in die Maincondition setzen lassen.

Warum unterstützt die WebUI das nicht ? Gute Frage… Nächste Frage

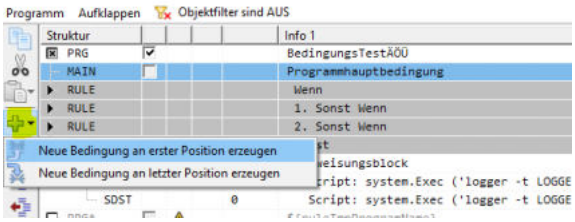
Kann die WebUI Mainconditions anzeigen ? Nein

Es gibt so eine goldene Regel: in einer Maincondition Bedingungen immer nur auf Nur Prüfen stellen.

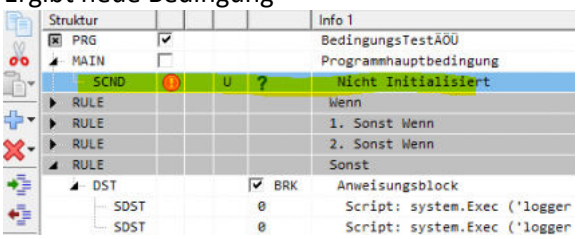
Kleines Beispiel für ein kleines Programm, welches bei Systemboot nicht ausgeführt werden soll und nur ausgeführt werden soll, wenn AllwaysTrue auch true ist.
Das Original Programm:



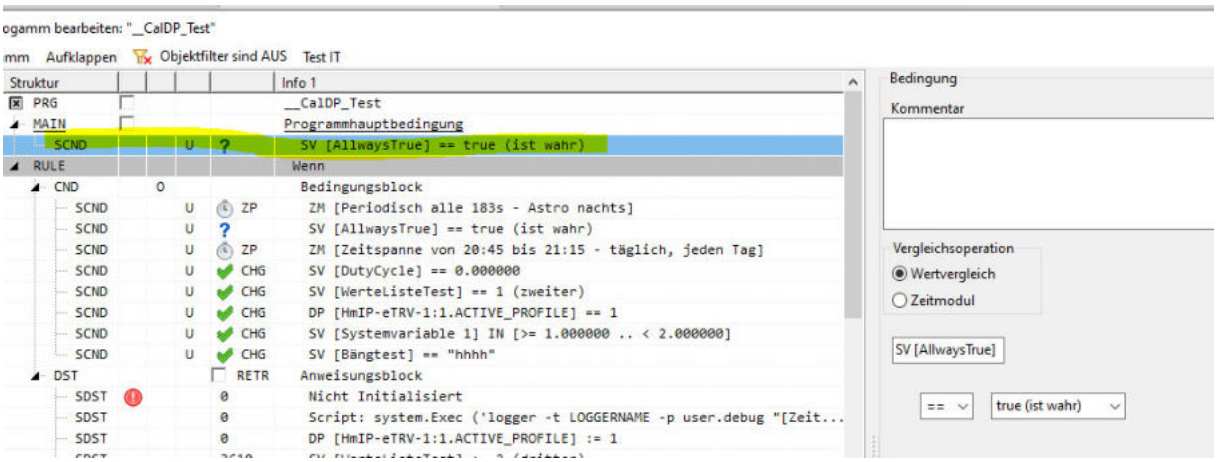
In den Wenn und Sonst wenn ließe sich die oben genannte Bedingung noch einbauen, im Sonst aber nicht
Also die Maincondition markieren, + Button und neue Bedingung anlegen



Ergibt neue Bedingung



Die neue Singlecondition müssen wir noch Logisch mit der Reboot Bedingung verknüpfen. Also in das nicht definiert doppelklicken und Allways True auswählen



12 License Disclaimer

Folgende Software wurde verwendet bei der Konzeptionierung und Programmierung des SDV

12.1 OpenSSl

Mit diesem Programm werden 2 DLLs des OpenSSL Project mitgeliefert

Original Lizenztext:

LICENSE ISSUES
==========

The OpenSSL toolkit stays under a dual license, i.e. both the conditions of
the OpenSSL License and the original SSLeay license apply to the toolkit.
See below for the actual license texts. Actually both licenses are BSD-style
Open Source licenses. In case of any license issues related to OpenSSL
please contact openssl-core@openssl.org.

OpenSSL License

```
* ====================================================================
* Copyright (c) 1998-2016 The OpenSSL Project.  All rights reserved.
*
* Redistribution and use in source and binary forms, with or without
* modification, are permitted provided that the following conditions
* are met:
*
* 1. Redistributions of source code must retain the above copyright
*    notice, this list of conditions and the following disclaimer.
*
* 2. Redistributions in binary form must reproduce the above copyright
*    notice, this list of conditions and the following disclaimer in
*    the documentation and/or other materials provided with the
*    distribution.
*
* 3. All advertising materials mentioning features or use of this
*    software must display the following acknowledgment:
*    "This product includes software developed by the OpenSSL Project
*    for use in the OpenSSL Toolkit. (http://www.openssl.org/)"
*
* 4. The names "OpenSSL Toolkit" and "OpenSSL Project" must not be used to
*    endorse or promote products derived from this software without
*    prior written permission. For written permission, please contact
*    openssl-core@openssl.org.
*
* 5. Products derived from this software may not be called "OpenSSL"
*    nor may "OpenSSL" appear in their names without prior written
*    permission of the OpenSSL Project.
*
* 6. Redistributions of any form whatsoever must retain the following
*    acknowledgment:
```

*   "This product includes software developed by the OpenSSL Project
*   for use in the OpenSSL Toolkit (http://www.openssl.org/)"
*
* THIS SOFTWARE IS PROVIDED BY THE OpenSSL PROJECT ``AS IS'' AND ANY
* EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
* IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
* PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE OpenSSL PROJECT OR
* ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
* SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT
* NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES;
* LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
* HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT,
* STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
* ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED
* OF THE POSSIBILITY OF SUCH DAMAGE.
* ==
*
* This product includes cryptographic software written by Eric Young
* (eay@cryptsoft.com). This product includes software written by Tim
* Hudson (tjh@cryptsoft.com).
*
*
Original SSLeay License
 ----

* Copyright (C) 1995-1998 Eric Young (eay@cryptsoft.com)
* All rights reserved.
*
* This package is an SSL implementation written
* by Eric Young (eay@cryptsoft.com).
* The implementation was written so as to conform with Netscapes SSL.
*
* This library is free for commercial and non-commercial use as long as
* the following conditions are aheared to. The following conditions
* apply to all code found in this distribution, be it the RC4, RSA,
* lhash, DES, etc., code; not just the SSL code. The SSL documentation
* included with this distribution is covered by the same copyright terms
* except that the holder is Tim Hudson (tjh@cryptsoft.com).
*
* Copyright remains Eric Young's, and as such any Copyright notices in
* the code are not to be removed.
* If this package is used in a product, Eric Young should be given attribution
* as the author of the parts of the library used.
* This can be in the form of a textual message at program startup or
* in documentation (online or textual) provided with the package.
*
* Redistribution and use in source and binary forms, with or without
* modification, are permitted provided that the following conditions
* are met:
* 1. Redistributions of source code must retain the copyright
*   notice, this list of conditions and the following disclaimer.
* 2. Redistributions in binary form must reproduce the above copyright
*   notice, this list of conditions and the following disclaimer in the

```
*    documentation and/or other materials provided with the distribution.
* 3. All advertising materials mentioning features or use of this software
*    must display the following acknowledgement:
*    "This product includes cryptographic software written by
*     Eric Young (eay@cryptsoft.com)"
*    The word 'cryptographic' can be left out if the rouines from the library
*    being used are not cryptographic related :-).
* 4. If you include any Windows specific code (or a derivative thereof) from
*    the apps directory (application code) you must include an acknowledgement:
*    "This product includes software written by Tim Hudson (tjh@cryptsoft.com)"
*
* THIS SOFTWARE IS PROVIDED BY ERIC YOUNG ``AS IS'' AND
* ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
* IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
* ARE DISCLAIMED.  IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE
* FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
* DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
* OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
* HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
* LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
* OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
* SUCH DAMAGE.
*
* The licence and distribution terms for any publically available version or
* derivative of this code cannot be changed.  i.e. this code cannot simply be
* copied and put under another distribution licence
* [including the GNU Public Licence.]
*/
```

12.2 Synapse Ararat

Synapse TCP/IP and serial library
This project is freeware and open source under modified BSD style licence.

License

Each source file, what is protected by this license, have license text on the begin of file. Used license is very free for you. You can use it in any your application, include commercial. However follow next license statement, please!

International copyright law allows me to license you to use the Synapse code on these terms and conditions - if you do not comply with these terms and conditions then you cannot use the Synapse code!

If you use my libraries in commercial project, I will be very happy if you send me a some donation.
Thank you very much!

BDS style license

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:
Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
Neither the name of Lukas Gebauer nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.
THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS “AS IS” AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

12.3 Compiler und RAD

Programmiert wurde der SDV unter Lazarus unter Win64