Programmiersprache CL-Software

Die Programmiersprache des Systems ermöglicht es jedem Anwender ohne spezielle Programmierkenntnisse komplexe Steuerungsfunktionen zu realisieren. Für jedes Objekt kann ein Makro erstellt werden, welches durch die Änderung von Zuständen bzw. Werten die mit den Objekten verbundenen Aktoren steuert. Zustände und Werte von Objekten und Variablen sowie Zeiten können mit vielfältigen Bedingungen abgefragt und durch Zuweisungen verändert werden.

Makros von Sensor-Objekten werden normalerweise immer ausgeführt, wenn eine Meldung von dem Sensor empfangen wird, beispielsweise also wenn eine Taste einer Fernbedienung gedrückt wird oder die aktuelle Temperatur eines Temperatursensors empfangen wird. Dazu wird beim Anlegen eines Sensor-Objekts standardmässig automatisch das Optionsfeld “Ausführen bei Empfang“ aktiviert.

Makros von Aktoren werden üblicherweise immer in einem bestimmten Zeitintervall aktiviert, beispielsweise um den Aktor nach einer bestimmten Zeit auszuschalten oder zu bestimmten Situationen und Abhängigkeiten von den Zuständen mehrerer anderer Objekten oder Zeiten zu steuern. Nach Möglichkeit sollte das gewählte Zeitintervall für die Makroausführung nicht kleiner als nötig gewählt werden. Üblicherweise reicht eine Minute, wenn jedoch Uhrzeitabfragen im Makro vorkommen, bei denen keine ganzen Minuten sondern Sekunden abgefragt werden muss das Zeitintervall kleiner, normalerweise auf 5 Sekunden eingestellt werden. Bitte beachten Sie dabei, dass die normale Uhrzeit im 5-Sekundentakt aktualisiert wird, die Uhrzeitangabe in den Bedingungen also ebenfalls eine Uhrzeit im 5-Sekundenintervall sein muss.

Die Makros können bei der Studio-Version wahlweise durch Menüs automatisch erstellt werden oder über die Tastatur eingegeben werden. Anstelle der in der Beschreibung der einzelnen Anweisungen verwendeten Syntax kann auch die bei der automatischen Erstellung von Makro-Anweisungen verwendete Syntax über die Tastatur eingegeben werden verwendet werden.
In jeder Zeile eines Programms darf nur eine Anweisung stehen, Kommentarzeilen müssen mit 2 Schrägstrichen (//) beginnen, diese werden dann bei der Generierung des Codes ignoriert.
In den Beispielen sind Programmanweisungen eingerückt, dies ist nicht erforderlich, aber aufgrund besserer Übersichtlichkeit empfehlenswert.
Wenn während der Code-Erstellung Fehler auftreten, wird der fehlerhafte Quelltext mit einer entsprechenden Fehlermeldung angezeigt. Die fehlerhafte Zeile wird soweit möglich markiert, hier kann es eventuell zu Verschiebungen kommen, da Fehler im Quell-Code nicht immer einer Zeile zugeordnet werden können. Wenn in der markierten Zeile kein Fehler erkennbar ist, müssen die vorangegangenen Zeilen geprüft werden.
Anweisungsteile, die bei der Erklärung der Syntax in **<** spitzen Klammern **>** stehen sind optional und können weggelassen werden.
Die Uhrzeit wird nicht jede Sekunde, sondern alle 5 Sekunden aktualisiert. Bei Zeitvergleichen ist also zu beachten, dass die Sekunden immer durch 5 teilbar sein müssen.
Wenn Schaltmodule eingeschaltet werden, kann die Einschaltdauer angegeben werden.

Die Elemente der Programmiersprache sind die

- **Objekte** mit ihren Zuständen, Werten und Variablen. Zustände und Werte von Objekten (und somit die zugehörige Hardware) können abgefragt und verändert werden.
- **Operatoren** für logische Verknüpfungen und Vergleiche
- **Schlüsselwörter**, die in der Programmiersprache definierte Bedeutungen haben
- **Anweisungen**, die in der Programmzeile beschreiben was getan werden soll
- **Funktionen**, die in der Programmzeile Werte liefern, z.B. für Vergleiche.

Objekte

Hardwareobjekte
sind Objekte, die bestimmter Hardware zugeordnet sind und zum Schalten, Messen, Regeln etc. benutzt werden. Man unterscheidet zwischen Sensoren und Aktoren.

Sensoren sind Eingabeelemente wie Taster von Fernbedienungen, Bewegungsmelder, Fensterkontakte, Temperatursensoren, Wassermelder, Rauchmelder usw. In den Makros der Sensoren stehen üblicherweise Anweisungen um Aktionen auszuführen wenn eine Meldung von der zugehörigen Hardware empfangen wird. Eine Makroanweisung zu einem Taster könnte also z.B. einen Aktor schalten wird, das sieht z.B. so aus:

```
Stehlampe umschalten
```

Aktoren sind Schaltmodule, an die Endgeräte wie Beleuchtung, Rollläden, Schalter, Markisen etc. angeschlossen werden. Wenn Makros von Aktoren benutzt werden, werden diese normalerweise in bestimmten Zeitintervallen ausgeführt. So kann z.B. ein Gerät immer nach einer bestimmten Zeitdauer ausgeschaltet werden wie in folgendem Beispiel-Makro:

```
wenn Badventilator eingeschaltet und
     Schaltdauer(Badventilator) groesser "00:10:00" dann
  Badventilator ausschalten
endewenn
```

Virtuelle Objekte
sind Objekte, die keiner Hardware zugeordnet und z.B. nur zur Bildschirm-Ein- oder Ausgabe benutzt werden.
Im Fenster Einstellungen können über den Button "Neues Objekt" virtuelle Objekte erstellt werden. Dabei wird der Name, die Bezeichnung und der Typ festgelegt: z.B. Name: Anzeige, Bezeichnung: Anzeige, Typ: Zeichen.
In einem Makro kann dem virtuelle Objekt Anzeige dann ein Wert zugeweisen werden, z.B:

```
Anzeige:="aktuelle Temperatur ist " + RaumreglerWohnzimmer.Temperatur+" Grad C"
```

Variablen

Variablen dienen der Zwischenspeicherung von Werten. Variablen werden wie die Objektprogramme auf der Seite Programmierung der Objektdefinition definiert. Einer Variablen kann im Objektprogramm ein Wert zugewiesen werden. Dieser Wert kann der Inhalt einer anderen Variablen, ein Objektzustand oder eine Konstante sein. Die in einem Objektprogramm definierten Variablen können von allen anderen Objektprogrammen benutzt werden, wenn der Name des Objekts, in dem die Variable definiert wurde, der Variablen getrennt durch einen Punkt vorangestellt wird.
Beispiel:

```
Anzeige:=RaumthermostatWohnen.Temperatur
```

Für jede Variable muss ein Typ festgelegt werden. Es können die gleichen Typen wie für Objekte verwendet werden.
Typen sind z.B.: Licht, Schalter, Zeichen, Zahl
Zusätzlich zu diesen Typen können für Variablen auch Zeittypen (Uhrzeit, Datum) benutzt werden.
Bei Zuweisungen zwischen nicht gleichen Typen erfolgt die Zuweisung entsprechend der Reihenfolge der für diesen Typ definierten Zustände, unabhängig von der Bezeichnung des Zustands. Intern werden Zustande als Zahl entsprechend der Reihenfolge in der Definition des Typs behandelt. Bei Zuweisungen wird diese Zahl zugewiesen.

Es gibt eine vordefinierte Variable des Typs Uhrzeit für jedes Objekt. Der Name der Variablen ist **CT** (für **C**hange**T**ime). In dieser Variablen wird die jeweils letzte Änderungszeit eines Objekts gespeichert. Mit der Funktion **`STOPPUHR(Objektname.CT)`** kann ermittelt werden wie lange ein Objekt sich in seinem aktuellen Zustand befindet.
Bitte beachten Sie:
Wenn Sie eine Variable vom Typ Zahl definieren, gibt es zwei unterschiedliche Typen von Zahlen:
Zahlen ohne Komma (**Ganzzahl**) und Zahlen mit Komma (**Gleitkommazahl**). Von welchem Typ die Zahl ist, wird durch den Startwert festgelegt. Wird als Startwert eine Zahl mit Komma angegeben (z.B. 1,0) so wird die Variable als Gleitkommazahl behandelt, andernfalls als Ganzzahl.
Es macht bei Rechenoperationen und Zuweisungen einen großen Unterschied wie eine Zahl definiert ist.
Wenn einer Ganzzahl eine Gleitkommazahl zugewiesen wird, werden die Kommastellen einfach abgeschnitten.
Wenn eine Rechenoperation nicht das gewünschte Ergebnis liefert, prüfen Sie wie die in der Anweisung verwendeten Zahlen definiert sind.

Beispiel:
Die Minimal und Maximal-Temperatur sollen täglich auf einer Anzeige ausgegeben werden.
Im Objektprogramm des Objekts TempAussen (das Thermometer im Garten) sind die Variablen MIN und MAX als Zahl definiert. Das Makro des Objekts TempAussen lautet:

```
WENN TempWG > MAX DANN
 MAX:=TempWG
ENDEWENN
WENN TempWG < MIN DANN
 MIN:=TempWG
ENDEWENN
```

Das Makro des Objekts Anzeige, das zu jedem Tageswechsel aktiviert wird:

```
AnzeigeMinMax:="Min: "+TempWG.MIN+" Max: "+TempWG.MAX
```

Zuweisungen

Zustände bzw. Werte von Objekten oder Variablen können durch Zuweisungen geändert werden.
Die grundsätzliche Form einer Zuweisung im Programm ist
Ziel := Quelle
wobei Ziel ein Objekt oder eine Variable sein kann. Quelle kann ein Objekt, eine Variable oder eine Konstante sein kann. Bei Zuweisungen von Werten unterschiedlichen Typs wird soweit möglich automatisch eine Konvertierung entsprechend des Typs des Ziels vorgenommen.
Bei Zuweisungen zwischen nicht gleichen selbst definierten Typen erfolgt die Zuweisung entsprechend der Reihenfolge der für diesen Typ definierten Zustände, unabhängig von der Bezeichnung des Zustands. Zum Beispiel muss bei der Definition neuer Typen mit den Zuständen AUS/AN darauf geachtet werden, dass der Zustand AUS immer zuerst definiert wird, damit bei Zuweisung an andere AUS/AN - Objekte der richtige Zustand übertragen wird.

Bei der Verwendung von Schlüsselwörtern wie **einschalten** bei Zuweisungen oder **ausgeschaltet** in Wenn-Bedingungen ist es erforderlich, dass AUS immer als erster Zustand definiert wird, da intern mit der numerischen Reihenfolge der Zustände gearbeitet wird. Durch diese Verfahrensweise ist es möglich, auch Zuweisungen zwischen Typen mit unterschiedlicher Bezeichnung für gleiche Zustände durchführen zu können. Bei der Definition der Zustände von Fenstern und Türen muss **offen** als erster Zustand und **zu** als zweiter Zustand definiert werden, damit die entsprechenden Schlüsselwörter bei Zuweisungen und
Bedingungen benutzt werden können.

Schlüsselwörter für Zuweisungen
Für Schalter: **`an, aus, einschalten, ausschalten`**
Bei Lichtern oder Geräten: **`einschalten für`**
Für Rollladen: **`rauffahren, runterfahren`**

Beispiele für Zuweisungen:
`LichtBad:=SchalterBad` hat die gleiche Wirkung wie
`LichtBad wie SchalterBad`
`Licht1Garten einschalten` hat die gleiche Wirkung wie
`Licht1Garten:=an` oder **`Licht1Garten:=1`**
Die besonderen Schlüsselwörter **AN** und **AUS** können bei einer Zuweisung mit dem Zuweisungsoperator **:=** ohne Hochkommas verwendet werden.
Wird einem Objekt ein anderer Zustand als Text zugewiesen, so muss dieser in Hochkommas gesetzt werden und genau der Schreibweise der Typdefinition entsprechen, auch Groß - / Kleinschreibung muss berücksichtigt werden.

Beispiel: Anstatt
`Rolllade runterfahren`
Könnte man auch die Anweisung
`RolladeWohnen:= "unten"`
verwenden, das Wort **unten** muss genau wie in der Typdefinition geschrieben sein und in Hochkommas gesetzt werden.
Eine besondere Anweisung zum Einschalten von Lichtern oder Geräten ist die Anweisung
Objekt **einschalten für** *Zeitdauer*
als Zeitdauer kann die Zeit in Hochkommas oder als Variable angegeben werden.

Beispiel:
`Stehlampe einschalten für "00:30:00"`
oder:
`Dauer1:="00:10:00"`
`Gartenlicht einschalten für Dauer1`
wobei Dauer1 eine Variable vom Typ *Uhr* oder *Zeichen* sein muss.

Rechenfunktion bei Zuweisungen numerischer Werte
Bei Zuweisungen an ein Objekt bzw. eine Variable des Typs Zahl können die vier Grundrechenarten benutzt werden. Damit ist es z.B. möglich Eingabewerte von Sensoren für die weitere Verarbeitung zu verändern.
Beispiel:
`NeuTemp:=AltTemp-5`

Rechenfunktion bei Zuweisungen von Uhrzeiten
Uhrzeiten können mit Hilfe der Operatoren **+** und **-** addiert bzw. subtrahiert werden.
Beispiel:
Sie wollen, dass ein Makro alle morgendlichen Aktivitäten um eine Viertelstunde verzögert. Dies könnte so aussehen:
```
RollRaufZeit:= RollRaufZeit + "00:15:00"
KaffeeEinZeit:= KaffeeEinZeit + "00:15:00"
```

Rechenfunktion bei Zuweisungen von Datumswerten
Es ist auch möglich ein Datum durch eine Rechenoperation zu ermitteln.
Das Datum 11 Tage nach dem aktuellen Tag kann man z.B. einfach ermitteln mit der Anweisung:
```
Zukunft:=Datum+11
```
wobei die Variable Zukunft natürlich vom Typ Datum sein muss.

Verknüpfungsfunktion bei Zuweisungen von Texten
Mit Hilfe von Zuweisungen können zwei oder mehr Zeichenketten verbunden werden. Wenn in der Anweisung Operanden eines anderen Typs als Zeichenkette benutzt wird, werden diese soweit möglich automatisch konvertiert.
Beispiel:
AnzeigeKueche:="Garage ist "+Garagentor
Wenn das Objekt Garagentor die Typen *auf* und *zu*hat, und das Garagentor auf ist erscheint in der Anzeige
Garage ist auf
```
AnzeigeBad:=Uhrzeit+" Uhr"
```
Der Text in der Anzeige sieht dann so aus: ***22:30:00 Uhr***

wenn - Bedingungen
In Wenn-Anweisungen wird aufgrund von Bedingungen entschieden, welche weiteren Anweisungen ausgeführt werden. Eine Bedingung ist entweder WAHR oder FALSCH.

Vergleichsoperatoren
= oder **gleich**
<> oder **ungleich**
< oder **kleiner**
> oder **groesser**
<= (kleiner oder gleich)
>= (größer oder gleich)

Logische Operatoren
UND (im Sinne von und zugleich), **ODER** (im Sinne von oder auch),
NICHT (zur Negation der nachfolgenden Bedingung)

Bedingungen sind folgendermaßen aufgebaut:
WENN <**NICHT**> *Bedingung* <**UND/ODER**> *Bedingung* **DANN**
Anweisungen
<**SONST**>
Anweisungen
ENDEWENN
Die Wenn-Anweisung ist eine Anweisung, die sich über mehrere Zeilen erstreckt.
Mit der WENN-Anweisung ist es möglich, den weiteren Programmablauf von einer oder mehreren Bedingungen abhängig zu machen. Wenn-Anweisungen können auch verschachtelt werden, d.h. zwischen dem **WENN** und dem **ENDEWENN** (bzw. **SONST**) können weitere Wenn-Anweisungen stehen. Jede Wenn-Anweisung muss mit einer endewenn-Anweisung beendet werden, ansonsten wird bei der Code-Generierung eine entsprechende Fehlermeldung ausgegeben.

Beispiel:
```
WENN LICHTBAD EINGESCHALTET UND
      SCHALTDAUER(LICHTBAD) GROESSER "00:00:30" DANN
 VENTILATOR EINSCHALTEN
ENDEWENN
```

Vor jeder Bedingung kann ein **NICHT** gesetzt werden, dann wird die Anweisung hinter dann ausgeführt wenn die Bedingung **nicht** zutrifft. **Bitte beachten Sie:**
Das Wort **NICHT** muss vor der eigentlichen Bedingung stehen, es darf nicht in der Bedingung stehen.
Beispiel:
Falsch wäre die umgangsprachliche Formulierung:
`wenn LichtBad NICHT ausgeschaltet oder Tag NICHT="Montag" dann`
es würde ein Syntaxfehler angezeigt.

Richtig ist:
`wenn NICHT LichtBad ausgeschaltet oder NICHT Tag="Montag" dann`

Bitte beachten Sie:
Häufige Syntaxfehler in wenn-Anweisungen sind, dass das Wort **DANN** vergessen wird und dass die Anweisung nicht mit einem **ENDEWENN** abgeschlossen wird.
Da wenn-Anweisungen sich immer über mehrere Zeilen erstrecken, kann die Zeile für einen Syntaxfehler oft nicht bestimmt werden. Wenn in einem Makro mit einer wenn-Anweisung ein Syntaxfehler ohne Fehlerbeschreibung auftritt, prüfen Sie alle Elemente der wenn-Anweisung um den Fehler zu finden.

Beispiel:
WENN Temperatur < 21,5 DANN
Wenn die Operanden unterschiedlichen Typs sind, findet soweit möglich eine automatische Konvertierung statt.
Wenn einer der Operanden eine Konstante ist, muss diese als *Operand2* stehen, damit eine korrekte Konvertierung durchgeführt werden kann.
Beispiel:
Richtig: WENN Uhrzeit = "15:15:00" DANN
Falsch: WENN "15:15:00" = Uhrzeit DANN

Besondere Vergleichsbedingungen
`wenn Objekt` **`eingeschaltet`** hat die gleiche Wirkung wie
`wenn Objekt = 1` bzw. `wenn Objekt =` **`"an"`**
`wenn Objekt` **`ausgeschaltet`** hat die gleiche Wirkung wie
`wenn Objekt = 0` bzw. `wenn Objekt =` **`"aus"`**

Schlüsselwörter für Bedingungen
Für Schalter: **eingeschaltet, ausgeschaltet**
Für Türen und Fenster: **geoeffnet, geschlossen**
Beispiel:
`wenn Fenster` **`geoeffnet`** `dann`
…

Zeit-Vergleiche
Bei Zeitvergleichen mit Uhrzeiten ist zu beachten, dass die Uhrzeit im 5-Sekunden-Takt aktualisiert wird, die Sekunden der Uhrzeit bei der Prüfung auf Zeitgleichheit also immer durch 5 teilbar sein müssen.
`wenn` **`Uhrzeit = "HH:MM:SS" .....`**
bei dieser Bedingung ist auch zu beachten, dass die Uhrzeit in Hochkommas gesetzt wird

`wenn` **`Datum = "TT.MM.JJ"`**
bei dieser Bedingung ist zu beachten, dass das Datum in Hochkommas gesetzt wird.

`wenn` **`Tag =`** `"Sonntag"`
Prüfung auf einen Wochentag. Der Wochentag wird in Hochkommas gesetzt und muss mit einem Großbuchstaben und nachfolgenden Kleinbuchstaben geschrieben werden.

`WENN Monat > 4 UND Monat < 10 DANN`
bewirkt, dass die folgenden Anweisungen nur zwischen Mai und September ausgeführt werden.

Vergleichsoperator =* für Vergleiche mit Jokerzeichen
Beim Vergleich von Uhrzeit und Datum mit einer Konstanten können auch Jokerzeichen verwendet werden. Als Vergleichsoperator muss **=*** verwendet werden. Operand2 muss eine Konstante in Hochkommas sein.
Beispiele:
Um Anweisungen jede volle und halbe Stunde auszuführen:

```
wenn Uhrzeit =* "**:00:00" oder Uhrzeit =* "**:30:00" dann
  ......
```

Um Anweisungen immer am Ersten eines Monats auszuführen:

```
wenn Datum =* "01.**.**" dann
  ......
```

Bedingung für Zeiträume
Manchmal ist es erforderlich Aktionen nur innerhalb bestimmter Zeiträume auszuführen.
Dies ist möglich mit der Bedingungsanweisung :
wenn Uhrzeit/Datum zwischen "*Zeitkonstante*" **und** "*Zeitkonstante*" **dann**
  ......

Beispiel:

```
WENN Uhrzeit ZWISCHEN "23:00:00" UND "05:00:00" DANN
  ......
```

Vergleichsoperator =+ für Vergleiche mit Wochenmaske
Mit diesem Vergleichsoperator ist es möglich, Anweisungen nur an bestimmten Wochentagen ausführen zu lassen. Als Operand2 muss eine 7-stellige Konstante bestehend aus Nullen und Einsen in Hochkommas verwendet werden. Jede Stelle der Konstanten steht für einen Wochentag, beginnend mit Sonntag. Die Bedingung ist wahr wenn an der Stelle des aktuellen Wochentags eine Eins steht.
Beispiel: Es soll geprüft werden, ob der aktuelle Tag ein Freitag, Samstag oder Sonntag ist.

```
WENN Wochentag =+ "1000011" DANN
```