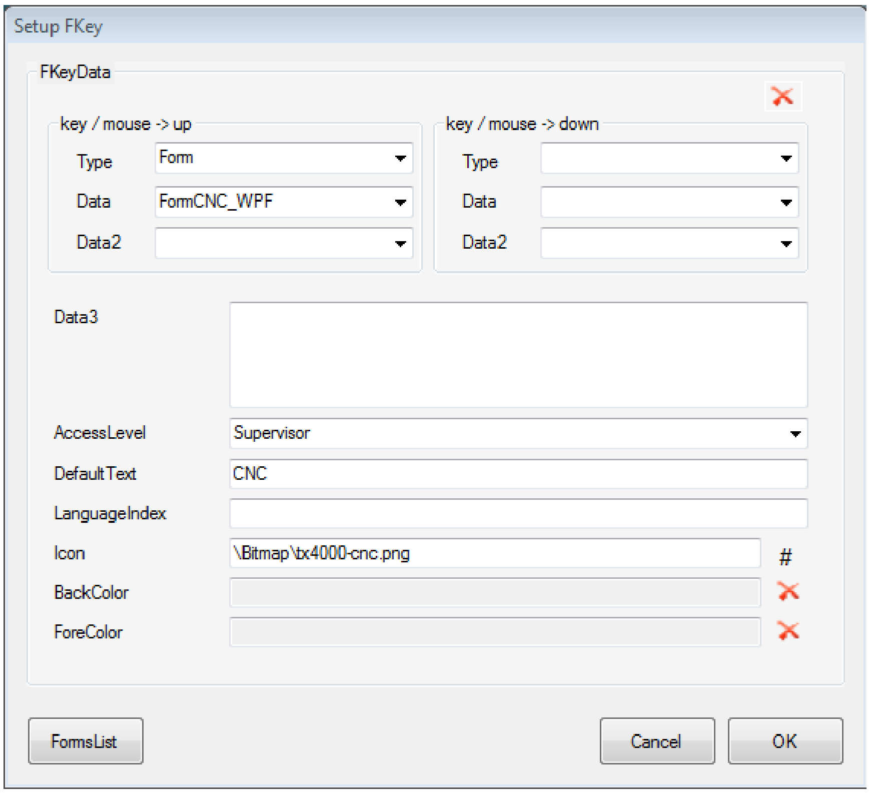
User Interface of the Menumanager

- The following functionality of configuration is only possible for userlevel "Administrator"
- To enter the configuration mode of the Menumanagers press <Ctrl> <Alt> and simultaneously click on a key or F key. All managed from Menumanager buttons are now displayed in color. By again using the above-mentioned combination of key presses Config mode is left again.
- The Config mode is invoked by a click of a button. The configuration of this key:

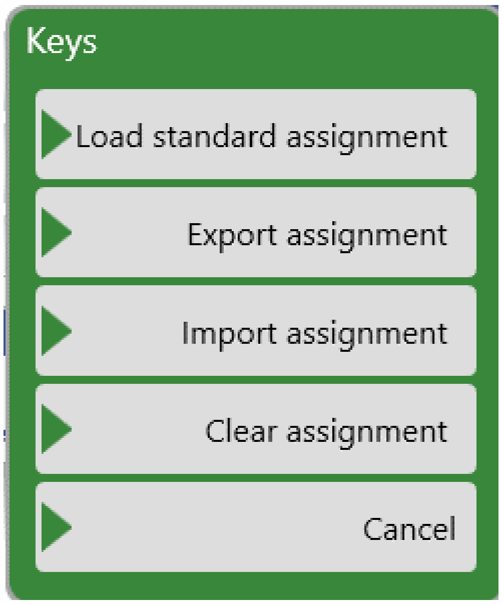


	key / mouse -> up	key / mouse -> down
Type	Form	
Data	FormCNC_WPF	
Data2		

Data3	
AccessLevel	Supervisor
DefaultText	CNC
LanguageIndex	
Icon	\Bitmap\tx4000-cnc.png
BackColor	
ForeColor	

- The entries correspond to the described XML entries. (See end of this document)
- It is possible to define events for "up" and "down"

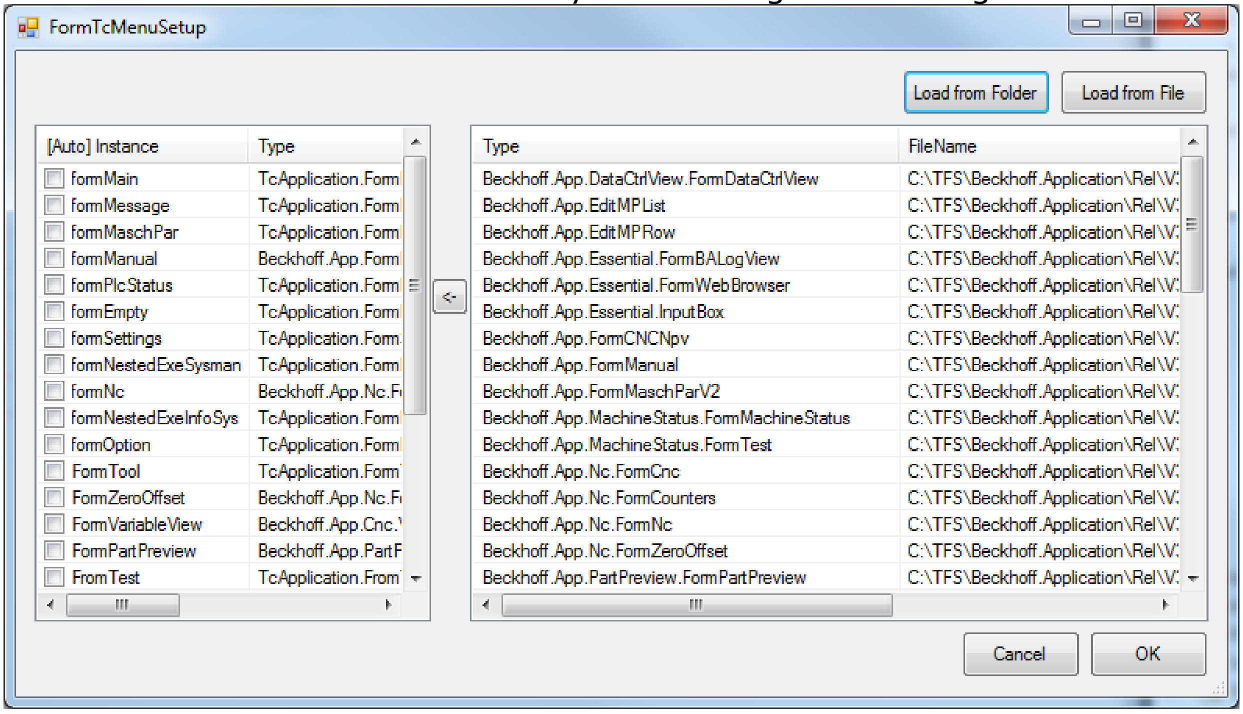
- If a key is triggered together with <Ctrl> <Shift> (clicked or pressed) then the following menu will be displayed:



- **Load standard assignment**: The default assignment defined inside the code of the form is loaded.
- **Export assignment**: The actual assignment is exported to an XML file. A dialog for the target file name is displayed.
- **Import assignment**: a previously exported assignment is loaded.
- **Clear assignment**: the complete function key assignment is cleared.
- **Cancel**: close the dialog and do nothing

FormsList

- Here the Forms which can be used by Menumanager are managed.


FormTcMenuSetup

Load from Folder | Load from File

[Auto] Instance	Type
☐ formMain	TcApplication.Form
☐ formMessage	TcApplication.Form
☐ formMaschPar	TcApplication.Form
☐ formManual	Beckhoff.App.Form
☐ formPlcStatus	TcApplication.Form
☐ formEmpty	TcApplication.Form
☐ formSettings	TcApplication.Form.
☐ formNestedExeSysman	TcApplication.Form
☐ formNc	Beckhoff.App.Nc.F
☐ formNestedExeInfoSys	TcApplication.Form
☐ formOption	TcApplication.Form
☐ FormTool	TcApplication.Form
☐ FormZeroOffset	Beckhoff.App.Nc.F
☐ FormVariableView	Beckhoff.App.Cnc.\
☐ FormPartPreview	Beckhoff.App.PartF
☐ FromTest	TcApplication.From

<-

Type	FileName
Beckhoff.App.DataCtrlView.FormDataCtrlView	C:\TFS\Beckhoff.Application\Rel\V:
Beckhoff.App.EditMPList	C:\TFS\Beckhoff.Application\Rel\V:
Beckhoff.App.EditMPRow	C:\TFS\Beckhoff.Application\Rel\V:
Beckhoff.App.Essential.FormBALogView	C:\TFS\Beckhoff.Application\Rel\V:
Beckhoff.App.Essential.FormWebBrowser	C:\TFS\Beckhoff.Application\Rel\V:
Beckhoff.App.Essential.InputBox	C:\TFS\Beckhoff.Application\Rel\V:
Beckhoff.App.FormCNCNpv	C:\TFS\Beckhoff.Application\Rel\V:
Beckhoff.App.FormManual	C:\TFS\Beckhoff.Application\Rel\V:
Beckhoff.App.FormMaschParV2	C:\TFS\Beckhoff.Application\Rel\V:
Beckhoff.App.MachineStatus.FormMachineStatus	C:\TFS\Beckhoff.Application\Rel\V:
Beckhoff.App.MachineStatus.FormTest	C:\TFS\Beckhoff.Application\Rel\V:
Beckhoff.App.Nc.FormCnc	C:\TFS\Beckhoff.Application\Rel\V:
Beckhoff.App.Nc.FormCounters	C:\TFS\Beckhoff.Application\Rel\V:
Beckhoff.App.Nc.FormNc	C:\TFS\Beckhoff.Application\Rel\V:
Beckhoff.App.Nc.FormZeroOffset	C:\TFS\Beckhoff.Application\Rel\V:
Beckhoff.App.PartPreview.FormPartPreview	C:\TFS\Beckhoff.Application\Rel\V:

Cancel | OK

- **[Auto] Instance**
  Instance name of the form. Multiple instances of a form can also be created.
  The check mark in front of the instance name indicates whether the form is automatically instantiated when starting.

- **Type**
  The class type of the instance with full namespace

- **FileName**
  The file in which to find the class.

- **Load from File, Load from Folder**
  You can create new forms and add them to the list. To this end, the file name (Load from File) (an EXE or DLL) or folder (Load from folder) is selected, in which new Forms are to be found.
  After you have selected a location, all possible forms appear in the right listview. Then with the "<-" button, a selected form may be included in

BECKHOFF

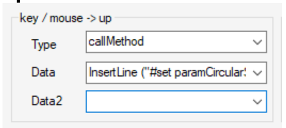
the list on the left.

To apply to the list an instance name for the Form must be specified.

XML **Definition File in the File System:**

- When starting the application, the first Form defined in TcMenu.xml automatically instantiated and displayed.
  Should the first form not displayed at startup, any other form will be set to startup form with the entry <StartupForm> true </ startup form>.
- The configuration file for TcMenu is located in the "System" folder and is called "TcMenu.xml".
- In the TcMenu.xml all possible forms are defined. Within the definition what happens when you press a function key is defined.
- Definition of a form within <section name = "formBsp"> </ section>:
  - <Type>TcApplication.FormCnc</Type>
    - Type or class definition of the form, including namespace.
  - <Filename>TcDrilling.dll</Filename>
    - File in where the type can be found. If this entry is searched in the current. Exe.
  - <CreateOnStartup>true</CreateOnStartup>:
    - true : Form is generated automatically when you start
    - false: Form is created on the first call
  - Within <FKeys> ... </ fkeys> the function key assignment is defined.
- Definition of a function key: <Key index="1"> </ Key> (eg: function key 1)
  - <DefaultText> Workpiece programming </DefaultText>
    - Default Label
  - <TcLM-Index> Menu - part programming </TcLM-Index>
    - Index for the Language Manager (for the language selection). If under index no text stored, it will be created with the default label.

- <Type>Form</Type>
  - Type of Post (The data will specify in <DATA>:
    - Form: a form that is defined in the XML file. The content of "Data2" is specified as a parameter in the constructor. (A "String parameter" in the constructor must be defined)
    - startProcess: starts an external application
    - Back: Jump back in the call stack
    - Close: Jump back in the call stack and close the current form.
    - ExitProgram: end application
    - ShutDown: End application and shutdown computer
    - CallMethod:
      - Call a method of the current Form.
      - It can be called with methods with no parameters and methods with exactly one parameter, the return value is void. The list is displayed in Data1.
      - When the optional parameter of this is given in parentheses after the method name. In addition, another parameter can then be defined in the method that receives the key data:
        ```
        public void mibtest(int art, IBAFKeyData fkeyData)
        ```
      - More complex parameters can be put in quotation marks:

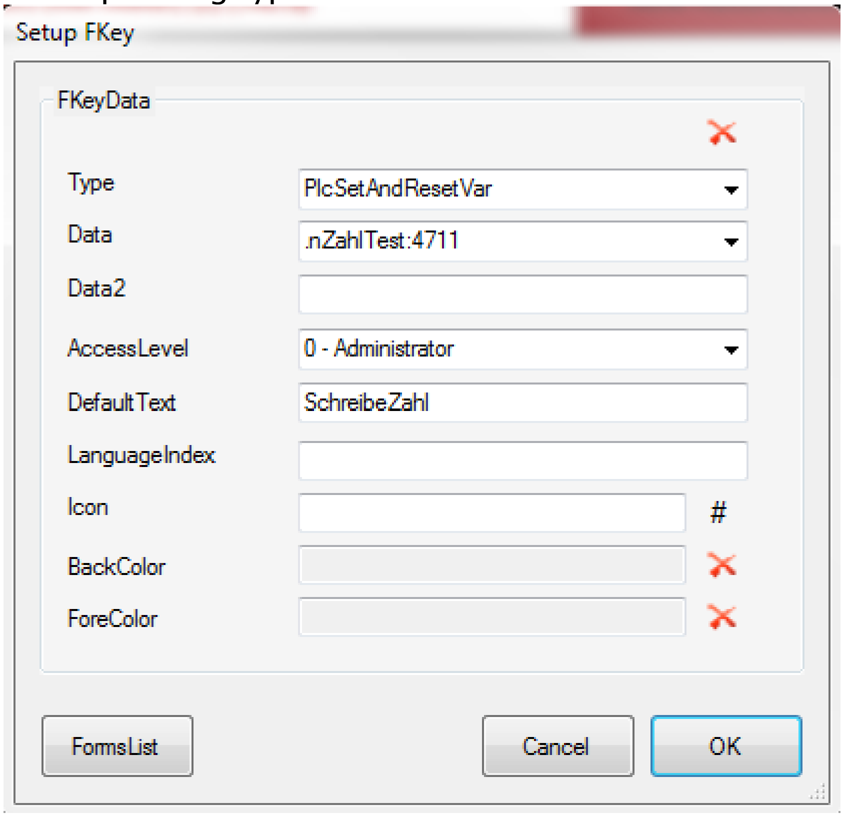

key / mouse -> up

Type	callMethod
Data	InsertLine ("#set paramCircular!
Data2	

```
InsertLine ("#set paramCircularSmoothing(R57)#")
```

BECKHOFF

- PlcToggleVarAndFeedback or PlcSetAndResetVar:
    - PlcToggleVarAndFeedback: The "BOOL" variable from Data1 is toggled when pressed. If the variable name with a ":" separated a given value to be written, it is written to the PLC in the corresponding typed variable.



Setup FKey

FKeyData

Type	PlcSetAndResetVar
Data	.nZahlTest:4711
Data2	
AccessLevel	0 - Administrator
DefaultText	SchreibeZahl
LanguageIndex	
Icon	
BackColor	
ForeColor	

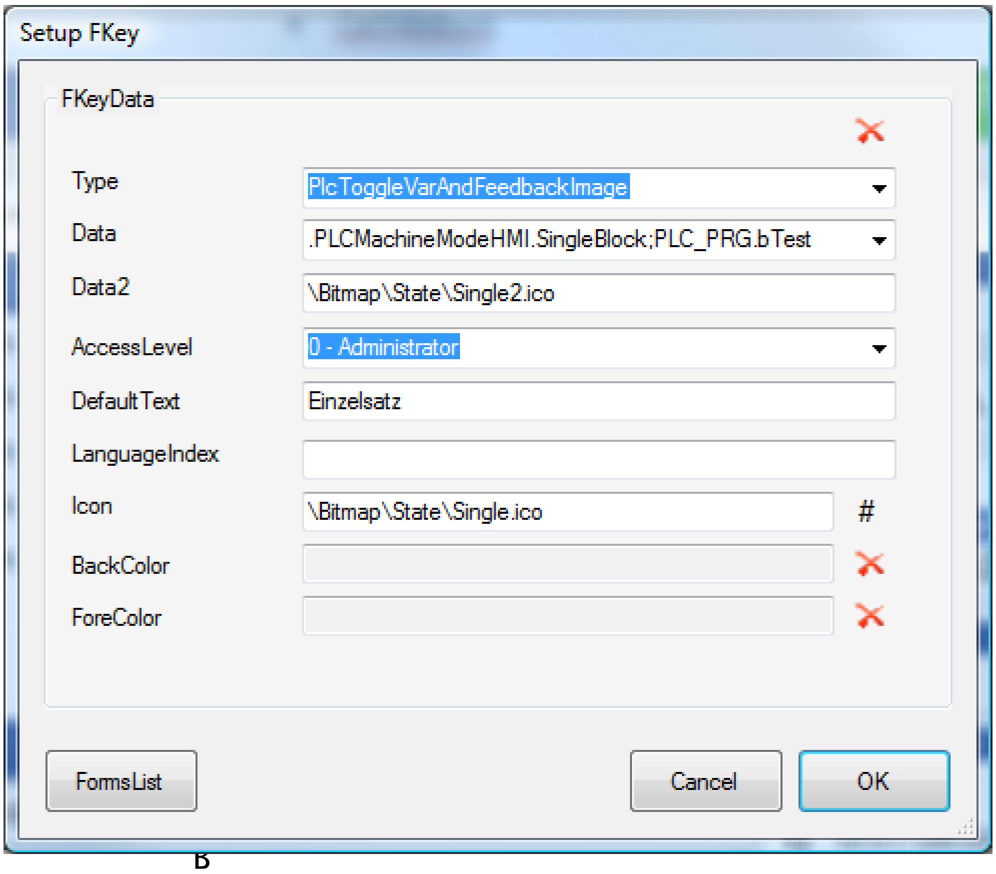
    - PlcSetAndResetVar: If Data1 is set to a bool variable, when you release the button, the variable is reset.
        - If Data2 is of type Bool, True and False are shown each as a different color of the button.
          The colors are in the settings under "Color Theme - color1" - set and "color2 Color Theme".

- If Data2 is string type, the button label is taken from the PLC.
- If Data2 is DINT type, the DINT value is interpreted as a color value and represented the key in the appropriate color.
  The color is determined as follows:
  nFarbe = 16#AARRGGBB, in which
    - AA = Alpha (Full Color = FF)
    - RR = Red
    - GG = Green
    - BB = Blue
    - E.g.: Full Green:
      nFarbe = 16#FF00FF00

- BlockUserInput
    - It will block the entire input (keyboard, mouse, touch) for a period of 5 seconds. In data, the time (s) can be adjusted. This functionality can be used for "cleaning" a touch screen.

- PlcToggleVarAndFeedbackImage
    - The "BOOL" variable from Data1 is toggled on keypress.
    - The image is specified in Data2 and is displayed if ToggleVariable is a "TRUE".
    - If you wish the toggling variable and the variable to be displayed to be different, they can be specified, separated by semicolons in Data1:

(E.g.: Toggling. PLCMachineModeHMI.SingleBlock, display of PLC_PRG.bTest)


Setup FKey

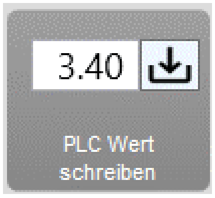
FKeyData

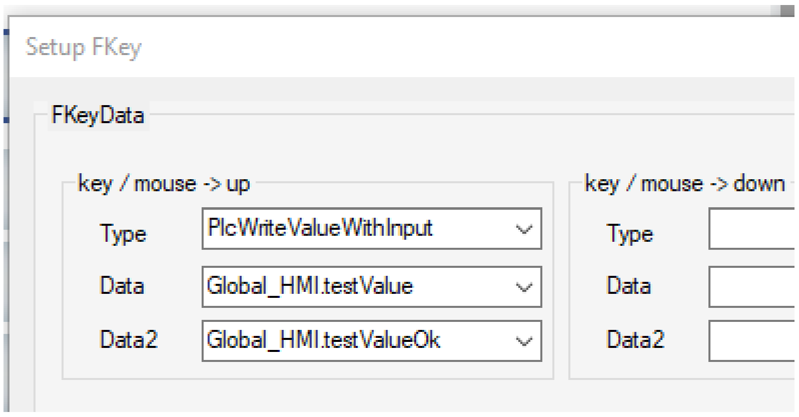
Type	PlcToggleVarAndFeedbackImage
Data	.PLCMachineModeHMI.SingleBlock;PLC_PRG.bTest
Data2	\Bitmap\State\Single2.ico
AccessLevel	0 - Administrator
DefaultText	Einzelsatz
LanguageIndex	
Icon	\Bitmap\State\Single.ico
BackColor	
ForeColor	

B

- <Data>formWorkpieceInput</Data>
  - Data belonging to the entry <Type> (for example, the form name in the type form or the method name for type CallMethod.
- <Icon>\\Bitmap\\CNCIcon.bmp</Icon>
  - Icon for the function key
- <Access-Level>0</Access-Level>
  - Access level from which the button is enabled:
    0 = Administrator, 1 = SuperVisor, …
- <ForeColor>red </ForeColor >
  - Foreground color and font color of the button
- <BackColor>red </BackColor>
  - Background color of the button

- PlcWriteValueWithInput
    - Shows a dialog.
      In this dialog, you can see the actual value of the LREAL PLC variable from "data1".
      After changing it, the key <ENTER> transfers it to PLC.
    - <ESC> cancels without transfer.
    - If data2 is a BOOL variable in PLC, every successful transfer will write TRUE this this variable.




Setup FKey

FKeyData

key / mouse -> up

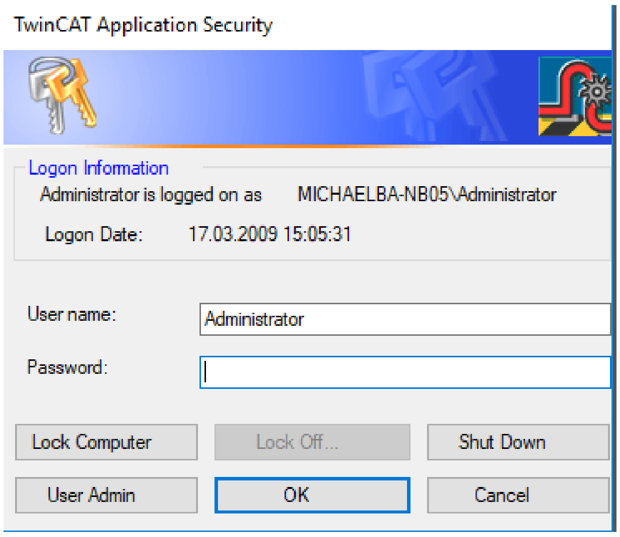
Type	PlcWriteValueWithInput
Data	Global_HMI.testValue
Data2	Global_HMI.testValueOk

key / mouse -> down

Type	
Data	
Data2	

BECKHOFF

- LogonDialog
    - Logon Dialog is called:


TwinCAT Application Security

Logon Information

Administrator is logged on as MICHAELBA-NB05\Administrator

Logon Date: 17.03.2009 15:05:31

User name: Administrator

Password:

BECKHOFF

Influence HMI keys from the PLC

Connect any key to PLC variables to remote control this key from the PLC. To do this, defines the function and PLC variable name in the field data3:

- $(enable)VariablenameEnable:
  BOOL variable to enable or disable the key.

- $(text)VariablennameText:
  String variable to change the text

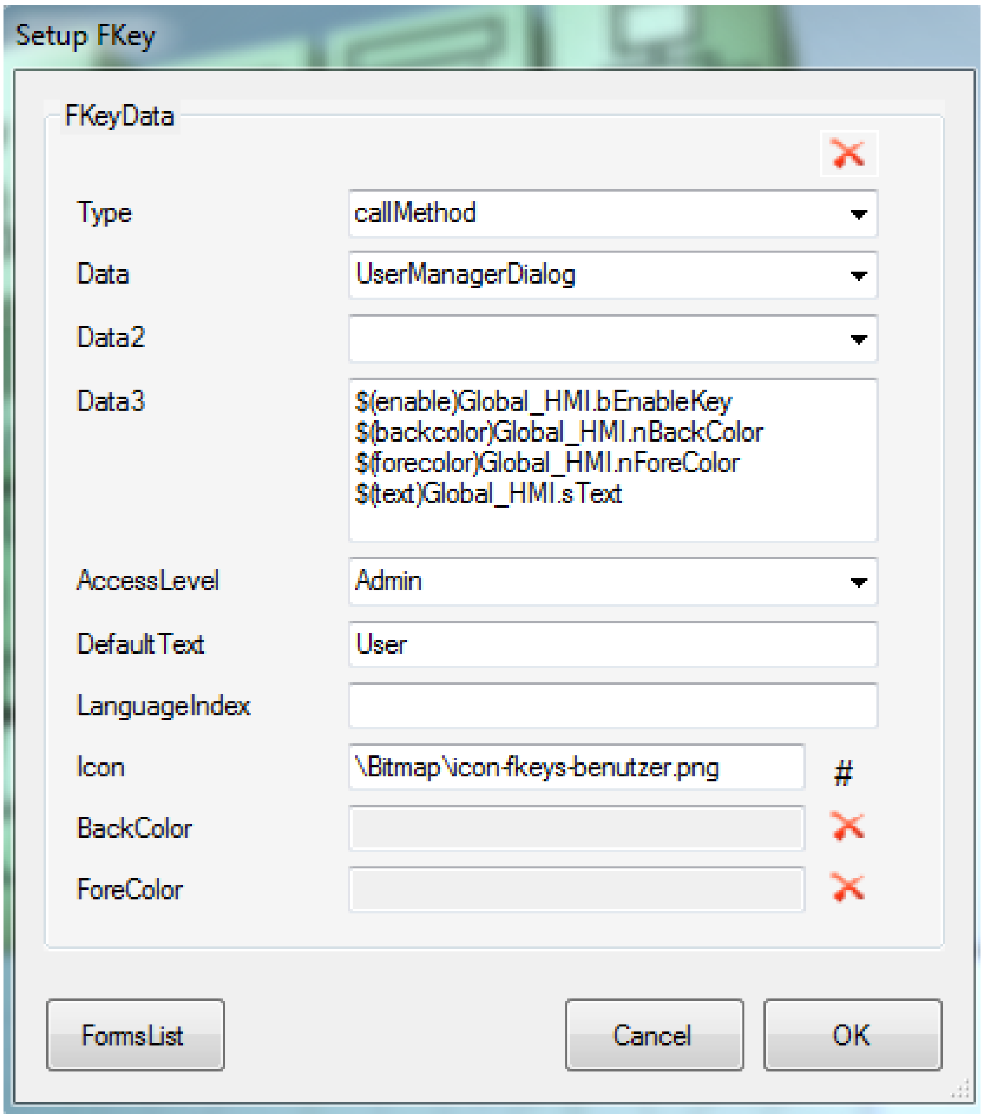
- $(backcolor)VariablenameBackcolor
  UDINT variable to defined the back color of the key
  (16#AARRGGBB)

- $(forecolor)VariablenameForecolor
  UDINT variable to defined the fore color of the key
  (16#AARRGGBB)

- $(trigger)VariableNameTriggerKey
  BOOL variable to simulate a key press or key release
  Rising edge -> key is pressed
  falling edge -> key is released

- $(hide)VariablenNameHideKey
  BOOL variable to hide the key

The HMI connects “OnChange” to the variables.
Example:



Type	callMethod
Data	UserManagerDialog
Data2	
Data3	$(enable)Global_HMI.bEnableKey $(backcolor)Global_HMI.nBackColor $(forecolor)Global_HMI.nForeColor $(text)Global_HMI.sText
AccessLevel	Admin
DefaultText	User
LanguageIndex	
Icon	\Bitmap\icon-fkeys-benutzer.png
BackColor	
ForeColor	