



OrangeApps

# OrangeEdit 2.0 HowTo

User documentation

Date: 23<sup>rd</sup> of July, 2021, Version 0.3

© Copyright 2020

OrangeApps GmbH  
Arnikaweg 1  
87471 Durach  
Germany  
[www.orangeapps.de](http://www.orangeapps.de)

This documentation may – even partially – be copied and reposted. In the excerpt's reproduction a reference to the copyright owner and to this document must be noted.

The contents of this document have been tested with the described software. Since deviations cannot be excluded, no guarantee for full compliance can be taken.

**Documentation validity**

Version documentation	Software version	release	date
	Of	To	
<b>0.2</b>	2.0	Mayer	04'th November, 2018
<b>0.3</b>	2.0	Mayer	23'rd July, 2021

**History of the document versions**

version	date	author	Reason for change / comment
<b>0.1</b>	30'th January 2015	Daniel Schmidt	Initial creation
<b>0.2</b>	06'th May, 2020	Christian Mayer	general revision
<b>0.3</b>	23'rd July, 2021	Christian Mayer	Added chapter 8 and 9

**content**

<b>1</b>	<b>introduction .....</b>	<b>5</b>
	1.1 target group .....	5
	1.2 Presentation of information .....	5
	1.3 Terms used .....	5
<b>2</b>	<b>Generally.....</b>	<b>6</b>
	2.1 OrangeEdit HMI .....	6
<b>3</b>	<b>"Extras" functions.....</b>	<b>7</b>
	3.1 Rename positions .....	7
	3.2 Rename Pdat and Ldat .....	8
	3.3 Reverse order .....	9
	3.4 Edit block.....	10
	3.5 Clean data list.....	12
	3.6 Position transformation .....	13
	3.7 Mirror positions.....	14
	3.8 Adjust status / turn .....	16
	3.9 Sort data list .....	19
	3.10 Array wizard .....	20
	3.11 myHMI Designer.....	21
	3.12 Calculator (geometric operations) .....	23
	3.13 Tool and basic definitions.....	26
	3.14 Long text editor.....	27
	3.15 Active environment.....	28

<b>4</b>	<b>Object browser .....</b>	<b>30</b>
<b>5</b>	<b>Handling archives .....</b>	<b>31</b>
<b>6</b>	<b>Dealing with WorkVisual projects .....</b>	<b>32</b>
<b>7</b>	<b>Active environment .....</b>	<b>33</b>
<b>8</b>	<b>Inlineforms and KRL – commands.....</b>	<b>35</b>
<b>8.1</b>	<b>Inline forms.....</b>	<b>35</b>
<b>8.1.1</b>	<b>Example: Inline form “PTP movement”.....</b>	<b>35</b>
<b>8.2</b>	<b>KUKA standard inline forms for KSS 8.5 or higher .....</b>	<b>37</b>
<b>8.3</b>	<b>KRL commands.....</b>	<b>37</b>
<b>9</b>	<b>Syntax check .....</b>	<b>39</b>

# 1 introduction

## 1.1 target group

This documentation is aimed at users with the following knowledge:

- User knowledge of KUKA robot programming
- User knowledge of OrangeEdit 2.0

## 1.2 Presentation of information



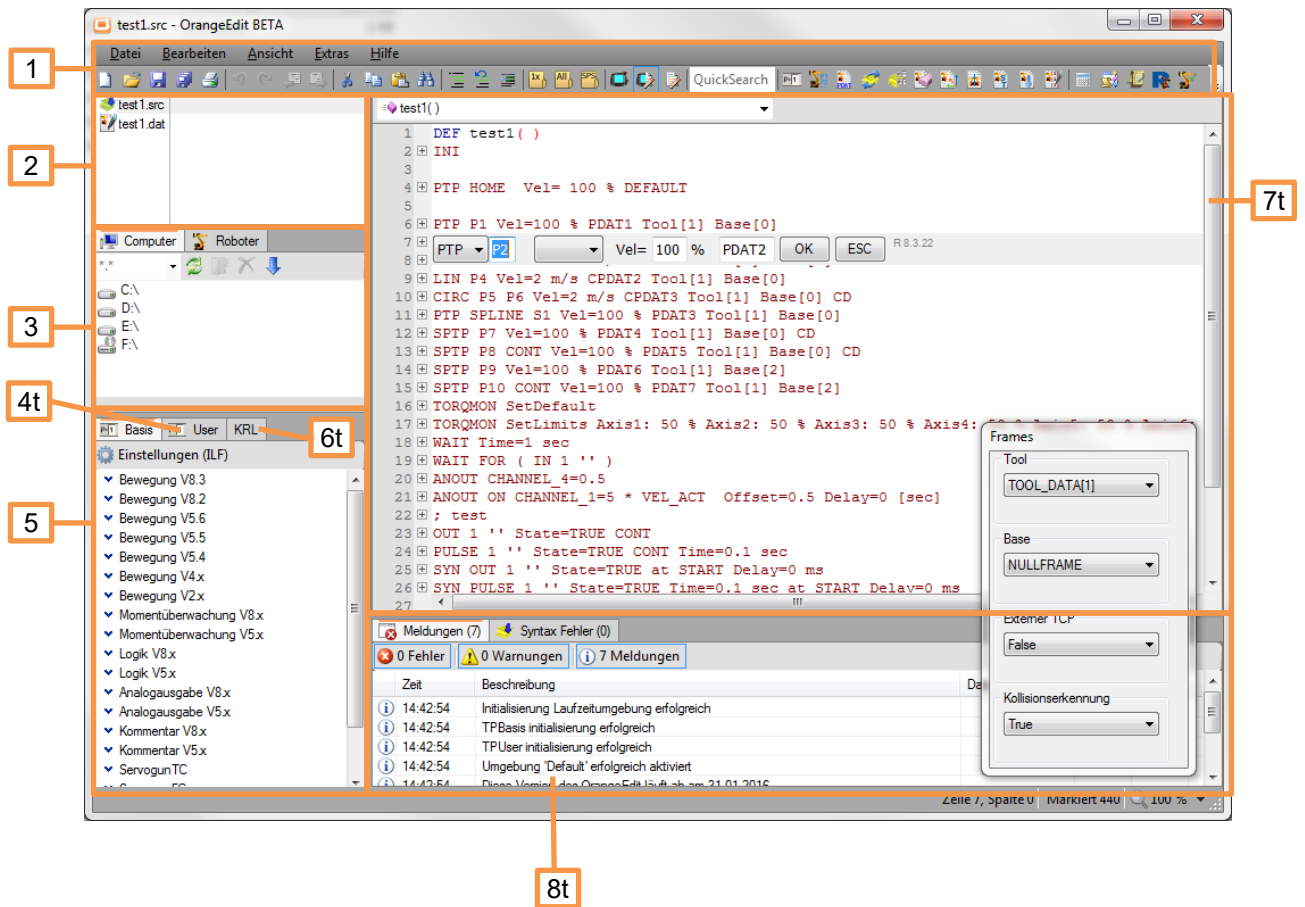
These notes contain useful tips or special information for the current topic.

## 1.3 Terms used

term	description
HMI	The Human-Machine Interface (HMI) is an interface through which a human communicates with a machine.
KSS	KUKA system software
smartPad	Robot control unit
KRL	KUKA Robot Language (KUKA robot programming language)
KFD	KUKA Form Description (inline form programming language)
TPBase	Standard inline form commands of the robot interface
TPUser	Inline forms that were developed on the basis of UserTech in KFD

## 2 Generally

### 2.1 OrangeEdit HMI



1. Menu and tool bar
2. List of open files
3. Explorer / drive window, the open archives are located under the "Robot" tab
4. User tab contains UserTech inline forms
5. Basic tab contains Basic (standard) inline forms
6. KRL tab contains KRL code templates for easy insertion
7. Program / editor window
8. Message window / syntax error display

## 3 "Extras" functions

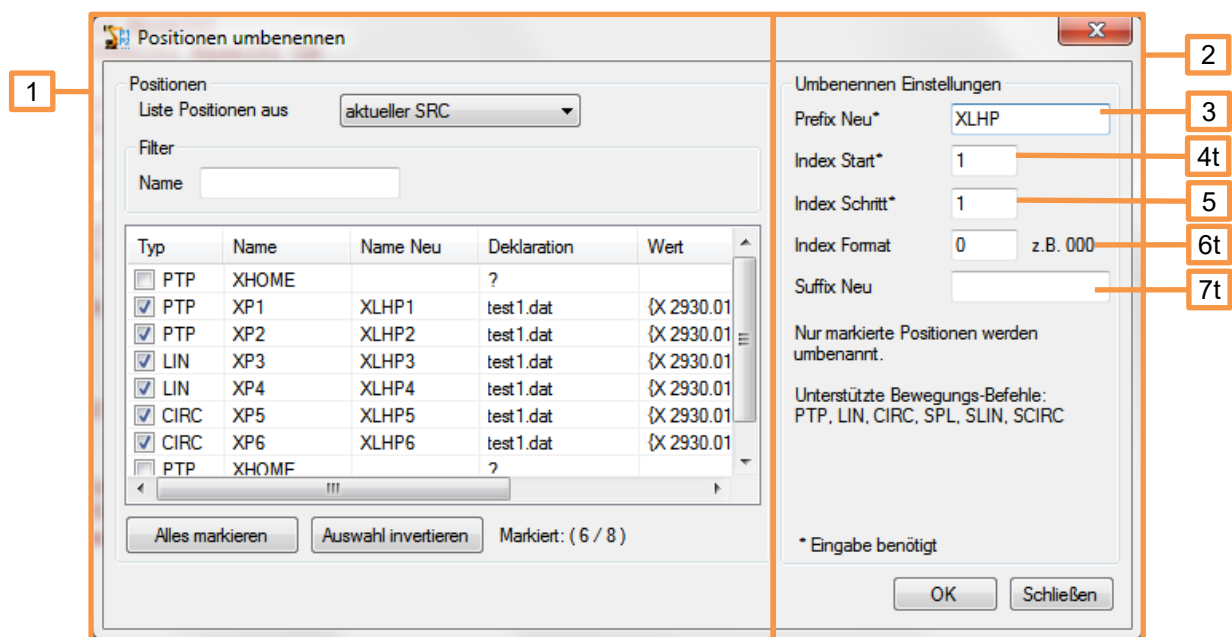
### 3.1 Rename positions

This function allows items created as inline forms to be renamed or renumbered. This may be necessary if items have been added at any point in an existing program or the naming does not correspond to the standard.

#### call

- Rename via menu Extras -> Positions
- Via the corresponding symbol in the toolbar

#### presentation



1. In this part of the window, the positions are selected which are to be renamed. Various filters help you select the points you want.
2. In this part of the window, all settings that affect the renaming are made.
3. Prefix New: The new point name begins with these characters.
4. Index start: For the consecutive numbering of the points, it is started from this entry.
5. Index step: The value of this information is incremented for each point.
6. Index format: If the numbering is to be preceded by one or more "0", the desired format can be specified via this input.
7. Suffix New: the new point name can optionally be supplemented with a fixed character string.

#### use

- Open the robot program in OrangeEdit
- Open the "Rename positions" function
- Select the desired positions to be renamed
- Make settings for renaming

- Start the priority with "Ok"



The renaming process is irreversible. A backup of the program should be created beforehand.

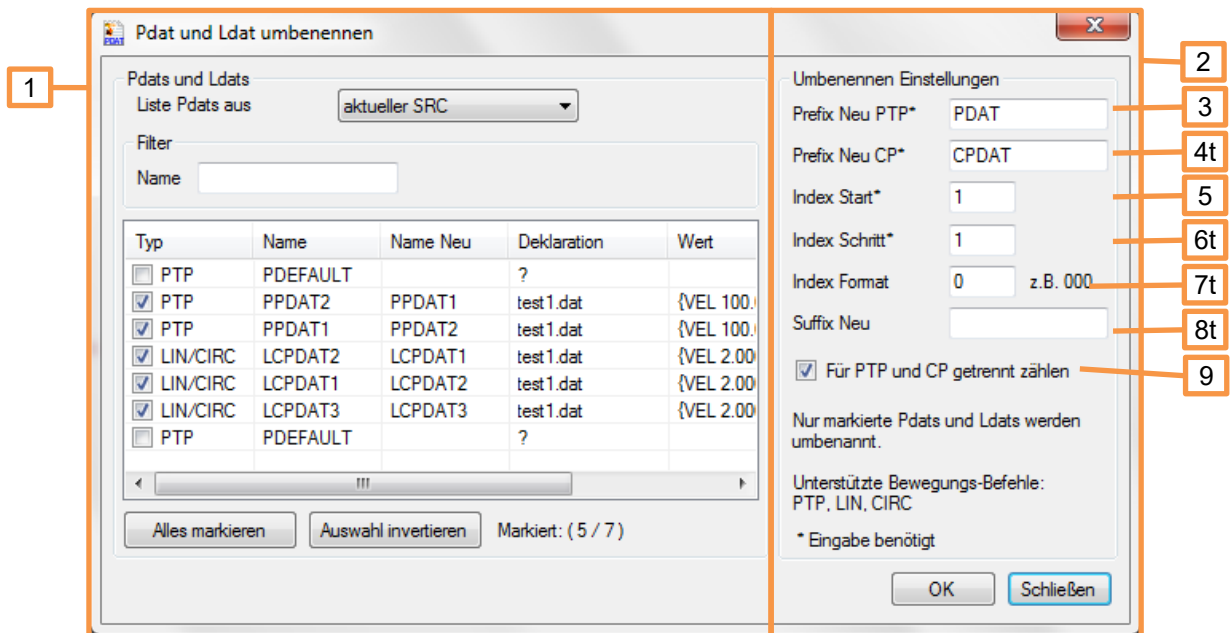
### 3.2 Rename Pdat and Ldat

The "Rename Pdat and Ldat" function enables the Pdat and Ldat to be renamed in the corresponding movement inline forms. This may be necessary if the naming does not correspond to the standard, or to give a program more "order".

#### call

- Rename via menu Extras -> Pdat and Ldat
- Via the corresponding symbol in the toolbar

#### presentation



- In this part of the window, the positions are selected which are to be renamed. Various filters help you to select the points you want.
- In this part of the window, all settings that affect the renaming are made.
- Prefix New PTP: The new name of the Pdat of a PTP movement begins with this character string.
- Prefix New CP: The new name of the Ldat of a LIN or CIRC movement begins with this character string.
- Index start: For the consecutive numbering of the Pdat and Ldat, the system starts with this entry.
- Index step: The value of this information is incremented for each Pdat and Ldat.
- Index format: If the numbering is to be preceded by one or more "0", the desired format can be specified via this input.



8. Suffix New: the new Pdat or Ldat name can optionally be supplemented with a fixed character string.
9. If this checkmark is set, Pdats and Ldats are counted separately.

#### use

- Open the robot program in OrangeEdit
- Open the "Rename Pdat and Ldat" function
- Select the desired positions / Pdat and Ldat that are to be renamed
- Make settings for renaming
- Start the priority with "Ok"

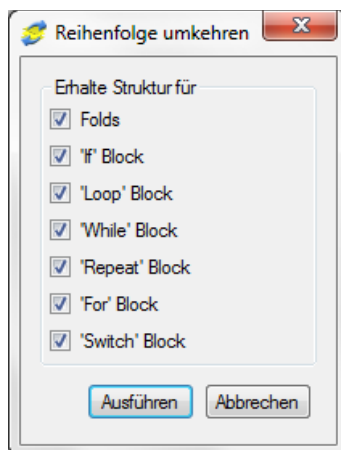
### 3.3 Reverse order

The "reverse order" function enables complete programs or sections of them to be reversed in the order in which they were processed. This may be necessary, for example, to follow a programmed path backwards (e.g. robot hemming, gluing).

#### call

- Via menu Extras -> Reverse order
- Via the corresponding symbol in the toolbar

#### presentation



- "Folds", fold structures and their contents are retained and are not reversed.
- "If" block, control structure for "IF, THEN, ELSE, ENDIF" is retained, contents are not reversed.
- "Loop" block, control structure for "Loop, Endloop" is retained, contents are not reversed.
- "While" block, control structure for "While, Endwhile" is retained, contents are not reversed.
- "Repeat" block, control structure for "Repeat, Until" is retained, contents are not reversed.
- "For" block, control structure for "For, Endfor" is retained, contents are not reversed.

- "Switch" block, control structure for "switch, end switch" is retained, contents are not reversed.
- "Execute", the process is started and transferred to the robot program.

#### use

- Open the robot program in OrangeEdit
- Mark the area in the robot program (e.g. several lines with movement points)
- Open the "Reverse order" function
- change settings
- Start the priority with "Execute"



It is advisable to leave all ticks set, otherwise the robot program will show errors when uploading to the robot controller.



If no area is marked in the robot program at the beginning, the function asks whether the whole program should be "reversed".



When changing from PTP to LIN, or with CIRC movements, the exact path is not followed backwards after reversing. The robot has to be reworked by hand here.

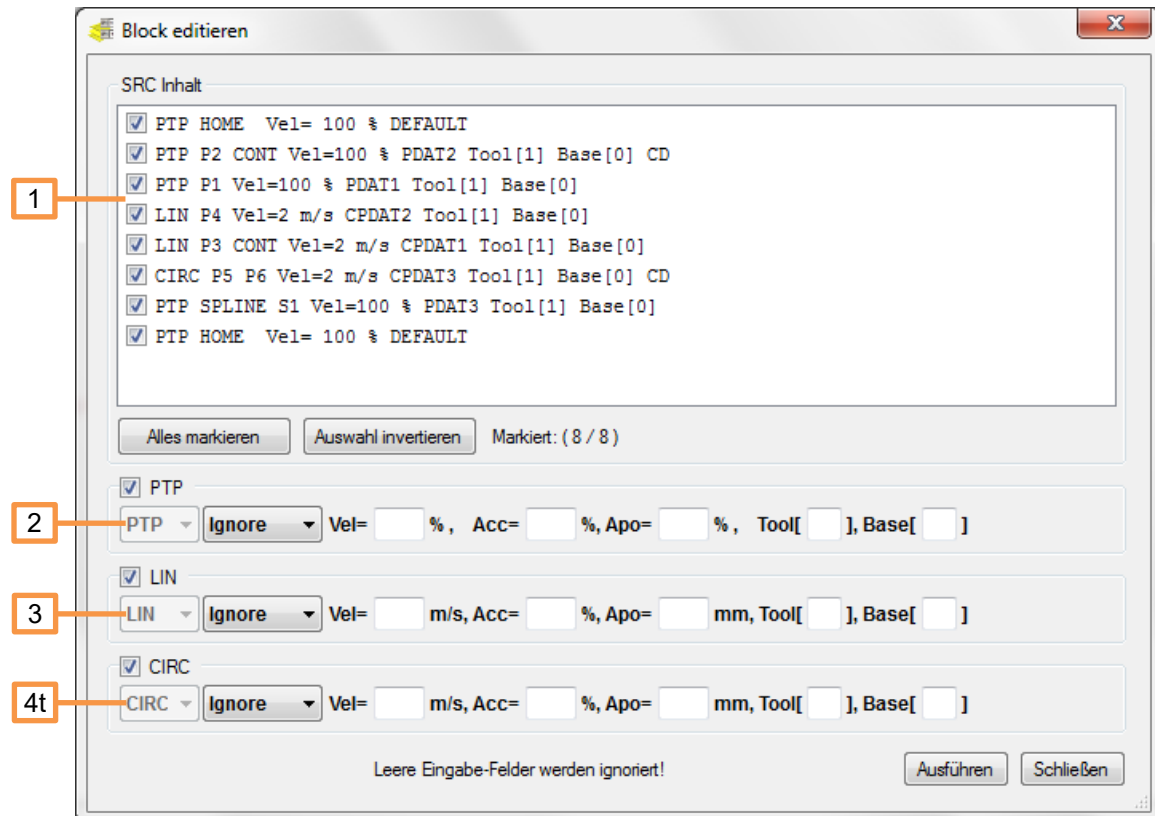
### 3.4 Edit block

The "Edit block" function enables various parameters of several movement inline forms to be changed at once. This can be useful if, for example, the speed needs to be adjusted for a number of points.

#### call

- Via menu Extras -> Edit block
- Via the corresponding symbol in the toolbar

## presentation



1. In this list, the positions are selected which are to be processed.
2. Parameters for PTP inline forms. Empty fields in this line are ignored, the relevant parameters in the inline forms are not changed.
3. Parameters for LIN inline forms. Empty fields in this line are ignored, the relevant parameters in the inline forms are not changed.
4. Parameters for CIRC inline forms. Empty fields in this line are ignored, the relevant parameters in the inline forms are not changed.

## use

- Open the robot program in OrangeEdit
- Mark the area in the robot program (e.g. several lines with movement points)
- Open the "Edit block" function
- change settings
- Start the priority with "Execute"



If you want to adjust all movements in a program, place the cursor anywhere without highlighting anything. The "Edit block" function now lists all movement points of the current program.



If the tool or base is changed, the point coordinates are currently not transformed. The taught position remains unchanged.

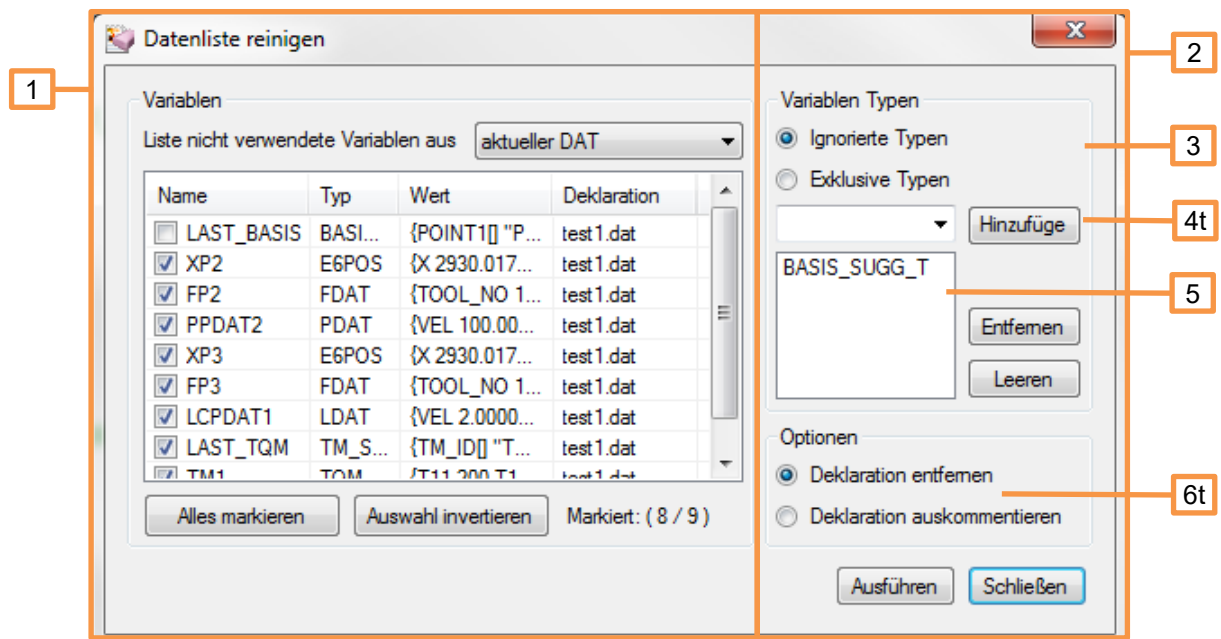
### 3.5 Clean data list

The "Clean data list" function enables unused variables to be automatically removed from the data list or to be commented on. By removing unused variables, less memory is required for the respective data list.

#### call

- Via menu Extras -> Clean data list
- Via the corresponding symbol in the toolbar

#### presentation



1. In this part of the window, the variables are selected which are to be removed. Only unused variables are displayed in the list.
2. In this part of the window, filters and settings for cleaning are made.
3. Here it is determined whether the variable types from the list below should be ignored or exclusively, only these variable types should be removed.
4. Here further variable types can be added to the filter list below.
5. List of variable types which, depending on the setting, should be ignored. A marked entry can be removed with the "Remove" button. The "Empty" button removes all entries from the list.
6. Here you can choose whether the selected variables are to be deleted or only commented out.

#### use

- Open the robot program in OrangeEdit
- Open the "Clean data list" function
- Variables that are not used are displayed in the list
- Make selection and settings
- Start the priority with "Execute"

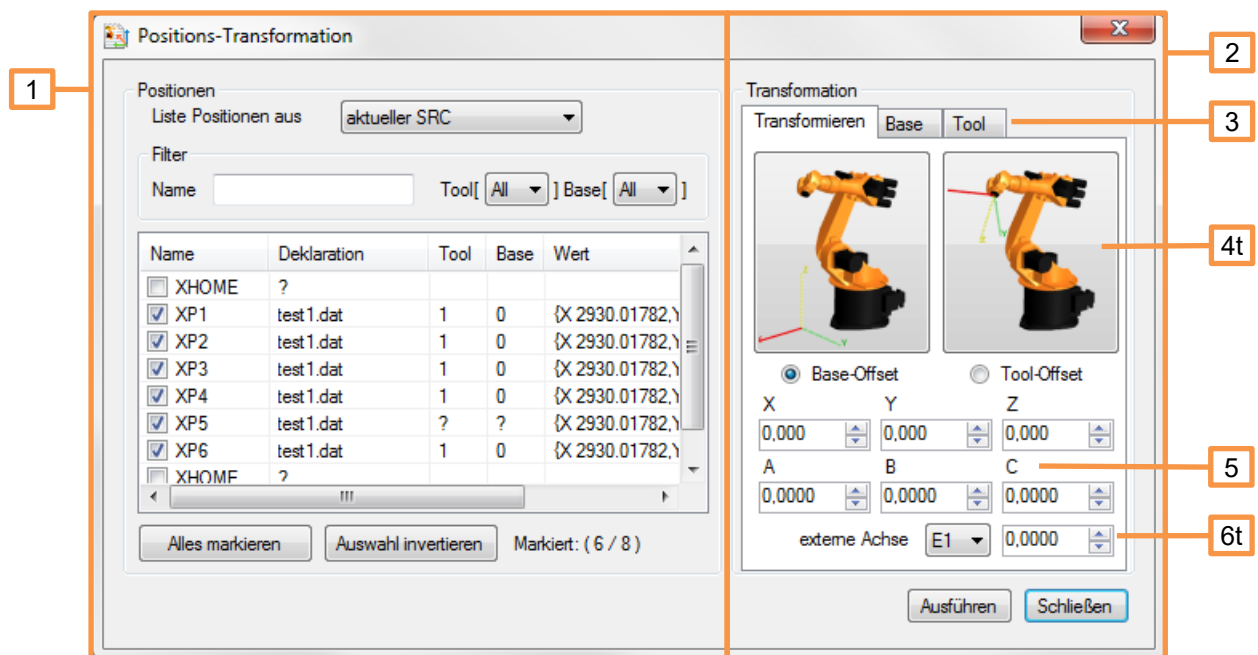
### 3.6 Position transformation

The "position transformation" enables points to be geometrically transformed (moved). It is possible to transform in the tool or base direction; there is also the option of performing a transformation between different coordinate systems by entering an old and a new base, or an old or new tool.

#### call

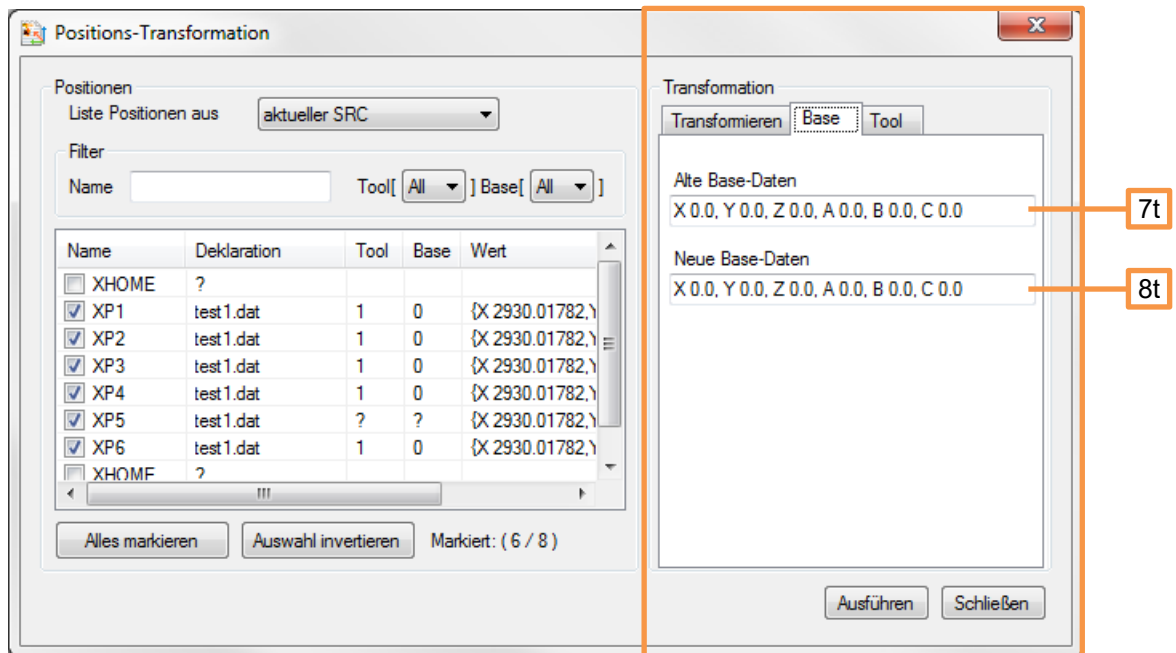
- Via menu Extras -> Position transformation
- Via the corresponding symbol in the toolbar

#### presentation



1. In this part of the window, the positions are selected which are to be transformed. Various filters help you select the points you want.
2. In this part of the window, all settings that affect the transformation are made.
3. Here you can choose between the following types of transformation:
  - Transform: Positions are transformed with the entered value.
  - Base: Here the positions are transformed in order to shift from the old to the new base.
  - Tool: Here the positions are transformed in order to move from the old to the new tool.
4. Direction in which the transformation affects.
5. Transformation values in X, Y, Z, A, B, C.

6. Optional shift value of E1 (external axis). This shift is "only" added to the E1 value of the respective position.



7. Input field for the old base data with which the positions were taught.
8. Input field for the new base data into which the positions are to be transformed.

#### use

- Open the robot program in OrangeEdit
- Open the "Position transformation" function
- Select the desired positions to be transformed
- Make settings for the transformation
- Start the priority with "Execute"



If the positions are taught with external TCP and you want to change the external TCP (eg adhesive nozzle), the transformation must be carried out as a tool offset or the transformation type "Tool" must be selected. If the gripper TCP is to be changed with external TCP, the transformation must be carried out as a base offset, or the transformation type "Base" must be selected.



You can find more information on the subject of "transformation" under "Calculator" (geometric operations)

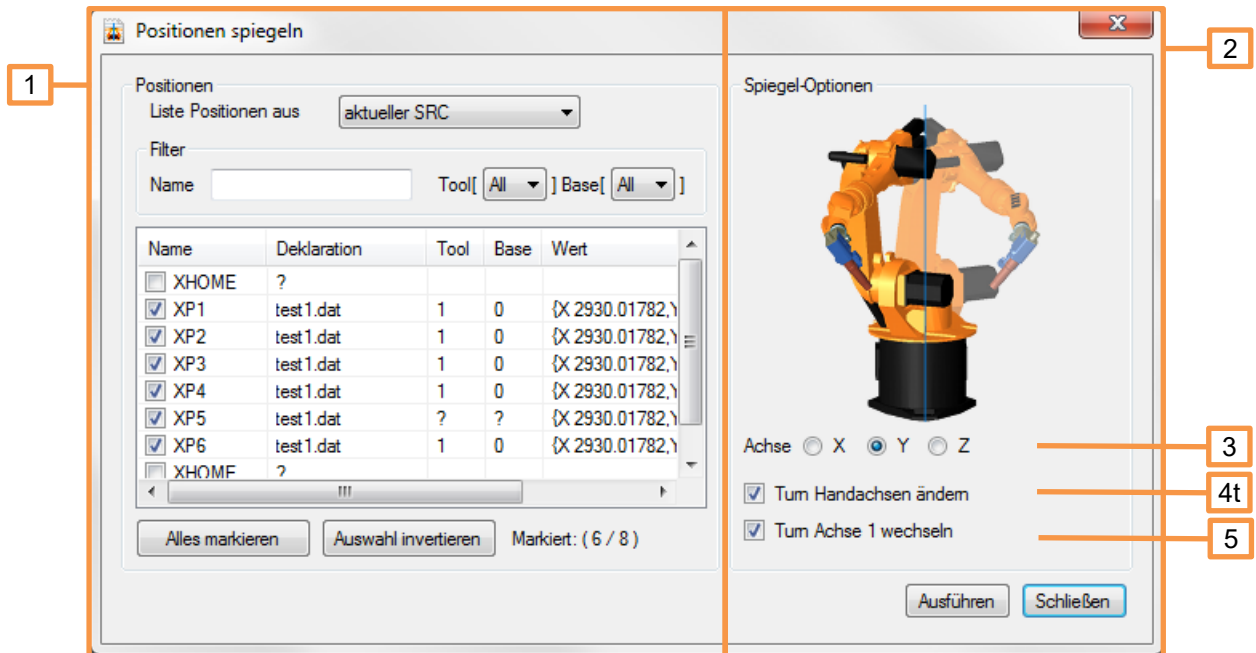
### 3.7 Mirror positions

The "Mirror positions" function enables selected positions to be mirrored in a selectable plane. This can be necessary, for example, to create a program for a mirrored component using the positions of the original component.


#### call

- Via menu Extras -> Mirror positions
- Via the corresponding symbol in the toolbar

**presentation**



1. In this part of the window, the positions are selected which are to be mirrored. Various filters help you select the points you want.
2. In this part of the window, all settings that affect the mirror transformation are made.
3. Mirror axis:
  - X →Positions are mirrored on the YZ plane. The sign of the X coordinate changes.
  - Y →Positions are mirrored on the XZ plane. The Y coordinate changes the sign. (Default)
  - Z →Positions are mirrored on the XY plane. The Z coordinate changes its sign.
4. With this option the turn bits of the wrist axes are inverted.
5. With this option the turn bit of axis 1 is inverted.

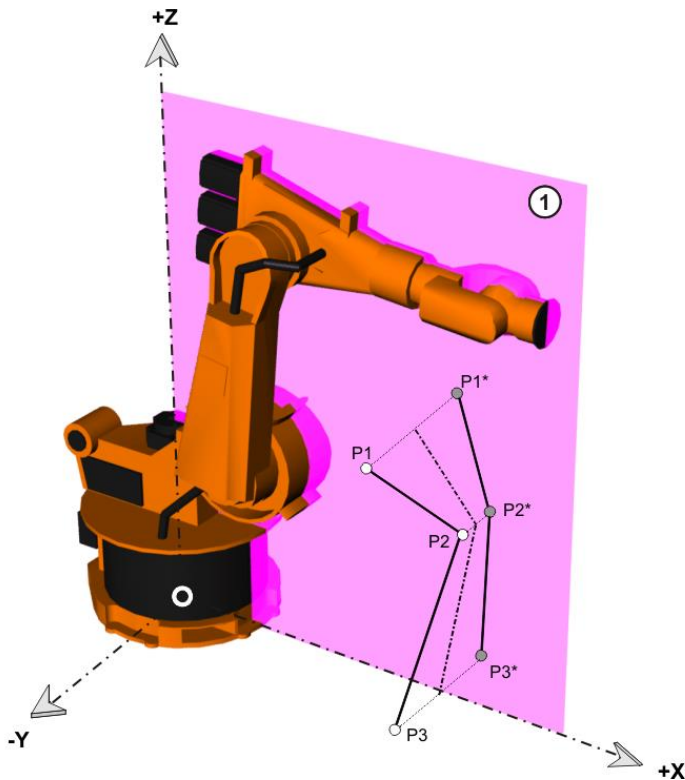
 You can find more information on the subject of "turn bit" under "Adjust status / turn".

**use**

- Open the robot program in OrangeEdit
- Open the "Mirror positions" function
- Select the desired positions to be mirrored
- Make settings for mirroring
- Start the priority with "Execute"

**example**

The points P1, P2, and P3 are mirrored in "Y" (XZ plane). The new positions of the points are P1\*, P2\* and P3\*.



After the axis mirroring, the tool used must also be mirrored in the XZ plane.



Since OrangeEdit cannot take the robot kinematics model into account, there is a probability that mirrored points cannot be reached with the respective axis configuration. It is recommended to approach such points with LIN, since the status / turn information is not taken into account.

**3.8 Adjust status / turn**

The "Adjust status / turn" function enables the status and turn information to be adjusted in one position (E6POS data type) in a simple white manner. This may be necessary after positions have been transformed. If, for example, a robot is rotated on the base, it is possible that positions cannot be reached even after a new base measurement. Often this is due to the change in sign of the axis 1 position.

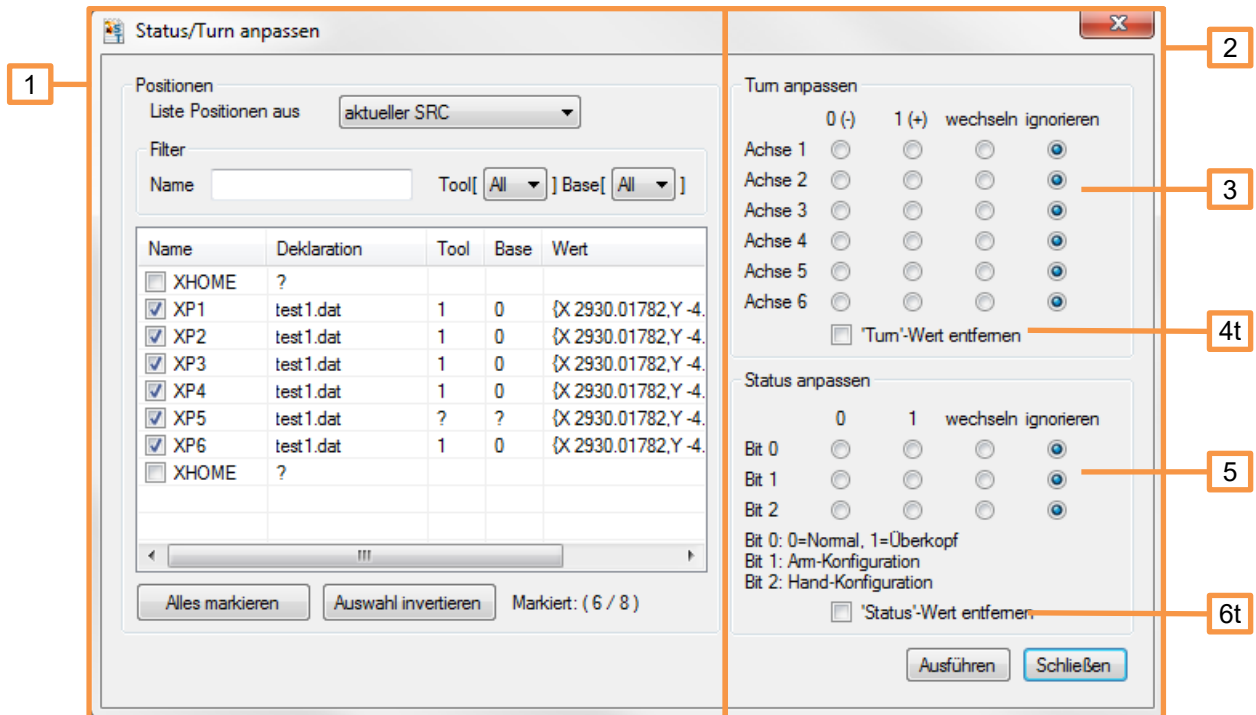
**call**

- Adjust via menu Extras -> Status / Turn



- Via the corresponding symbol in the toolbar

presentation



1. In this part of the window, the positions are selected whose status / turn information is to be adjusted. Various filters help you select the points you want.
2. In this part of the window, all settings relating to status and turn are made.
3. Here, a specification for the turn information can be made for each robot axis.
  - 0 (-), set the turn information of the axis to minus (axis value is <0).
  - 1 (+), set the turn information of the axis to plus (axis value is >= 0).
  - Change, the turn information is inverted. 1 becomes 0, 0 becomes 1.
  - Ignore, the turn information is not changed for this axis.
4. The turn information is completely removed from the E6POS declaration (all axes). When approaching, the robot now selects the shortest axis value (Attention! Check the movements of the robot).
5. The respective status bit can be adjusted here (see pictures below).
  - 0, the status information is set to the value 0.
  - 1, the status information is set to the value 1.
  - Change, the status information is inverted. 1 becomes 0, 0 becomes 1.
  - Ignore, the status information remains unchanged.
6. The status information is completely removed from the E6POS declaration. When approaching, the robot now selects the shortest axis value (Attention! Check the movements of the robot).

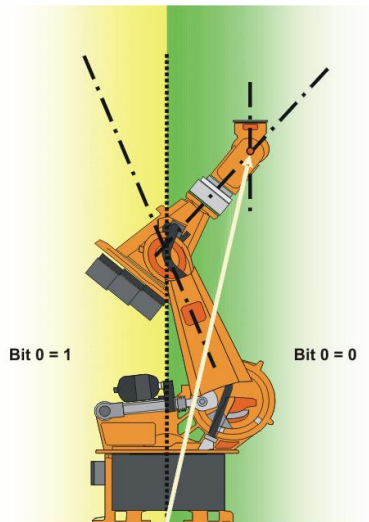
use

- Open the robot program in OrangeEdit
- Open the "Adjust status / turn" function
- Select the desired positions to be adjusted

- Make settings for the adjustment
- Start the priority with "Execute"

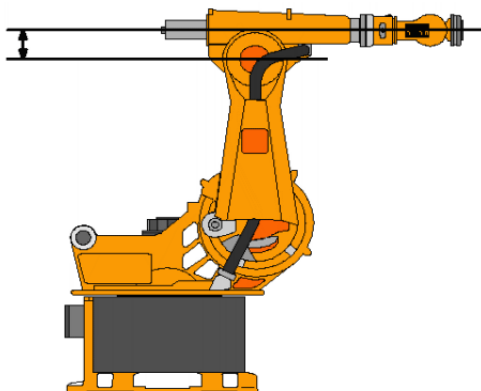
### Status bit 0

The status information prevents ambiguities in the axis position. Bit 0 indicates the position of the intersection of the wrist axes (A4, A5, A6).



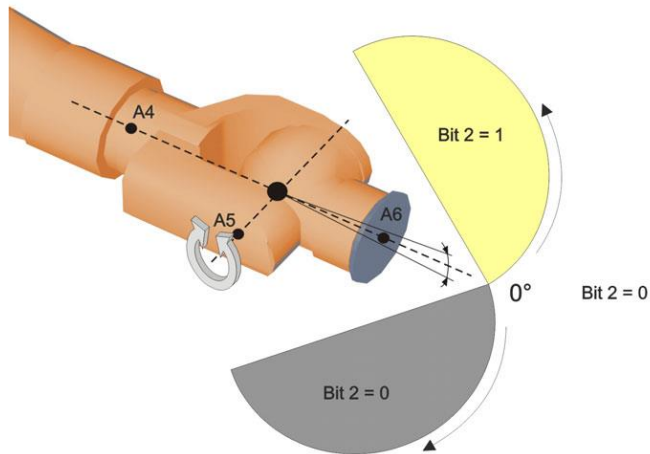
### Status bit 1

Bit 1 indicates the position of axis 3. The angle at which the value of bit 1 changes depends on the robot type.



### Status bit 2

Bit 2 indicates the position of axis 5. The sign of the angle of A5 is not decisive! The decisive factor is the direction in which the A5 is tilted, i.e. up or down. The direction indication in turn relates to the zero position of the A4.



You can find more information on the topic of "Status / Turn" in the KUKA documentation:

*KUKA System Software 8.3 Operating and programming instructions for system integrators >> Chapter 9.10*

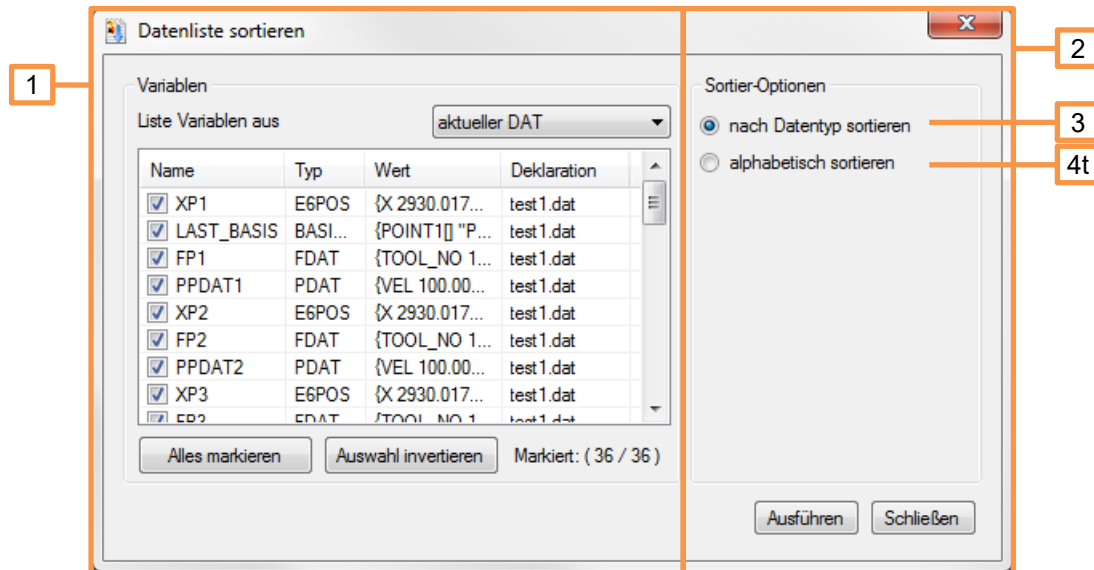
### 3.9 Sort data list

With the "Sort data list" function, it is possible to sort the declarations in a data list. This brings a little more order and readability to a data list. The function of the robot remains unaffected.

#### call

- Via menu Extras -> Sort data list
- Via the corresponding symbol in the toolbar

## presentation



1. In this part of the window, the variables are selected which are to be sorted.
2. In this part of the window, all settings that affect the sorting are made.
3. "Sort by data type", all selected variables are grouped and sorted by data type.
4. "Sort alphabetically", all selected variables are sorted and sorted alphabetically.

## use

- Open the robot program in OrangeEdit
- Open the "Sort data list" function
- Select the variables you want to be sorted
- Make settings for sorting
- Start the priority with "Execute"

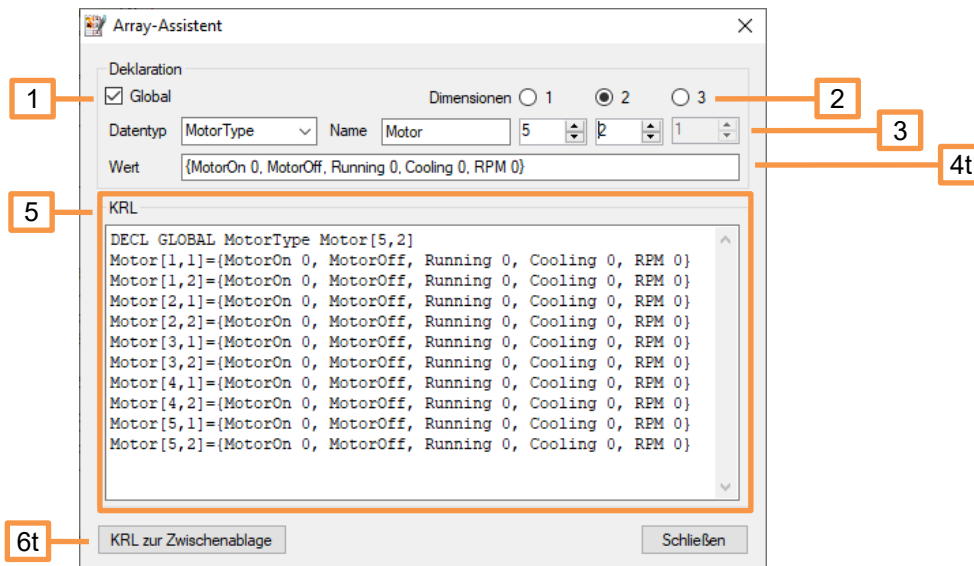
## 3.10 Array wizard

The array wizard creates the declaration of one to three-dimensional arrays and thus saves the tedious manual typing.

## call

- Via menu Extras -> Array Assistant
- Via the corresponding symbol in the toolbar

## presentation



1. Declaration as global or local variable.
2. Dimension of the array
3. Data type and name of the array variables
4. Values of the elements of the array
5. Representation of the array in KRL
- 6t. Copies the array to the clipboard

### 3.11 myHMI Designer

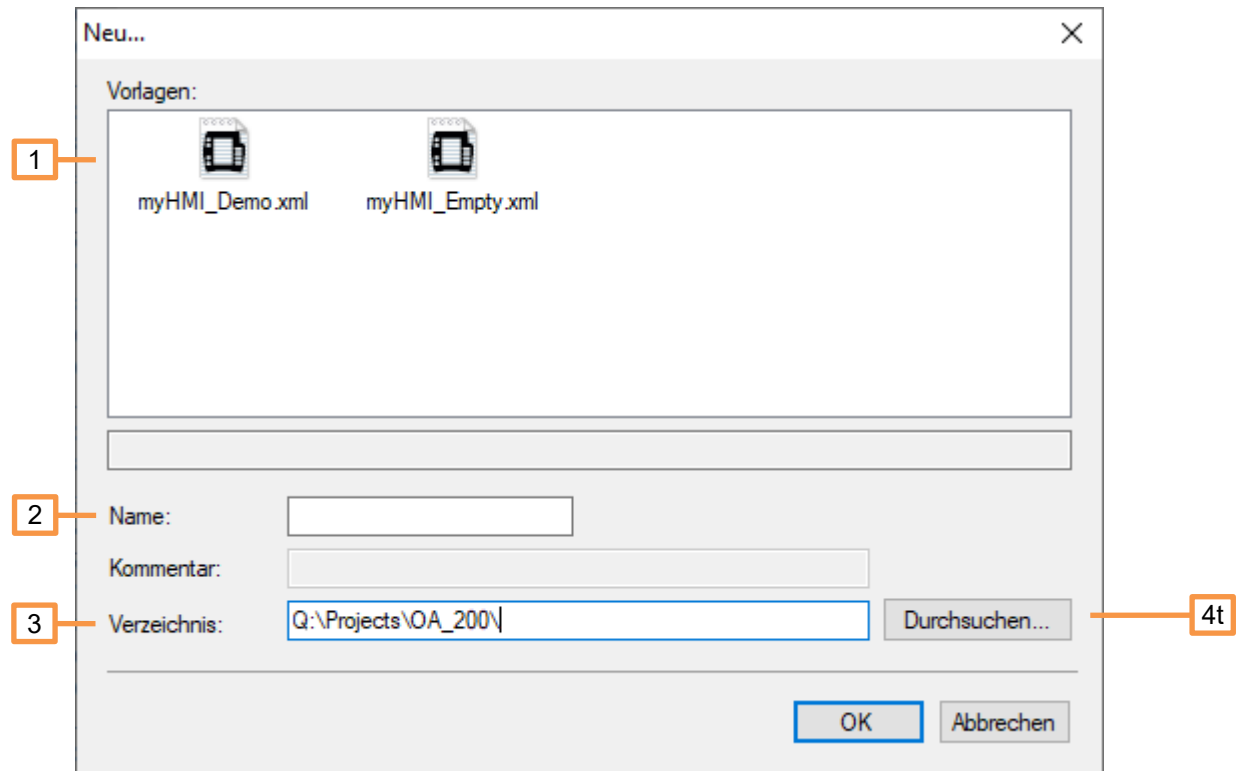
Designer for creating user interfaces for the "myHMI" software.

#### call

- Via the Extras -> myHMI Designer menu
- Via the corresponding symbol in the toolbar

After selecting this command, a file dialog opens for selecting an existing surface or one of the two implemented templates.

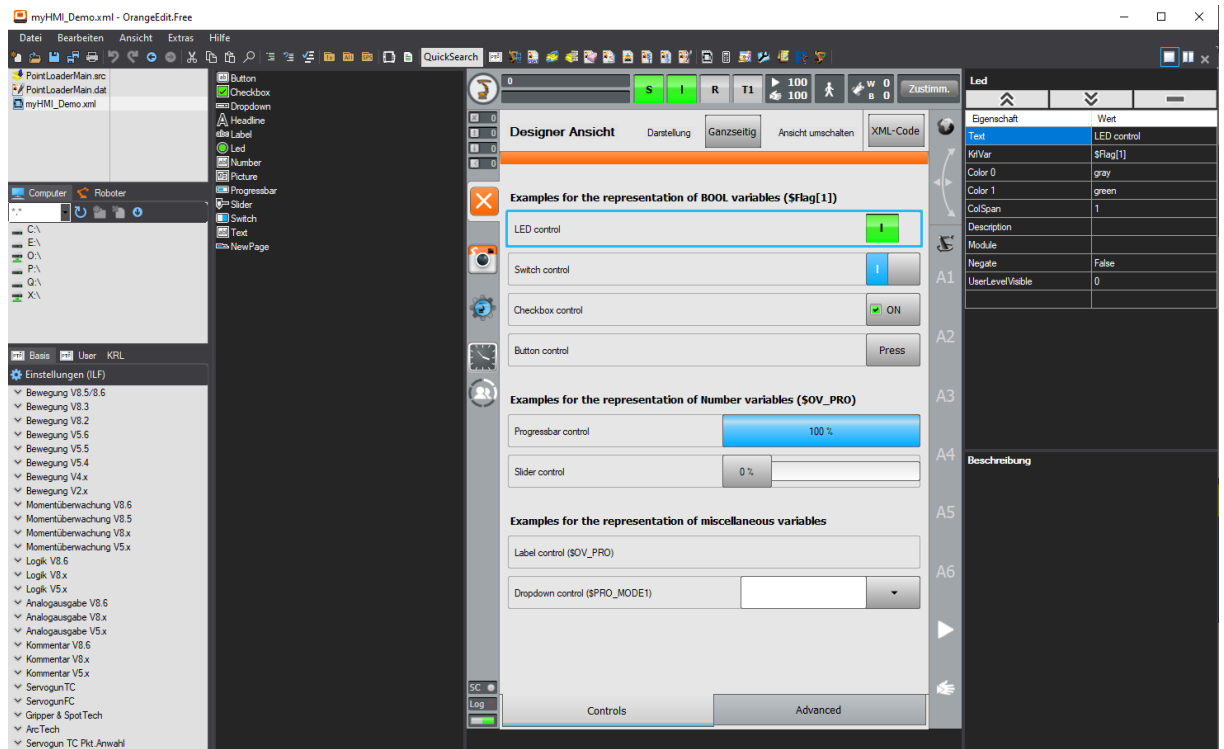
## Representation of the file selection dialog



1. Select template.
2. Enter the name of the HMI (no spaces allowed)
3. File path for the existing surface
4. Folder selection dialog for existing surface

After selecting a template or an existing surface, the designer window opens.

## presentation



A description of the operation can be found in the documentation for the "myHMI" software.

Link to the software and instructions: <https://orangeapps.de/?lng=de&page=apps%2Fmyhmi>

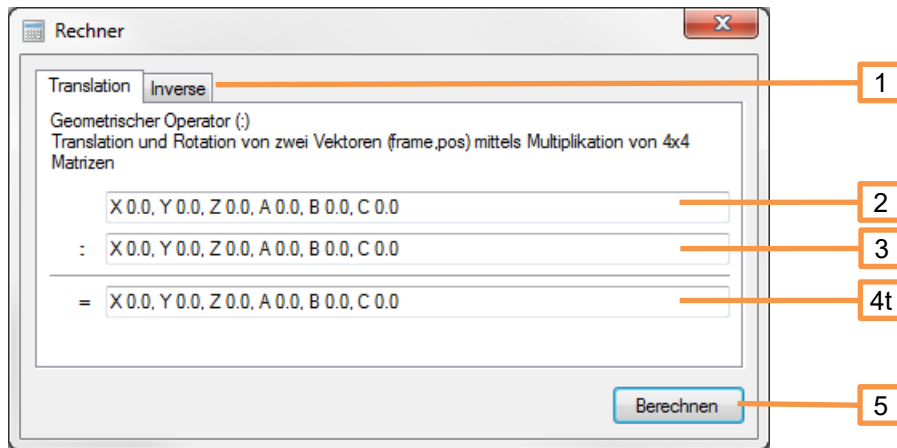
### 3.12 Calculator (geometric operations)

With the "calculator" in OrangeEdit, geometric operations can be carried out. For example, an offset can be calculated on a TCP, a base shift and much more.

#### call

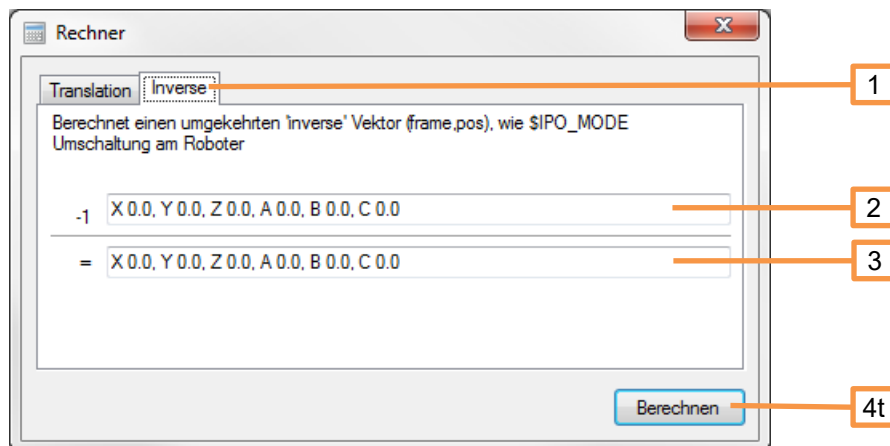
- Via menu Extras -> Calculator
- Via the corresponding symbol in the toolbar

### Representation (transformation)



1. Switching between transformation and inverse.
2. Input field for vector 1.
3. Input field for vector 2.
4. Result of the transformation.
5. "Calculate" button, triggers the calculation.

### Representation (inverse)



1. Switching between transformation and inverse.
2. Input field for the vector.
3. Result (inverse) of the entered vector.
4. "Calculate" button, triggers the calculation.

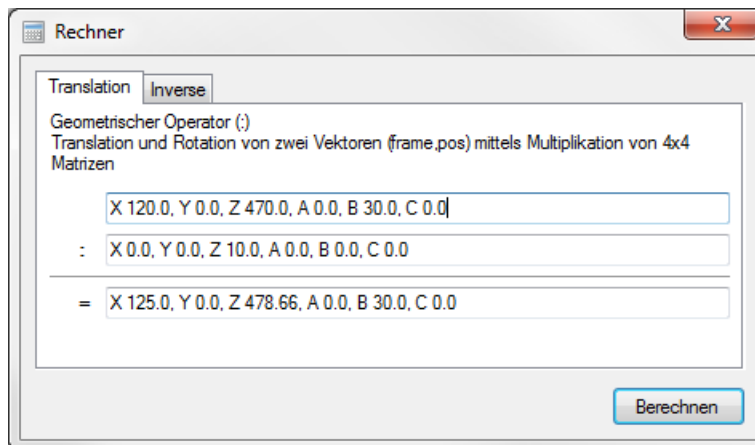
### Example 1 (add offset to TCP)

You want to add 10mm to a TCP of a tool in the Z direction of the tool. The tool is attached to the robot flange at a 30 ° angle.

Tool = X 120.0, Y 0.0, Z 470.0, A 0.0, B 30.0, C 0.0

Enter the data in the first field and the desired shift in the second.





As a result, you will receive new tool data for your tool, to which 10 mm have been added in the Z direction.

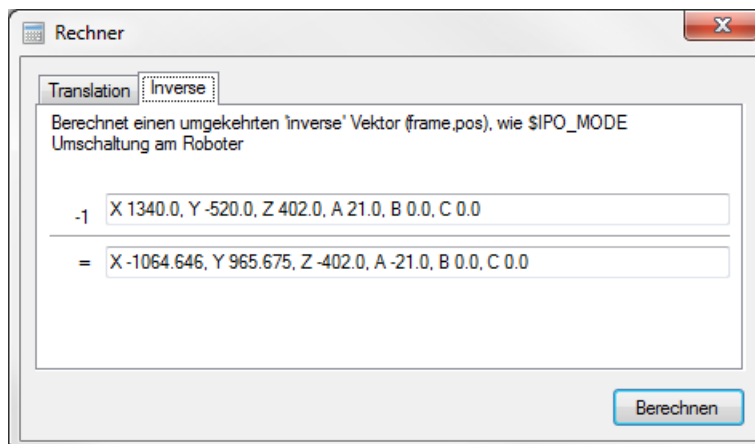
### Example 2 (determining the difference between two base data)

You have re-measured a base because a processing station has moved. Now you want to know how big the difference is in relation to the old base.

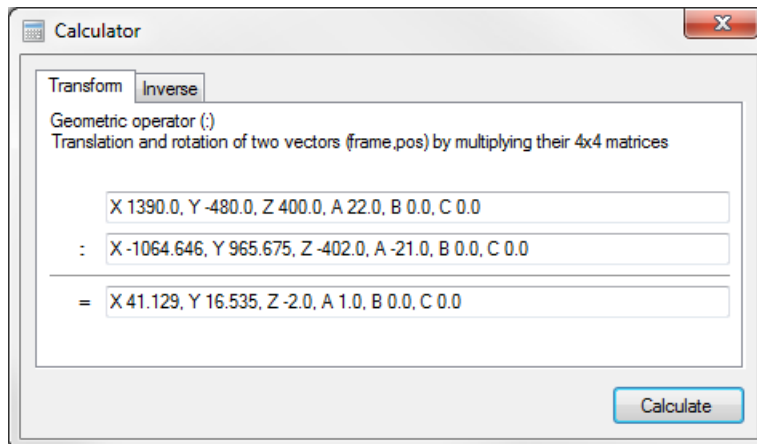
Old base: X 1390.0, Y -480.0, Z 400.0, A 22.0, B 0.0, C 0.0

New base: X 1340.0, Y -520.0, Z 402.0, Q 21.0, B 0.0, C 0.0

1. Calculate the inverse of the new base.



- Enter the old base in the first field under Translation, the result of the inverse in the second field and click "Calculate".



The result is the difference between the old base and the new base in the coordinate network of the old base.

This value can now be used as a transformation vector to move positions.

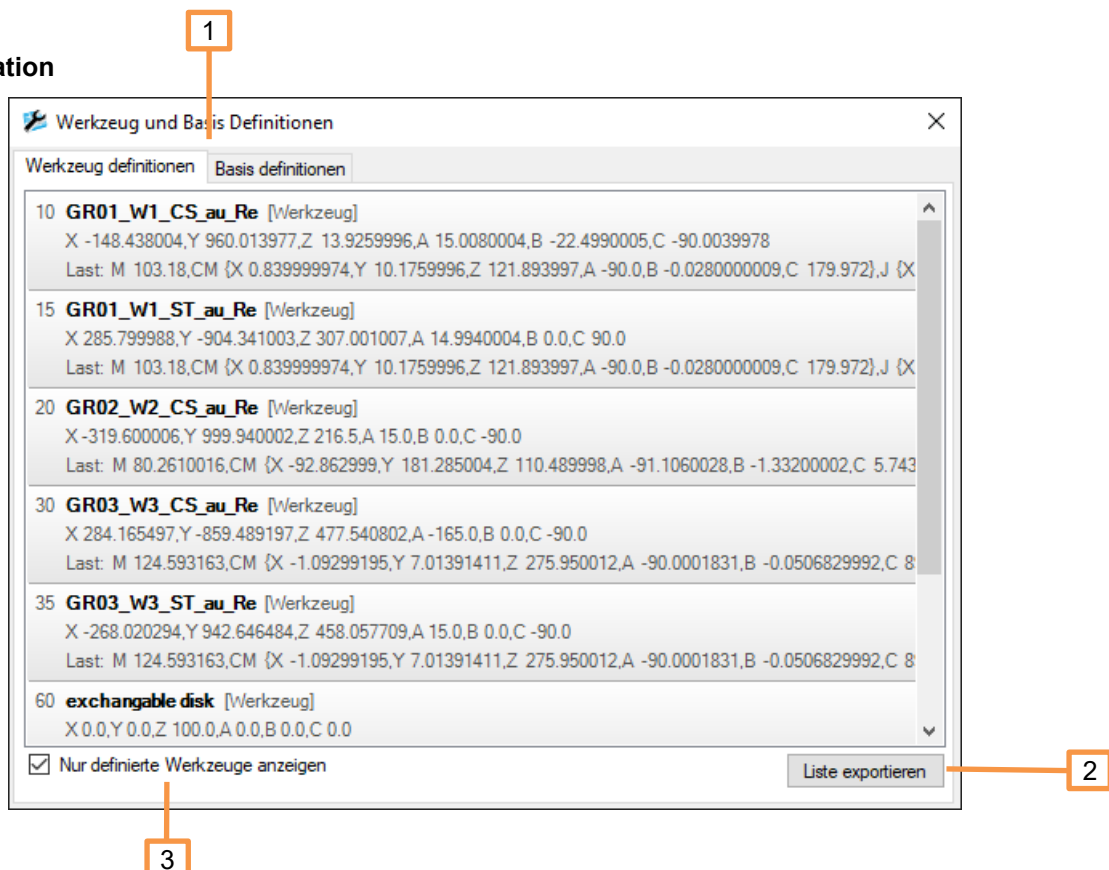
### 3.13 Tool and basic definitions

Displays the tool and basic data of the active environment

call

- Via menu Extras -> Calculator
- Via the corresponding symbol in the toolbar

presentation



1. Switching between tool and basic definitions
2. Exports the list to a CSV file
3. Only shows defined tools or basic systems

### 3.14 Long text editor

The long text editor enables the long texts for inputs, outputs, analog I / Os, timers, counters and flags to be edited. The long texts are stored KRC1 / 2 compatible in an Access database and can be exported KRC2 and KRC4 (WorkVisual) compatible.

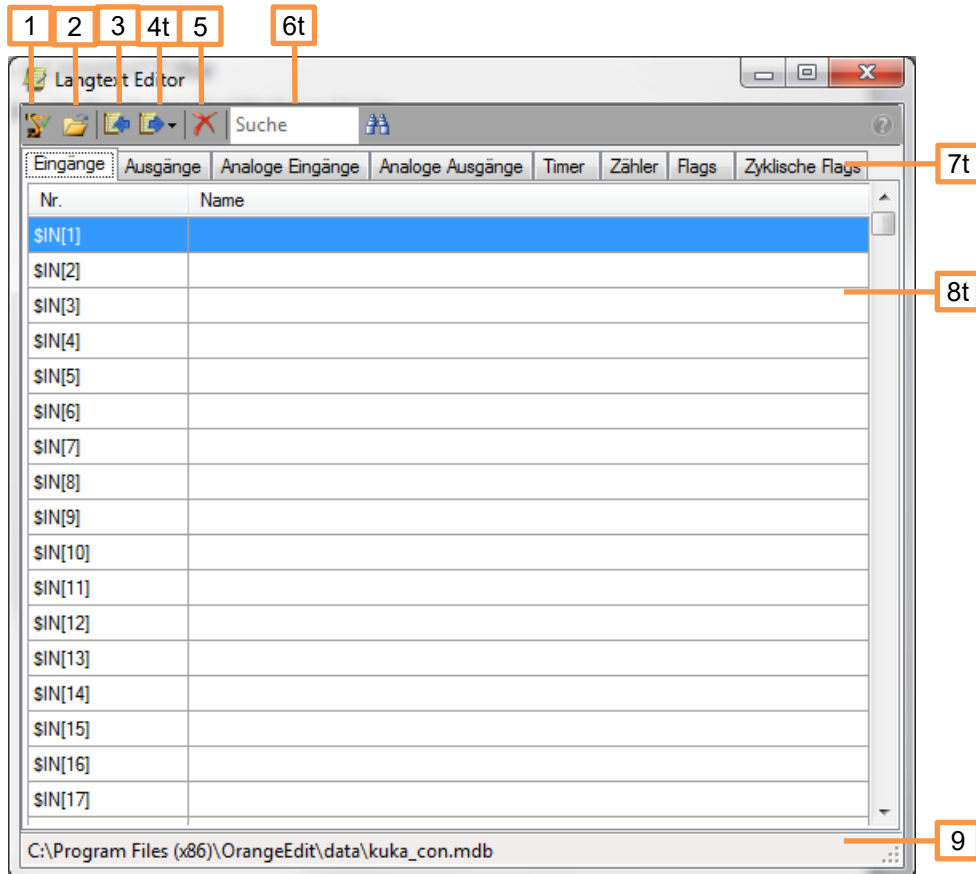
#### features

- Open and edit long text databases from a KRC1 / 2/4 robot
- Import of long texts from .csv or .txt files
- Export of long texts to .csv or .txt
- Empty (delete all) a long text database
- search

#### call

- Via menu Extras -> Long text editor
- Via the corresponding symbol in the toolbar

## presentation



1. Load long texts from the active environment
2. Load long text database (.mdb)
3. Import long texts from .txt or .csv
4. Export long texts to .txt or .csv
5. Empty database (delete all long texts from the database)
6. Search, allows you to search for a long text
7. Tabs for the different long texts
8. View and edit list of long texts
9. Path to the currently loaded long text database

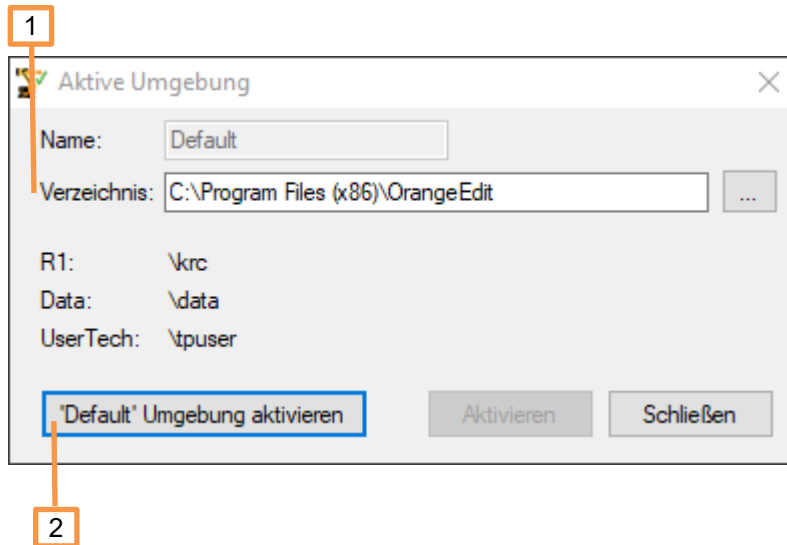
### 3.15 Active environment

Displays the current active environment or sets it to a selectable folder. This folder is searched for the "KRC" folder and then for the "R1", "DATA" and "TPUSER" folders.

#### call

- Via menu Extras -> Active Environment
- Via the corresponding symbol in the toolbar

## presentation



1. Shows the current directory of the active environment
2. Activates the default environment C: \ Program Files (x86) \ OrangeEdit

It is not necessary that all sub-folders are in the selected folder, ie if there is only one R1 folder there, then only this folder is activated.


## 4 Object browser

Displays functions, variables, structures and enumeration types. A description text is displayed for some variables. It is possible to search for values using a filter.

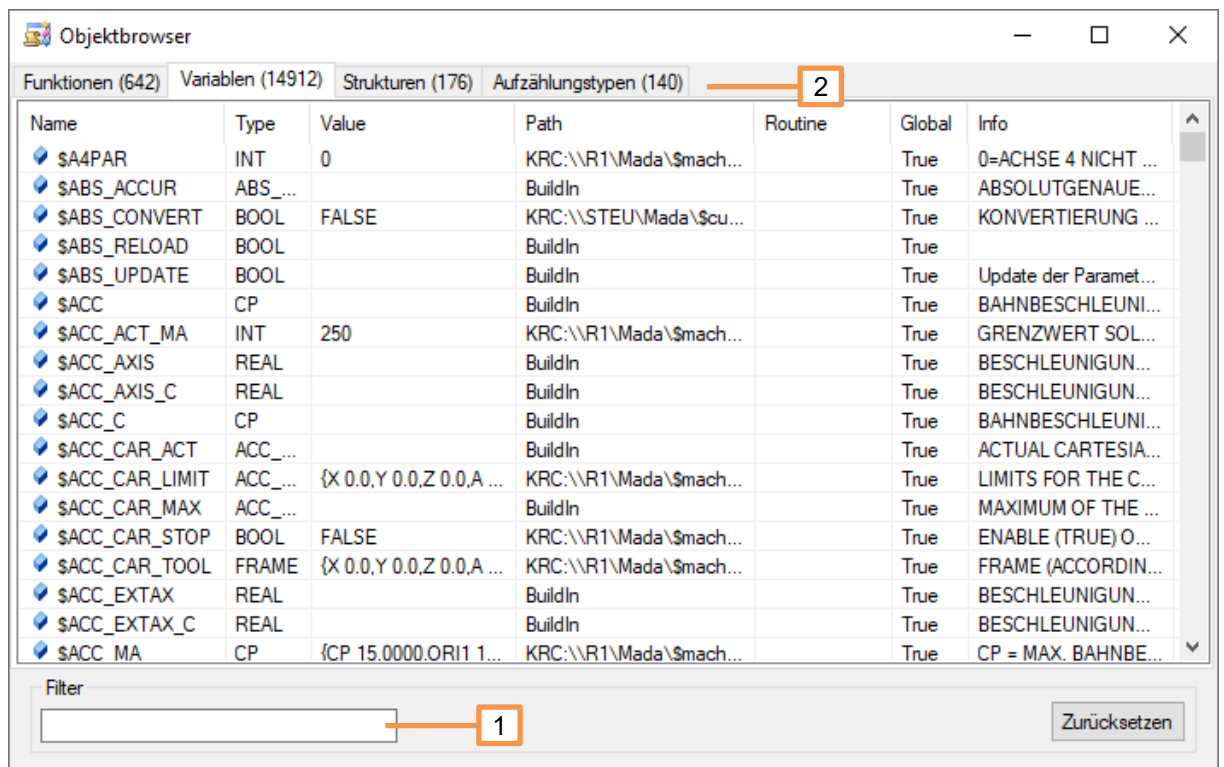
### features

- Display of system and user-defined functions, variables, structures and enumeration types
- search

### call

- Via the View -> Object Browser menu
- Via the symbol  in the toolbar

### presentation



1. Search, allows you to search for an object
2. Tabs for the different objects

## 5 Handling archives

OrangeEdit supports the opening, editing and saving of archives. ZIP archives can be loaded via the menu sequence File-> Open or by dragging and dropping onto the window with the open files or the Explorer window.

### What happens when you open it

- The archive is completely unpacked into the temporary directory (% temp%) and displayed in OrangeEdit.

### What happens when you save an archive

- All files from the previously unzipped archive are packed back into a ZIP file.



If files from the archive have been changed and saved in OrangeEdit, it is also necessary to save the archive, otherwise the changed files are not contained in the archive.

### What happens when an archive is closed

- After the dialog as to whether the archive should be saved, all files and folders that were created when opening / unzipping are deleted.

## 6 Dealing with WorkVisual projects

OrangeEdit supports the opening and editing of WorkVisual projects (\* .wvs and \* .asz). The files can be loaded via the menu sequence File-> Open or by dragging and dropping onto the window with the open files or the Explorer window.

### What happens when you open it

- The archive is completely unpacked into the temporary directory (% temp%) and displayed in OrangeEdit.

### What happens when you save an archive

- No files from the previously unzipped project are saved back into the project.



If files from the archive have been changed and saved in OrangeEdit, it is also necessary to save the changed files manually in a directory of your choice.

### What happens when an archive is closed

- A dialog appears asking you to save the changed files.



## 7 Active environment

An "active environment" refers to a directory or archive that OrangeEdit reads in and loads in full. All global variables or functions contained therein are available for selection with the syntax completion. The base and tool names for the inline forms are also loaded. If available, all UserTech inline forms are also made available as commands under "User". If the active environment is a KRC1 / 2 archive, the long text database is also loaded; this enables the creation of logic commands as an inline form, including the long texts.

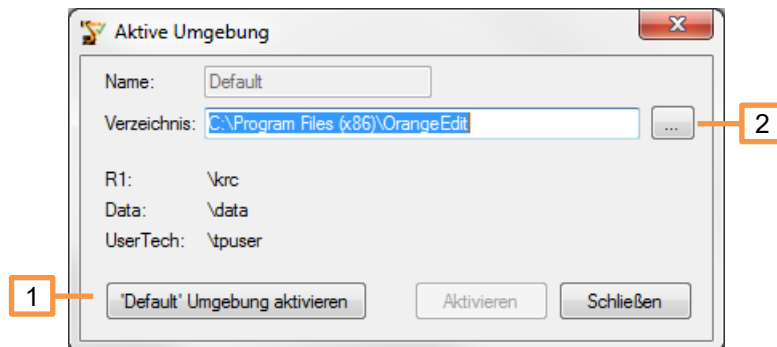
An environment consists of the following parts:

- KRC directory, contains all KRL files (\*.src, \*.dat).
- Data directory, with KRC1 / 2 contains the "KUKACON.MDB" file, a database with the respective long texts.
- UserTech, contains inline forms (.kfd files) which are listed in OrangeEdit under the "User" page.

### call

- Via menu Extras -> Active environment
- Via the corresponding symbol in the toolbar

### Representation (transformation)



1. Load "Default" environment. Loads as the active environment the KRC directory supplied with the installation (e.g. C: \ Programs \ OrangeEdit \ Krc)
2. Any directory can be loaded as an active environment here. The directory must contain at least one KRC folder, otherwise an error message will appear.

### Default environment

The default environment loads the following directories in the OrangeEdit installation folder:

- KRC: .. \ OrangeEdit \ krc \
- Data: .. \ OrangeEdit \ data \
- UserTech: .. \ OrangeEdit \ tpuser \



The folders of the default environment can be supplemented with your own inline forms and KRL files. The inline forms, as well as variables and functions, are then available in OrangeEdit.

Attention: when updating OrangeEdit, existing files will be overwritten. A copy should be made here beforehand.

### **Archive as an active environment**

A KUKA robot archive can also be defined as an active environment.

#### **Procedure (open archive)**

- Open archive in OrangeEdit.
- Answer the dialog "Set this archive as active environment?" With "Yes".

#### **Procedure (already opened archive)**

- Right click on the archive (robot symbol).
- Click "set as active environment".

## 8 Inlineforms and KRL – commands

### 8.1 Inline forms

Inline forms make programming of the robot easier. OrangeEdit provides numerous inline forms for this purpose. By double-clicking on the desired inline form, it is inserted into the program and, depending on the inline form, a corresponding data record is created in the data list, if not already available.

Inline forms existing in programs are opened by double-clicking and can be edited and saved by the user.

The data from the active environment or from open data lists are used to edit the parameters in the inline forms

#### Window inline forms



#### Call of the window ILF

Click on "Basis" in the left bottom window

#### 8.1.1 Example: Inline form "PTP movement"

A PTP movement will be inserted to the program example.

## 1. Open program

```

1  DEF Modul ( )
2  ⊕ INI
3
4  ⊕ PTP HOME Vel= 100 % DEFAULT
5
6
7  ⊕ PTP HOME Vel= 100 % DEFAULT
8
9  END

```

## 2. Set the cursor to the desired place and double click on the PTP command

^ Bewegung V8.5/8.6	
<input type="checkbox"/> PTP	8.5/8.6
<input type="checkbox"/> SLIN	8.5/8.6
<input type="checkbox"/> SCIRC	8.5/8.6
<input checked="" type="checkbox"/> PTP	8.5/8.6
<input type="checkbox"/> LIN	8.5/8.6
<input type="checkbox"/> CIRC	8.5/8.6

→ the PTP point will be inserted

```

1  DEF Modul ( )
2  ⊕ INI
3
4  ⊕ PTP HOME Vel= 100 % DEFAULT
5  |
6  ⊕ PTP P1 Vel=100 % PDAT1 Tool[0] Base[0]
7
8  ⊕ PTP HOME Vel= 100 % DEFAULT
9
10 END

```

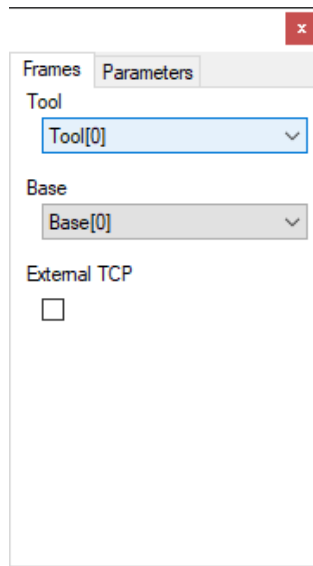
## 3. The point is created in the data list of the program with all the required parameters. These can be edited by opening the inline form. The inline form is opened with a double click on the command line.

```

1  DEF Modul ( )
2  ⊕ INI
3
4  ⊕ PTP HOME Vel= 100 % DEFAULT
5
6  ⊕     [%]    
7
8  ⊕ PTP HOME Vel= 100 % DEFAULT
9
10 END

```

- All parameters can be changed by clicking on the respective parameter fields. Additional windows may open for this purpose.



## 8.2 KUKA standard inline forms for KSS 8.5 or higher

As of control version 8.5, the KUKA standard inline forms have changed significantly. A chargeable plug-in is required so that these can be used in OrangeEdit.

The standard inline forms include:

- Movement: SPTP, SLIN, SCIRC, PTP, LIN, CIRC
- Torque monitoring: SetDefault, SetLimits, Off, SaveMax, UseDataSet
- Logic: Out, Pulse, SynOut, SynPulse, Wait, WaitFor
- Analog output: Static, Dynamic
- Comment: comment, stamp

Further information is available on our website:

<https://orangeapps.de/?lng=en&page=apps%2Forangeditilf>

## 8.3 KRL commands

OrangeEdit provides predefined KRL commands. These are inserted into the program by double-clicking.

### Calling up the window

Click on "KRL" at the left bottom window

### KRL commands window

Routine
Function
FOR loop
GOTO
IF block
INTERRUPT
LOOP
REPEAT loop
SWITCH block
TRIGGER Distance
TRIGGER Path
WHILE loop
PTP XYZ
PTP Axis
PTP Rel
Wait Sec
Wait for FALSE
Signal IN
Signal OUT
Notify message (simple)
Notify message (advanced)
Quit message (simple)
Quit message (advanced)
State message (simple)
State message (advanced)
Dialog message (simple)
Dialog message (advanced)
Loop message
Loop message reset

## 9 Syntax check

OrangeEdit has a limited syntax check. Errors found are marked with a red bar and a corresponding message is generated in the message window.

Being checked:

- Completeness of control structures
- Completeness of brackets
- Instructions outside the instruction section`
- Declaration of transfer parameters



No guarantee is given for the correctness of the check.