



# HomeMatic-Script Documentation

## Part 3: Examples

**FOR FANS. BY FANS.**

**This version is not official and was created  
*with Google Translate***

V 1.1

English Version 0.2

## Table of Contents

1	Introduction.....	3
2	Example 1: Show maximum temperature.....	4
2.1	Overview.....	4
2.2	The system variable "maximum temperature".....	4
2.3	The program "Maximum temperature".....	5
2.4	Operation and Status View.....	6
2.5	Discussion.....	7
3	Example 2: Send data to 19-button remote control.....	8
3.1	Programming.....	8
3.2	Symbols.....	9
3.3	Text.....	9
3.4	Show Unit.....	10
3.5	Output sound.....	10
3.6	Backlight.....	10
3.7	Data transfer.....	11
3.8	Clear the display.....	11
3.9	Example.....	11
4	Example 3: Turn off all actuators and dimmers.....	12
4.1	Overview.....	12
4.2	The program "Light off".....	12
4.3	The favorite "Myfavorite".....	13
4.4	Operation.....	14
4.5	Discussion.....	15

# 1 Introduction

With the HomeMatic central it is possible to execute a script in response to an event. It uses its own scripting language, hereinafter referred to as the HomeMatic script.

HomeMatic script is used in the HomeMatic CCU in the WebUI within programs. This scripting language can be used to access the HomeMatic central logic layer. This allows, for example, taught-in HomeMatic components to be operated.

The aim of the HomeMatic script documentation is to give the user an insight into the programming of scripts that can be used by programs from the HomeMatic CCU.

The HomeMatic script documentation consists of the following 4 documents:

∇ Part 1: Language description

Describes the HomeMatic script language.

∇ Part 2: Object model

Describes in part the object model, which is based on the HomeMatic Central.

∇ Part 3: Examples

A collection of examples illustrating the handling of HomeMatic-script in a practical way.

∇ Part 4: Data points

Provides an overview of the available data points of the HomeMatic devices.

This document is part 3, examples. This part provides a collection of sample scripts. This is to illustrate the way the scripting language is working. In addition, some interesting applications are presented.

## 2 Example 1: Show maximum temperature

The **wireless** Combi sensor "HM-WDS100-C6-o" provides a multitude of information, such as the current temperature, relative humidity, rainfall, wind speed and much more. However, the sensor does not provide a maximum temperature display.

This example introduces a HomeMatic script that calculates the maximum temperature for incoming weather data.

### 2.1 Overview

In the following, the state variable "maximum temperature" is first created and linked to the wireless combi sensor. This means that the state variable is displayed in the status display of the wireless combi sensor.

The maximum temperature program is then created. It responds to incoming temperature data and updates the maximum temperature. If the program is executed manually, it should initialize the maximum temperature with the current temperature value.

Finally, the program operation and its pros and cons are discussed briefly.

### 2.2 The maximum temperature system variable

The maximum temperature system variable should contain the maximum temperature value. It is a numerical value whose range of values is between -50 and 100 °C. In order for the system variable to be used in the status display of the **wireless** combi sensor, it must be assigned to the corresponding channel.

The configuration page of the system variables can be reached in the HomeMatic WebUI via the menu item "System Configuration -> System variable". There you create the New system variable "maximum temperature".

Systemvariable bearbeiten					
Name	Beschreibung	Variablentyp	Werte	Maßeinheit	Kanalzuordnung
maximale Temperatur		Zahl	Wertebereich: Minimalwert = 0 Maximalwert = 100	°C	Funk-Kombisensor <input type="radio"/> ohne <input checked="" type="radio"/> mit kanalauswahl
<input type="button" value="Abbrechen"/> <input type="button" value="OK"/>					

The variable type is set to "number", the range of values should go from -50 to 100. The unit of measurement is "°c". Channel mapping is also important. Here the only channel of the **wireless** Combi sensor is used. In the example, it is named "Wireless Combi Sensor".

After the system variable has been created, it is displayed in the status display of the wireless combi sensor. However, their value is prefigured with "0.00".

### 2.3 The maximum temperature program

After the system variable "maximum temperature" has been created, the program "Maximum temperature" is to be created next. Note the different notation (small "m", Big "M"). Although the HomeMatic Control Panel allows the names of objects of different types to be identical, the function "dom.GetObject" however does only work correctly unless the identifiers are **unique**. Because the HomeMatic script is case-sensitive, the maximum temperature identifiers (the system variable) and the maximum temperature (the program) are different and "dom.GetObject" works as expected.

The maximum temperature program has two tasks:

1. Update of the maximum temperature for incoming weather data
2. Initialization of the maximum temperature during manual tripping

In the HomeMatic WebUI you reach via the menu item "Shortcuts and programs --> Program creation and Central linking" to the Program Overview. Here you can create a new program:

Name	Beschreibung	Bedingung (Wenn...)	Aktivität (Dann..., Sonst...)	Aktion
Maximale Temperatur		Kanalzustand: Funk-Kombisensor bei Lufttemperatur im Wertebereich größer oder gleich -40.00° C auslösen auf Aktualisierung	Skript: var source... sofort ausführen	<input type="checkbox"/> system intern
<b>Bedingung: Wenn...</b> <div> <div>Kanalzustand</div> <div>Funk-Kombisensor bei Lufttemperatur</div> <div>im Wertebereich größer oder gleich -40.00° C</div> <div>auslösen auf Aktualisierung</div> </div> <div> <div>+</div> <div>UND</div> </div> <div> <div>+</div> <div>ODER</div> </div>				
<b>Aktivität: Dann...</b> <input checked="" type="checkbox"/> Vor dem Ausführen alle laufenden Verzögerungen für diese Aktivitäten beenden (z.B. Retriggern). <div> <div>Skript</div> <div>var source = dom.GetObject("\$src\$"); var maxTemp = dom.GetObject("\$src\$").value</div> <div>sofort</div> </div>				
<b>Aktivität: Sonst...</b> <input checked="" type="checkbox"/> Vor dem Ausführen alle laufenden Verzögerungen für diese Aktivitäten beenden (z.B. Retriggern). <div> <div>+</div> </div>				

The air temperature update is used as a condition for the execution of the program. The program runs every update. The following script is entered as an activity:

# HomeMatic-Script Documentation

---

```
var source = dom.GetObject("$src$");
var maxTemp = dom.GetObject("maximum temperature");

if (source)
{
    if (source.Value() > maxTemp.Variable())
    {
        maxTemp.Variable(source.Value());
    }
}
else
{
    var wds = dom.GetObject ("Wireless combi sensor");
    var curTemp = wds.DPByHssDP ("Temperature");
    maxTemp.Variable(curTemp.Value ());
}
```

If the program is triggered by incoming weather data, its source (\$src\$) is defined. Because the trigger is the current air temperature, you can use "source.Value ()" is used to access the temperature value. The maximum temperature is set to the current temperature value if it is greater than the previous maximum temperature.

If there is no source for the program, the symbol \$src\$ is not replaced and the call to dom.GetObject "In the first row returns "zero". This is the case, for example, if the program is triggered manually. At this point, the maximum temperature value should be initialized with the current temperature.

The current temperature is hidden in the data point "temperature", which hangs on the only channel of the **wireless** Combi sensor. The script first gets this channel by "dom.GetObject ". The data point is then read using the method "DPByHssDP " and the system variable "maximum temperature" is initialized with its value.

After the maximum temperature program has been created, it must be set to active and can then be used.

## 2.4 Operation and Status View

To initialize the maximum temperature value, the maximum temperature program must be started manually once. This is best done via the WebUI and the user interface for programs.

Then you can display the **wireless** combination sensor in the status display. In addition to the usual weather data, the maximum temperature is now also displayed:

Verfügbare Kanäle im Gewerk			
Name	Raum	Letzte Aktualisierung	Status
Filter	Filter		
Funk-Kombisensor		13.11.2008 17:39:41	Lufttemperatur 22.40 °C
			Relative Luftfeuchte 54%
			aktuell kein Regen
			Regenmenge 1.18 mm
			Windgeschwindigkeit 0.00 km/h
			Windrichtung 295°
			Windrichtung Schwankungsbreite 0°
			Sonnenscheindauer 0
			Helligkeit 14
			Regen heute 1.18 mm
			Regen gestern 0.00 mm
			maximale Temperatur 22.50 °C

## 2.5 Discussion

The maximum temperature program updates the maximum temperature. The script, which forms the heart of this program, can easily be adapted for other purposes. The minimum temperature, maximum and minimum humidity or wind speed can also be displayed.

However, the script only works with maximum or minimum values. However, the maximum daily temperature is often of interest. For this purpose, the framework program could be extended by a time module, which resets the maximum daily temperature once a day at a specified hour. A clue as to how such a program can look, is provided by the system internal program "Rainy Day Quantity", which updates the values "Rain Today" and "Rain yesterday" for each **teached-in wireless** combination sensor.

### 3 Example 2: Send data to 19-button remote control

The HM-RC-19 19-button remote control has a display that can display various icons and a short text. Via HomeMatic-Script it is possible to integrate the display of such a remote control into a program.

#### 3.1 Programming

The 19-button remote display has a large number of data points. These represent the individual icons and functions of the display. Programming is always based on the same scheme:

1. Setting the data points
2. Transferring the configuration

First, all data points are set to match the desired representation. The entire configuration is then transferred to the remote control.

The individual data points can be controlled by `dom.GetObject`. Each data point has a *unique* name that is constructed according to the following schema:

```
BidCos-RF. < Serial number of the remote control >:18. < Name of the data point >
```

For example, if a 19-button remote control had the serial number "ACB1234567", then the full name of the data point "text" would look like this:

```
BidCos-RF.ACB1234567:18.Text
```

A script could then access this data point:

```
var dpText = dom.GetObject("BidCosRF.ACB1234567:18.Text");
```

The data points that are available and what they cause are described below.



## 3.2 Symbols

The 19-button remote control can display different icons. There is a separate data point for each of these symbols. The following table lists the available data points and the associated symbols:

Data point	Symbol
Bulb	
Switch	
Window	
Door	
Blind	
Scene	
Phone	
Clock	
Bell	
ARROW_UP	
ARROW_DOWN	

Each symbol can be activated individually. Symbols that have not been activated also do not appear on the remote control. Assuming that a remote control had the serial number "ABC1234567", the following script activates the clock symbol:

```
dom.GetObject("BidCos-RF.ABC1234567:18.CLOCK").State (1);
```

## 3.3 Text

```
Dom. GetObject ("BidCos-rf. ABC1234567:18. Text"). State ("Hello");
```

Via the data point "text" you can display a short message on the display of the 19-button remote control. Up to 5 characters can be displayed. However, special characters are not allowed and are replaced with spaces.

## 3.4 Show Unit

```
dom.GetObject("BidCos-RF.ABC1234567:18.Unit").State (3);
```

The icon for a unit can be displayed on the right side of the display. The following units are available for selection:

- ▽ 0: No unit
- ▽ 1: The percent sign (%)
- ▽ 2: Watt (W)
- ▽ 3: degrees Celsius (°C)
- ▽ 4: Degrees Fahrenheit (°F)

## 3.5 Output sound

```
fom.GetObject("BidCos-RF.ABC1234567:18.Beep").State (1);
```

In addition to the optical output, an audible alarm can also be activated. The corresponding data point is called "Beep" and supports the following values:

- ▽ 0: Mute
- ▽ 1: Sound 1
- ▽ 2: Sound 2
- ▽ 3: Sound 3

## 3.6 Backlight

```
dom.GetObject("BidCos-RF.ABC1234567:18.Backlight").State (2);
```

With the data point "backlight" you can control the behaviour of the backlight:

- ▽ 0: No backlight
- ▽ 1: Backlight on
- ▽ 2: Backlight flashes slowly
- ▽ 3: Backlight flashes quickly

### 3.7 Transferring data

```
dom.GetObject("BidCos-RF.ABC1234567:18.Submit").State(1);
```

So far, it has only been determined which symbols the display should display and how the remote control behaves. Only when the configuration is transferred to the display will the settings take effect. The data point "submit" exists for the transfer.

Symbols that have not been set before the transfer are not displayed.

### 3.8 Clear the Display

To completely erase the display of the remote control, the following script can be used:

```
dom.GetObject("BidCos-RF.ABC1234567:18.Text").State("");  
dom.GetObject("BidCos-RF.ABC1234567:18.Submit").State(1);
```

In the example, the remote control has the serial number "ABC1234567". This must be replaced in practical use by the serial number of the respective remote control.

### 3.9 Example

The following script reports an alarm to the 19-button remote control. The word "alarm" appears on the display and the bell symbol is displayed. In addition the backlight of the remote control flashes quickly. Finally, Tone 3 is played for the acoustic background.

In the example, the remote control has the serial number "ABC1234567". This must be replaced in practical use by the serial number of the respective remote control.

```
dom.GetObject("BidCos-RF.ABC1234567:18.TEXT").State("ALARM");  
dom.GetObject("BidCos-RF.ABC1234567:18.BELL").State(1);  
dom.GetObject("BidCos-RF.ABC1234567:18.BACKLIGHT").State(3);  
dom.GetObject("BidCos-RF.ABC1234567:18.BEEP").State(3);  
dom.GetObject("BidCos-RF.ABC1234567:18.SUBMIT").State(1);
```

## 4 Example 3: Turn off all **actuators** and dimmers

You can organize channels in rooms and **functions** via the HomeMatic CCU. In addition, you can create favorites to simplify the operation and monitoring of functional units.

Often you want to access all channels within a room, **function** or a favorite at the same time. This example demonstrates how to turn off all **actuators** and dimmers in a favorite by HomeMatic script. The script has the advantage over a conventional program that you can record or remove a device in the favorites without changing any script. Furthermore, the script can be easily transferred to rooms and **functions**.

### 4.1 Overview

The aim of this example is to create the new favorites "Myfavorite". This contains a lot of **wireless -actuators** of the type HM-LC-SW1-PL (luminaires), **wireless** -plug-dimming actuators of the type HM-LC-Dim1L-PL (dimmer) and the program "Light off". Optionally, additional items can be added to the favorites "Myfavorite".

The light off program will later turn off all lights and dimmers associated with the favorite. To do this, the program "Light Off" is created first. Then the favorite "Myfavorite" is created.

### 4.2 The Light Off program

The light off program has the task of turning off all luminaires and dimmers that are assigned to the favorite "Myfavorite".

To create the program, a new program is created in the HomeMatic WebUI under "program creation and central links". The program receives the name "light off" and has no conditions. The following script is created as a new activity:

```
var myFavorite = dom.GetObject("MyFavorite");
String itemId;
foreach(itemId, myFavorite.EnumUsedIDs())
{
    var item = dom.GetObject(itemId);
    If item.IsTypeOf(OT_CHANNEL)
    {
        var device = dom.GetObject(item.Device());
        if (device.HssType() == "HM-LC-SW1-PL") {item.State(0); }
        if (device.HssType() == "HM-LC-Dim1L-PL") {item.State(0.0); }
    }
}
```

The script first picks up the favorite "Myfavorite". Then iterates through all the elements of the favorite. Favorites can have different elements: channels, programs, system variables, and separators. However, there are only channels of interest here.

If the script finds a channel, it decides what should happen depending on the device type. If it is a light ("HM-LC-SW1-PL") or a dimmer ("HM-LC-Dim1L-PL"), the channels are switched off. To do this, the state of the respective channel is set to 0. It is important to note that the state of a dimmer is represented by a floating-point number. For this reason, the dimmer is set to 0.0. All other channels are not affected by the script.

The completed program is represented by the HomeMatic WebUI as follows:

Name	Beschreibung	Bedingung (Wenn...)	Aktivität (Dann..., Sonst...)	Aktion
Licht aus			Skript: var myFavo... sofort ausführen	<input type="checkbox"/> system intern
Bedingung: Wenn...				
<div>+</div> Aktivität: Dann... <input checked="" type="checkbox"/> Vor dem Ausführen alle laufenden Verzögerungen für diese Aktivitäten beenden (z.B. Retriggern).				
<div>+</div> Skript <input type="text" value="var myFavorite = dom.GetObject('MyFavorite'); string itemId;..."/> <input type="text" value="sofort"/> <input type="button" value="..."/>				
<div>+</div> Aktivität: Sonst... <input type="checkbox"/> Vor dem Ausführen alle laufenden Verzögerungen für diese Aktivitäten beenden (z.B. Retriggern).				

After the program has been completed with "OK", it still has to be set to active.

### 4.3 The favorite "Myfavorite"

After the light off program is created, a new favorite can be created. This is done in the WebUI in the menu item system configuration -> Favorites .

Various lights and dimmers are added to the favorites and the "Light off" program is added. The following illustration shows an example of "Myfavorite":

# HomeMatic-Script Documentation

Favoritenbeschreibung

Name	Beschreibung	Spaltenzahl	Spaltenausrichtung	Positionierungsvorschau	Namensfeld Position	Endgerät	Benutzer	Aktion
MyFavorite		2	zentriert	<div>1 2</div> <div>3 4</div> <div>5</div>	oben	PC	Admin	Löschen





Kanäle und Systemvariable in der Favoritenseite

kanal hinzufügen

Systemvariable hinzufügen

Programm hinzufügen

Separator hinzufügen

Position	Name	Typenbezeichnung	Bild	Bezeichnung	Seriennummer	Gewerk	Raum	Funktion	Aktion
	Filter				Filter	Filter	Filter		
1	Leuchte 1	HM-LC-Sw1-Pl Ch:1		Funk- Zwischenstecker- Schaltaktor 1fach	EEQ0002927:1	Licht	Büro	<input checked="" type="radio"/> Bedienung <input type="radio"/> Status	Löschen
2	Leuchte 2	HM-LC-Sw1-Pl Ch:1		Funk- Zwischenstecker- Schaltaktor 1fach	EEQ0003057:1	Licht	Büro	<input checked="" type="radio"/> Bedienung <input type="radio"/> Status	Löschen
3	Leuchte 3	HM-LC-Sw1-Pl Ch:1		Funk- Zwischenstecker- Schaltaktor 1fach	EEQ0002772:1	Licht	Büro	<input checked="" type="radio"/> Bedienung <input type="radio"/> Status	Löschen
4	Dimmer (Küche)	HM-LC-Dim1L-Pl Ch:1		Funk- Zwischenstecker- Dimmaktor 1fach Phasenanschnitt	EEQ0010661:1	Licht	Küche	<input checked="" type="radio"/> Bedienung <input type="radio"/> Status	Löschen
5	Licht aus							<input checked="" type="radio"/> Bedienung <input type="radio"/> Status	Löschen

Optionally, you can include other items in your favorites.

## 4.4 Operation

The best way to work with the light off program is to demonstrate it in Favorites view. Under the "Favorites" item, the new favorite "Myfavorite" appears in the HomeMatic WebUI. Its favorites view should resemble the following image:

Leuchte 1	<div>Aus</div>	Ein		Leuchte 2	<div>Aus</div>	Ein	
Leuchte 3	<div>Aus</div>	Ein		Dimmer (Küche)	<div> <div>0%</div> <div>100%</div> </div> <div> <div>0%</div> <div>100%</div> </div>	<div>Ein</div> <div>Aus</div>	
Licht aus	<div>Aktiv</div>	Start					

For testing, all lights and dimmers can be switched on first. However, as soon as the light off program is started, the luminaires and dimmers in question are switched off.

### 4.5 Discussion

The script, which forms the heart of the "Light Off" program, turns off all lights and dimmers in the favorite "Myfavorite". Instead of the name "Myfavorite", it can be any other favorite. Rooms and functions can also be used in this way. The only condition is that the name of the favorite, room, or craft is *unique* so that it is retrieved from "dom.GetObject" is found.

However, it is important to note that there are no exceptions to the script: you may want to turn off many devices in one room, but some should remain active. Two different solutions are available here:

1. You define your own favourite, which contains only the devices that are actually to be switched off.
2. The script is extended with explicit exception handling. Then, however, you need to customize the script as soon as a device is added or removed for which the exception applies.