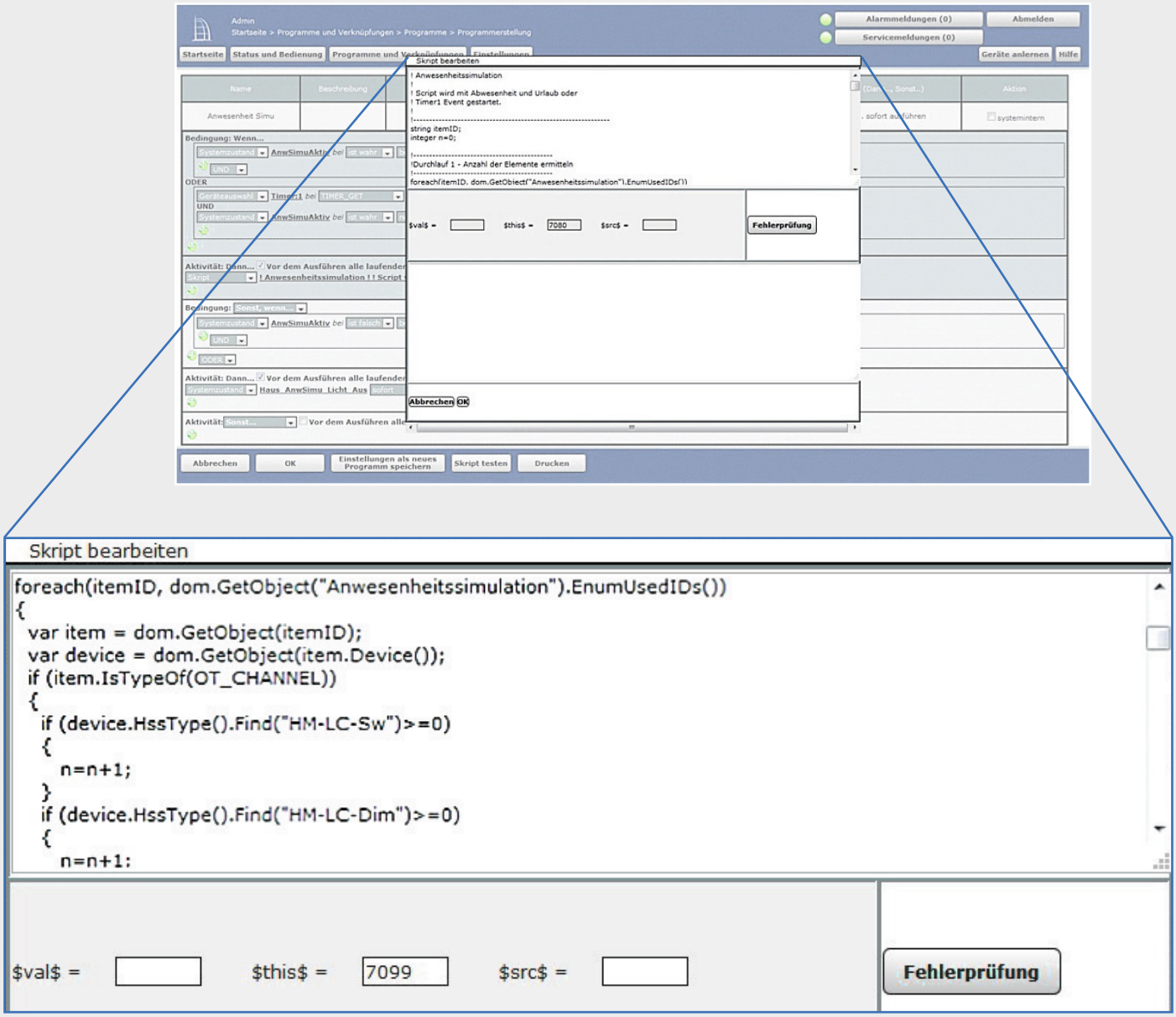
Homematic Scriptprogrammierung

Teil 11 – Sicherheit im und rund um das Haus

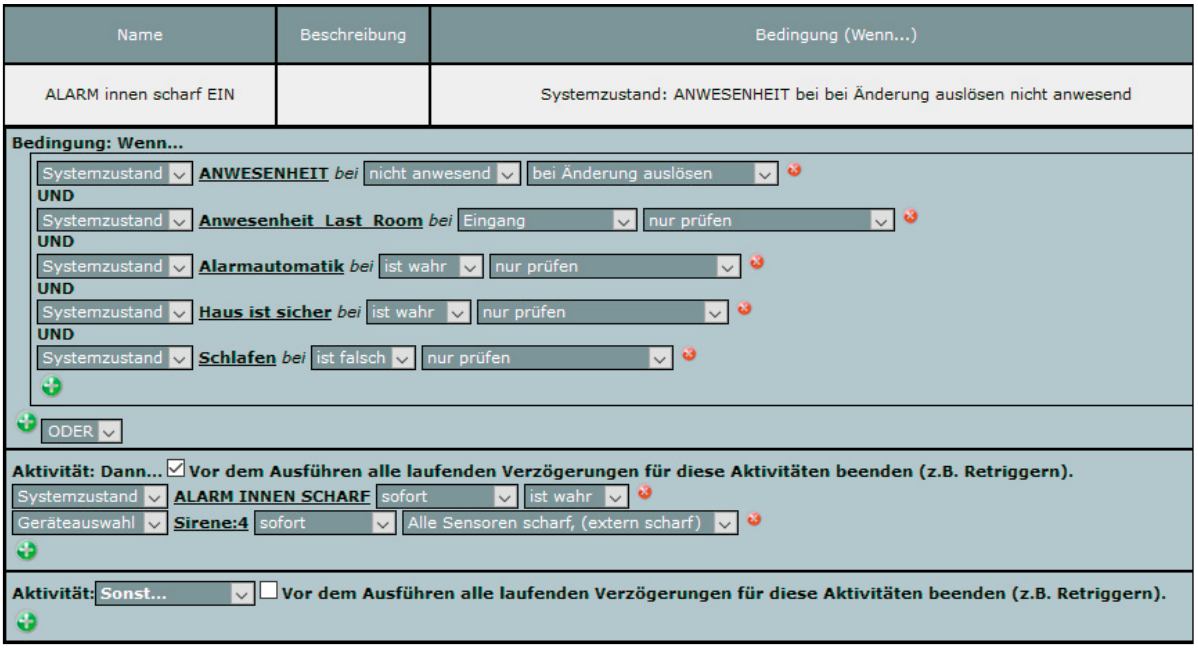


Im elften Teil der Artikelserie beschäftigen wir uns mit zwei Beispielen zum Thema Sicherheit im Haus.

Im letzten Teil der Serie haben wir Programme (Scripte) vorgestellt, mit deren Hilfe es möglich ist, eine Anwesenheit bzw. eine Abwesenheit im Haus (automatisch) festzustellen. Damit ist nun die Realisierung einer ganzen Reihe von Funktionen rund um die Sicherheit im Haus möglich.

Für die Konzeption der eigentlichen Alarmanlage bietet das Homematic System genügend Komponenten, es sind verschiedene Wege zur Realisierung möglich. So lassen sich z. B. an die Homematic Innensirene Sensoren sowohl direkt anlernen als auch über die Homematic CCU koppeln. Die Alarmanlagenfunktionen (scharf schalten, unscharf schalten, Alarm ...) lassen sich recht einfach über Zentralenprogramme realisieren, hier bietet die Scriptprogrammierung nicht unbedingt Vorteile.

Hier ein Beispiel für das Scharfschalten der Anlage:



Aber rund um die System-(Zentralen-)Variablen, die die An- oder Abwesenheit anzeigen, lassen sich weitere (automatische) Funktionen realisieren, die der Sicherheit dienen. Sinnvoll ist beispielsweise eine Übersicht bzw. eine Information darüber, ob alle Fenster und Türen, über die ein (unbefugtes) Betreten des Hauses möglich ist, tatsächlich beim Verlassen des Hauses geschlossen sind.

Neben der Anzeige über ein LED-Display oder eine andere der im Homematic Lieferprogramm erhältlichen Anzeigen bietet sich auch eine Sprachausgabe an, die mit dem Homematic MP3-Funk-Gong realisierbar ist.

Sprachausgabe: Ansage aller noch offenen sicherheitsrelevanten Fenster und Türen

Die Sprachausgabe erfolgt über einen MP3-Funk-Gong.

Voraussetzung für dieses Script ist ein Gewerk *Verschluss*. Alle Türen und Fenster, die beim Verlassen des Hauses verschlossen sein müssen und deren Zustand mithilfe dieses Scriptes angesagt werden sollen, werden diesem Gewerk zugeordnet.

Weiterhin sind für die eigentliche Sprachausgabe folgende System-(Zentralen-)Variablen notwendig:

`Room_Sound_List`	`Liste, in der den Räumen des Hauses Nummern (Soundnummern für den MP3-Funk-Gong) zugeordnet sind`
`Ger_Sound_List`	`Liste, in der den Geräten des Hauses Nummern (Soundnummern für den MP3-Funk-Gong) zugeordnet sind`

Hinweis:
Die genaue Funktionsweise zur Ermittlung der Soundfilenummern für die Räume und Geräte sowie der Formate der Listen dieser zwei Zentralenvariablen sind im Artikel 4 „Script: Verknüpfte Informationen" dieser Artikelserie beschrieben worden.

Das Script:

```
01  ! INFO AUDIO SICHERHEIT
02  !---------------------------------------
03
04  integer word_position = 0;
05  string daten = "";
06  string itemID;
07  integer counter = 0;
08  string daten;
09  string d_Raum;
10  string d_Geraet;
11  string d_Nummer;
12
13  !***************************************
14
15  d_Raum = "";
16  d_Geraet = "";
17  d_Nummer = "";
18  daten = "";
19  counter = 0;
20
21  !-----------------------------------------------
22  !Elemente aus dem Gewerk Verschluss
23  !-----------------------------------------------
24  var myAssembly = dom.GetObject("Verschluss");
25  string sep="";
26
27  !-----------------------------------------------------
28  ! Folgende Tueren und Fenster sind noch offen
29  !-----------------------------------------------------
30  string command = "1,1,108000,125";
31
32  foreach(itemID,myAssembly.EnumUsedIDs())
33  {
34      var item = dom.GetObject(itemID);
35      if(item.IsTypeOf(OT_CHANNEL))
36      {
37          var dev = dom.GetObject(item.Device());
38          if ((dev.HssType()=="HM-Sec-RHS") || (dev.HssType()=="HM-
    Sec-SC") || (dev.HssType()=="HM-Sec-SCo"))
39          {
40              !----------
41              !Geoeffnet
42              !----------
43              if (item.DPByHssDP("STATE").State()<> 0)
44              {
45                  counter = counter + 1;
46                  daten = item.Name();
47                  word_position = daten.Find("*");
48                  if (word_position>0)
49                  {
50                      d_Raum = daten.Substr(0, word_position);
51                      daten =
    daten.Substr(word_position+1,daten.Length()- word_position);
52                      word_position = daten.Find("*");
53                      if ((daten.Find("*")>0) && (counter < 4))
54                      {
55                          d_Geraet = daten.Substr(0,word_position);
56                          daten =
    daten.Substr(word_position+1,daten.Length()-word_position-4);
57                          if (daten.Find(":")>0) {
58                              daten = daten.Substr(0, daten.Find(":"));
59                          }
60                          d_Nummer = daten;
61                      }
62                  !-------------------------------
63                  ! Textnummer fuer Raum
64                  !-------------------------------
65                  daten = dom.GetObject("Room_Sound_List").State();
66                  word_position = daten.Find(d_Raum);
67                  !-----------------------------------------
68                  ! einfuegen in den Sprachausgabestring command
```

```
69                      !-----------------------------------------
70                      if (word_position>=0)
71                      {
72                          daten=
73                          word_position = daten.Find("*");
74                          command = command # "," #
75                      }
76                      !--------------------------------
77                      ! Textnummer fuer Geraet
78                      !--------------------------------
79                    daten = dom.GetObject("Ger_Sound_List").State();
70                    word_position = daten.Find(d_Geraet);
71                    if (word_position>0)
72                    {
73                          daten=
74                          word_position = daten.Find("*");
75                          command = command # "," #
76                    }
77                    !-----------------------------------
78                    ! Geraetenummer
79                    !-----------------------------------
80                    command = command # "," # d_Nummer;
81                }
82            }
83        }
84    }
85  }
86  if (counter > 0) {
87  }
88  else
89  {
90      string command = "1,1,108000,78";
91  }
92
93  !------------------
94  ! Ausgabe
95  !------------------
96  dom.GetObject("FunkGong2:1").DPByHssDP("SUBMIT").State(command)
```

Der Text mit der Textnummer 125 (Zeile 30) ist: *„Folgende sicherheitsrelevante Fenster oder Türen sind noch offen."*

Der Text mit der Textnummer 78 (Zeile 90) ist: *„Alle sicherheitsrelevanten Fenster und Türen sind geschlossen."*

Die Texte für die Geräte (Fensterschalter → z. B. Fenster, Türgriffschalter → z. B. Türe) und für die Räume finden sich in den oben angesprochenen Zentralenvariablen.

Nach der Deklaration der Scriptvariablen (Zeilen 4 … 11) und deren Initialisierung (Zeilen 15 … 19) werden in einer Schleife alle Sensoren des Gewerkes *Verschluss* durchgegangen (Zeilen 32 … 85). Wird ein Sensor gefunden, der *offen* meldet, so wird in den Zeilen 46 … 60 der Name des Raumes, der des Gerätes und eine fortlaufende Nummer gelesen, mit denen dann in einer Dekodierung aus den Zentralenvariablen *Room_Sound_List* und *Ger_Sound_List* die zugehörigen Soundnummern ermittelt werden. Die Nummern, die so in den Durchläufen ermittelt werden, werden aneinandergereiht, sodass eine Liste der gewünschten Elemente angesagt wird.

Über ein Zentralenprogramm kann nun dieses Script z. B. mit Drücken eines Tasters oder Öffnen der Haustür gestartet werden.

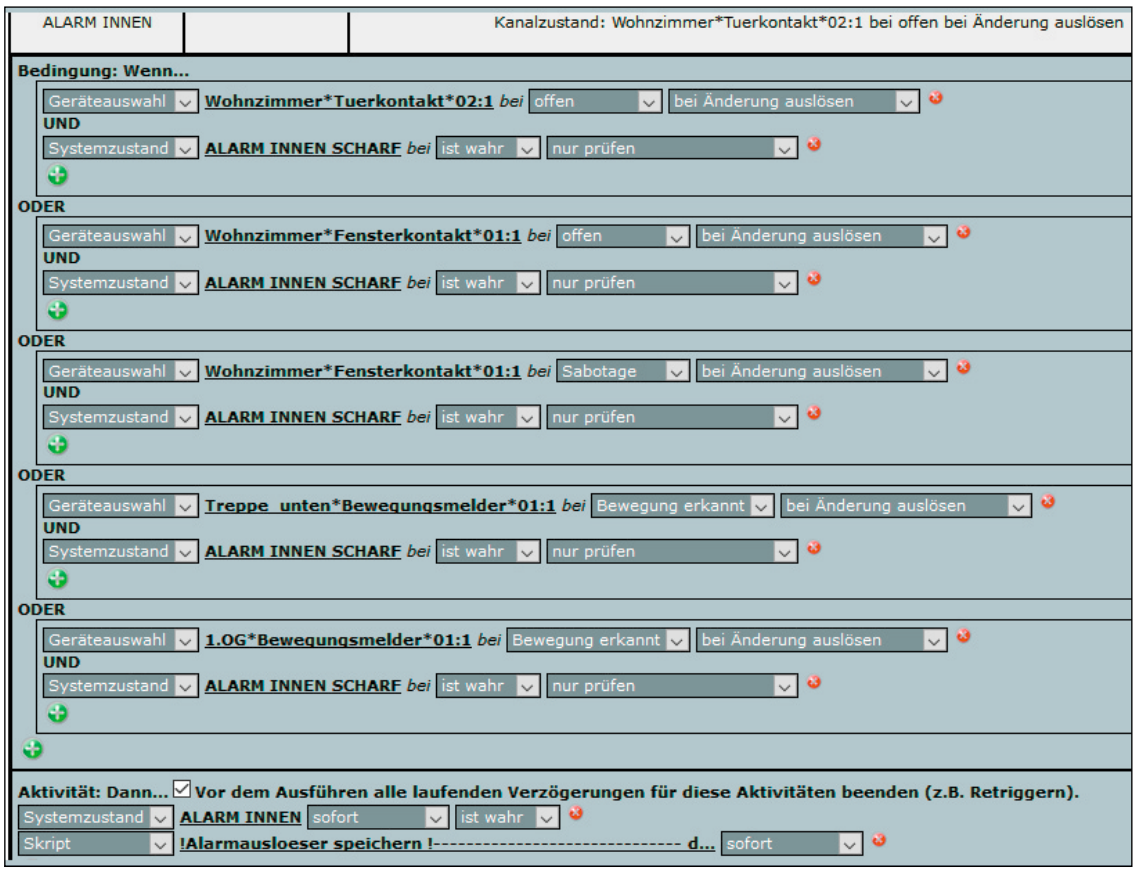
Speichern des Alarmauslösers

Das folgende kurze Script bietet die Möglichkeit, bei einem Alarm den Alarmauslöser in einer Zentralenvariablen (Alarmauslöser, Textvariable) zu speichern:

```
01  !Alarmausloeser speichern
02  !--------------------------------
03  dom.GetObject("Alarmausloeser").State(dom.GetObject((dom.GetObject((dom.GetObject("
    $src$")).Channel())).Device()));
```

Dieses Script muss im Auslösezweig für den Alarm aufgerufen werden.

Beispiel:

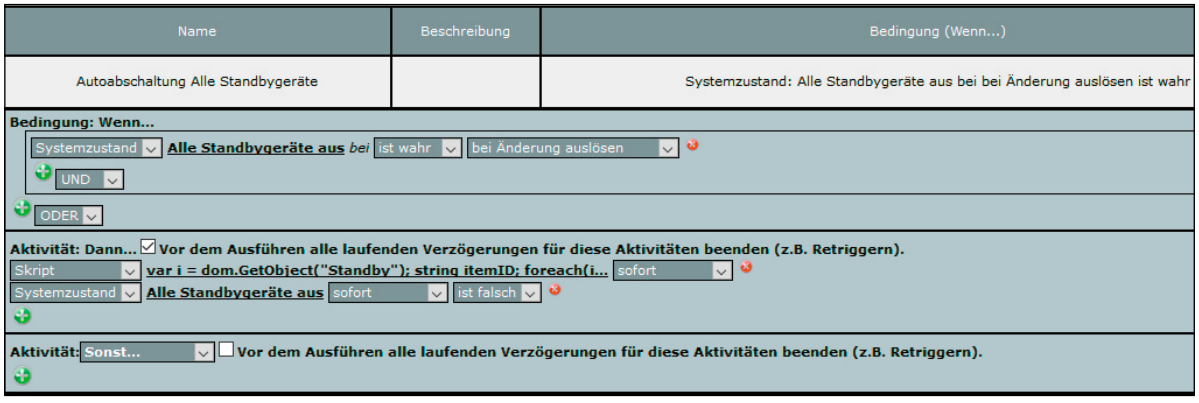


Abschalten aller Stand-by-Geräte beim Verlassen des Hauses

Die Homematic CCU bietet durch die Gewerke viele Möglichkeiten. So lassen sich bei Zuordnung von Aktoren zu einem Gewerk *Stand-by* beim Verlassen des Hauses alle Stand-by-Geräte (alle Geräte, die dem Gewerk *Stand-by* zugeordnet sind) sehr einfach über die Zentralenvariable *Anwesenheit* bzw. *Abwesenheit* abschalten.

```
01  ! Alle Geraete des Gewerkes Standby abschalten
02  !-------------------------------------------
03  var i = dom.GetObject("Standby");
04
05  string itemID;
06
07  foreach(itemID, i.EnumUsedIDs())
08  {
09    var item = dom.GetObject(itemID);
10    if (item.IsTypeOf(OT_CHANNEL))
11    {
12      var device = dom.GetObject(item.Device());
13      item.State(false);
14    }
15  }
```

Das Starten des Scriptes kann direkt über die Zentralenvariable *Anwesenheit* bzw. *Abwesenheit* erfolgen, sinnvoller ist aber, solche Scripte über eigene (Start-)Variablen zu starten.



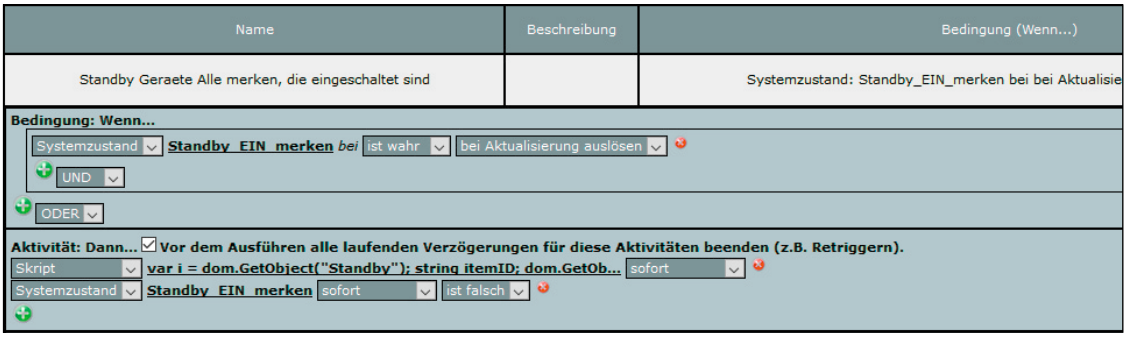
Die Zentralenvariable *Alle_Standbygeraete_aus* (TRUE) startet das Script, gleichzeitig wird sie wieder auf FALSE zurückgesetzt.

Will man sichergehen, dass beim Betreten des Hauses dann auch lediglich diejenigen Stand-by-Geräte wieder eingeschaltet werden, die beim Verlassen des Hauses auch eingeschaltet waren, dann muss sich die CCU diese Geräte „merken". Dies kann über folgendes Script realisiert werden:

```
! Alle eingeschalteten Geraete des Gewerkes Standby merken
!---------------------------------------------
var i = dom.GetObject("Standby");

string itemID;

foreach(itemID, i.EnumUsedIDs())
{
  var item = dom.GetObject(itemID);
  if (item.IsTypeOf(OT_CHANNEL))
  {
    var device = dom.GetObject(item.Device());
    item.State(false);
  }
    !wenn eingeschaltet, dann Seriennummer in Liste
    !speichern.
    !-------------------------------------------
    if (item.State()<>0)
    {
        if (dom.GetObject("War_eingeschaltet").State() <> "")
        {

dom.GetObject("War_eingeschaltet").State(dom.GetObject("War_ein
geschaltet").State()+",");

        }

dom.GetObject("War_eingeschaltet").State(dom.GetObject("War_ein
geschaltet").State()+sernr );

    }
  }
}
```

Der Inhalt der System-(Zentralen-)Variablen *War_eingeschaltet* sieht dann beispielsweise so aus:

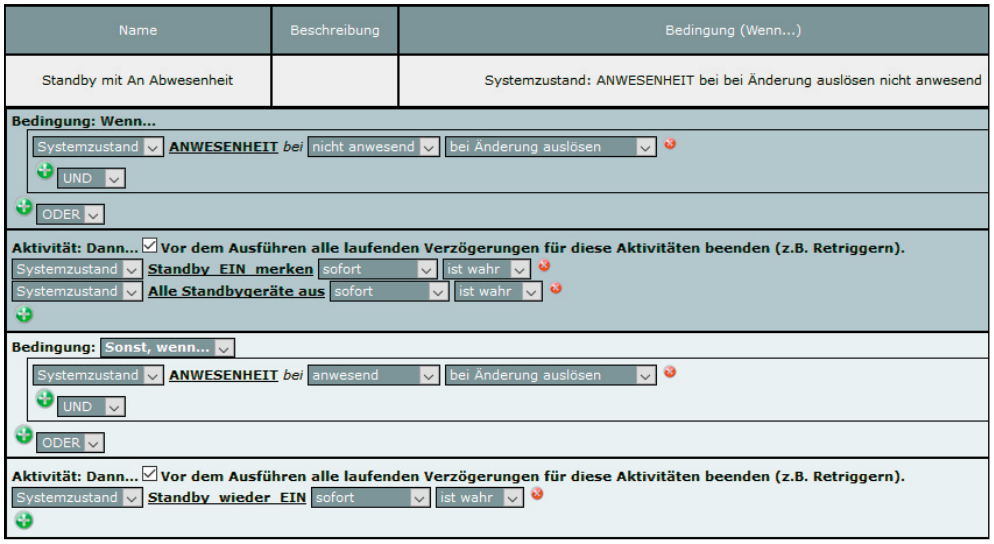
Auch dieses Script wird wiederum über eine Zentralenvariable gestartet:



Um lediglich diejenigen Geräte wieder einzuschalten, die auch tatsächlich abgeschaltet worden sind, könnte man folgendes Script verwenden:

```
! Geräte aus Zentralenvariable "War_eingeschaltet" einschalten
!-------------------------------------------------------------
string part;
foreach(part,
dom.GetObject("War_eingeschaltet").State().Split(","))
{
   State(dom.GetObject("BidCos-RF." + part +
":1.STATE").State(TRUE));
}
```

Damit kann die automatische Ab- und Wiedereinschaltung so aussehen:



Sobald die Zentralenvariable *ANWESENHEIT* in den Zustand *nicht anwesend* wechselt, wird zuerst die Zentralenvariable *Standby_EIN_merken* auf TRUE gesetzt, damit speichert die CCU wie oben beschrieben die Seriennummern der aktuell eingeschalteten Stand-by-Geräte. Danach werden alle Stand-by-Geräte ausgeschaltet.

Wechselt die Zentralenvariable *ANWESENHEIT* hingegen in den Zustand *anwesend*, dann werden alle Stand-by-Geräte, die beim Verlassen des Hauses eingeschaltet waren und automatisch ausgeschaltet wurden, wieder eingeschaltet. Die System-(Zentralen-)Variable *Standby_wieder_EIN* startet das Script zum Einschalten der in der Zentralenvariablen *War_eingeschaltet* gespeicherten Elemente.

Einbruchalarm-E-Mail verschicken

Im Alarmfall kann von der CCU eine Alarm-E-Mail mit dem Alarmauslöser verschickt werden. Der Alarmauslöser wird durch das oben beschriebene Script ermittelt und in der Systemvariablen (Zentralenvariablen) *Alarmausloeser* gespeichert.

Zum E-Mail-Versand muss auf der Zentrale das E-Mail-Add-on installiert und konfiguriert sein. Das Add-on kann kostenlos auf der Homematic Add-on-Seite heruntergeladen werden. Die Konfiguration ist in der dazugehörigen Dokumentation ausführlich beschrieben.

Das Script für den Versand sieht folgendermaßen aus:

Script-Alarm 1

```
01 !Alarmmail versenden
02 !-----------------------------------
03 string MailText =  system.Date("%d.%m.%Y") # " - " #
   system.Date("%H:%M:%S") # " - " # " Zugang ueber: " #
   dom.GetObject("Alarmausloeser").State();
04 !--------------
05 ! versenden
06 !--------------
07 string stdout;
08 string stderr;
09 system.Exec("/etc/config/addons/email/email 03
   '"+MailText+"'",&stdout,&stderr);
```

Da der (undokumentierte) *system.Exec*-Befehl nicht verwendet werden sollte, ist es sinnvoll, das CUxD-Add-on auf der CCU zu installieren und den Befehl durch den im CUxD enthaltenen *system.Exec*-Befehl zu ersetzen:

Script-Alarm 2

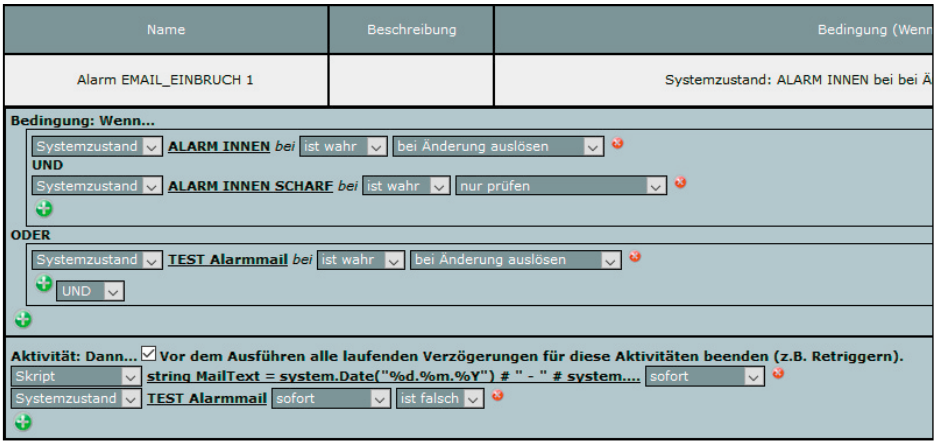
```
01 !Alarmmail versenden
02 !-----------------------------------
03 string MailText =  system.Date("%d.%m.%Y") # " - " #
   system.Date("%H:%M:%S") # " - " # " Zugang ueber: " #
   dom.GetObject("Alarmausloeser").State();
04 !--------------
05 ! versenden
06 !--------------
07 dom.GetObject("CUxD.CUX2800001:10.CMD_EXEC").State("/etc/config
   /addons/email/email 03 '"+MailText+"'");
```

In der Zeile 3 wird der E-Mail-Text gebildet, und zwar aus folgenden Daten:

```
system.Date("%d.%m.%Y")                 Tag, Monat und Jahr
system.Date("%H:%M:%S")                 Uhrzeit mit Stunden, Minuten und Sekunden
dom.GetObject("Alarmausloeser").State() dem gespeicherten Alarmauslöser
```

Die Zeile 9 im Script-Alarm 1 bzw. die Zeile 7 im Script-Alarm 2 versendet dann die E-Mail.

Das Script wird über ein Zentralenprogramm gestartet:



Die Systemvariable (Zentralenvariable) *Test_Alarmmail* wird nicht benötigt – sie dient lediglich dem Zweck, das Programm und das Script testen zu können.

Die E-Mail sieht dann beispielsweise folgendermaßen aus:

Die Möglichkeiten, die das Homematic System mit der CCU im Bereich Sicherheit bietet, sind vielfältig und können im Rahmen dieser Artikelserie nur in Teilen angesprochen werden. Die Beispiele in diesem Teil der Artikelserie dürften aber einige Anregungen zum Thema geben.

Im nächsten Teil der Artikelserie besprechen wir ein Projekt zum Einstellen, Abspeichern und Abrufen von Lichtszenen. ELV

Sehr geehrter Leser,
bei diesem Artikel zur Scriptprogrammierung handelt es sich um einen Fachbeitrag eines erfahrenen Homematic Users und Autors.
Die ELV/eQ-3 Unternehmensgruppe selbst nutzt die Möglichkeiten dieser Schnittstelle nicht, möchte aber den Anwendern der CCU2 den Zugang zu dieser Schnittstelle nicht verwehren.
Sollten Sie Schwierigkeiten bei der Verwendung dieser zusätzlichen Programmiermöglichkeit der CCU2 haben, so haben Sie bitte Verständnis dafür, dass wir Ihnen hierzu leider keinen Support geben können.
In den entsprechenden Foren und Internet-Plattformen rund um das Thema „Programmierung Homematic CCU" finden Sie jedoch sicherlich im Bedarfsfall die notwendigen Anregungen und Hilfestellungen für Ihr Projekt.

Mögliche Quellen im Internet:

https://www.homematic-inside.de/software/download/item/homematic-skript

https://homematic-forum.de/forum/viewtopic.php?f=19&t=18692