Orange Apps

myHMI

Custom HMI via XML

Version 1.1

Getting Started

Status: 22/02/2018, Version 1.1

Orange Apps GmbH
Arnikaweg 1
87471 Durach
Germany
www.orangeapps.de

History of document versions

Version	Date	Author	Reason for change / Remarks
1.0	14/10/2016	Christian Mayer	First position
1.1	20/02/2018	Christian Mayer	Implementation control “Picture”

Content

1 Introduction

1.1 Product Description

myHMI can be used as user-specific HMI for the display and manipulation of all known in the robot system KRL variables. The intention is to display BOOL, INT, REAL, ENUM and CHAR array (string) variables using graphical elements such as text boxes, switches, LEDs and drop controls.

The creation of an HMI is done via an XML file in which the specific entries are stated by using a standard text editor. A plugin interprets the entries in the XML file, and represents the elements in tabular form on a HMI. The user thus does not require any knowledge of programming of plugins and HMIs.

Any number of XML files and so any number of HMI's can be produced. Each HMI is called via a menu item in the main menu of the robot. The type of the display can be chosen between a half or full page. Like any the HMI plugin embeds seamlessly into the robot system. All of the standard menu and control items remain fully operable.

To enable a thematic separation within an HMI, the content can be distributed to a maximum of 5 tab pages. Each tab can display 32 items.

Each element can be dynamically linked to a user level.

The entire HMI supports multiple languages, so that a dynamic language switching is possible.

Changed values by the operator are recorded in the logbook KUKA. (Diagnosis / log).

For the creation of the menu entries of each HMI there’s a menu assistant available. Thus, no knowledge of the menu creation is necessary.

1.2 Characteristics

- HMI for display and manipulation of KRL variables
- Displayed content is defined using XML
- Content is thematically presented with tabs
- Input fields are displayed in tabular form with one by one with up to 3 columns
- Controls: Picture, Switch, Button, Text, Number, LED, Drop-down, Slider, Progressbar, Label, Headline
- Input fields can be enabled by specifying optional dependencies (user group, Submit u. Program status, drives enable, operating mode, etc.)
- Contents can be presented in several languages using KXR files
- User input is stored in the logbook
- The number of displayed HMI's is theoretically unlimited
- Up to 5 tabs can be defined each HMI
- Up to 32 units can be defined for each tab
- Menu assistant

This documentation is intended to enable a quick introduction to the product myHMI.

Using the included HMI “DemoMyHMI” this document shows how to create a new HMI.

Deeper information is provided in the documentation "OrangeApps.myHMI_User_Manual_*_en.pdf".

1.3 Scope of delivery

The software is delivered as technology package for installation directly on the robot (additional software). This includes all the necessary components for installation and operation:

- Plugin
  - myHMI.dll
  - SmartHMI.exe.myHMI.config
  - myHMI.kxr
- User documentation for the installation and operation of the software

- Plugin menu assistant
  - myHMIMenuConfigurator.dll
  - SmartHMI.exe.myHMIMenuConfigurator.config
  - myHMIMenuConfigurator.kxr

To help users get started, in the setup package a sample HMI is included with the following files:

- DemoMyHMI.xml→ contains examples of using tabs and controls
- DemoMyHMI.kxr→ Example of creating a speech database for Multilingualism
- Several images in the folder C:KRC\User\myHMI\PicsDemo

1.4 Application area / environment

The software runs on all KUKA robots with KSS8.2 / 8.3 and 8.5 without CPC protection.

1.5 CPC

If the software should be used on robots with CPC protection, a CPC certificate is required before installation. In this case please contact us.

2 Installation

The software is installed by the additional software option. This is located in the main menu under commissioning.

Steps

- Copy the software on the robot
- In the main menu choose Commissioning →*Additional Software*
- Select *New Software, configure the path to the setup folder* and install it

→ The software includes a sample HMI (DemoMyHMI.xml)

These files are installed for the sample HMI:

Folder	Files	Function
C:\KRC\SmartHMI	SmartHMI.exe.DemoMyHMI.config	Menu Item
C:\KRC\DATA	DemoMyHMI.kxr	Language database for Demo HMI
C:\KRC\USER\myHMI	DemoMyHMI.xml	HMI
C:\KRC\USER\myHMI\PicsDemo	Several images	

The sample HMI is fully operational and can be used as a basis for further HMI's.

3 Create your own HMI called "myFirstHMI"

To create your own HMI it's recommended to copy an existing one and modify according to your own whishes and needs.

Steps to create your own HMI

- Copy the file "DemoMyHMI.xml" in C:\KRC\USER\myHMI to myFirstHMI.xml
- Open the menu assistant and create your menu entry (see chapter 4)
- Edit myFirstHMI.xml on your laptop (or directly on the robot, keyboard should be attached) and modify it to your needs and wishes
- Copy the file back into the folder c:\KRC\User\myHMI
- Open the HMI from the menu → finished

For creating and editing XML files, "Notepad ++" is recommended. Firstly, the XML content is displayed in color legible and other basic XML error of the contained in Notepad ++ XML parser are already detected when saving the file.

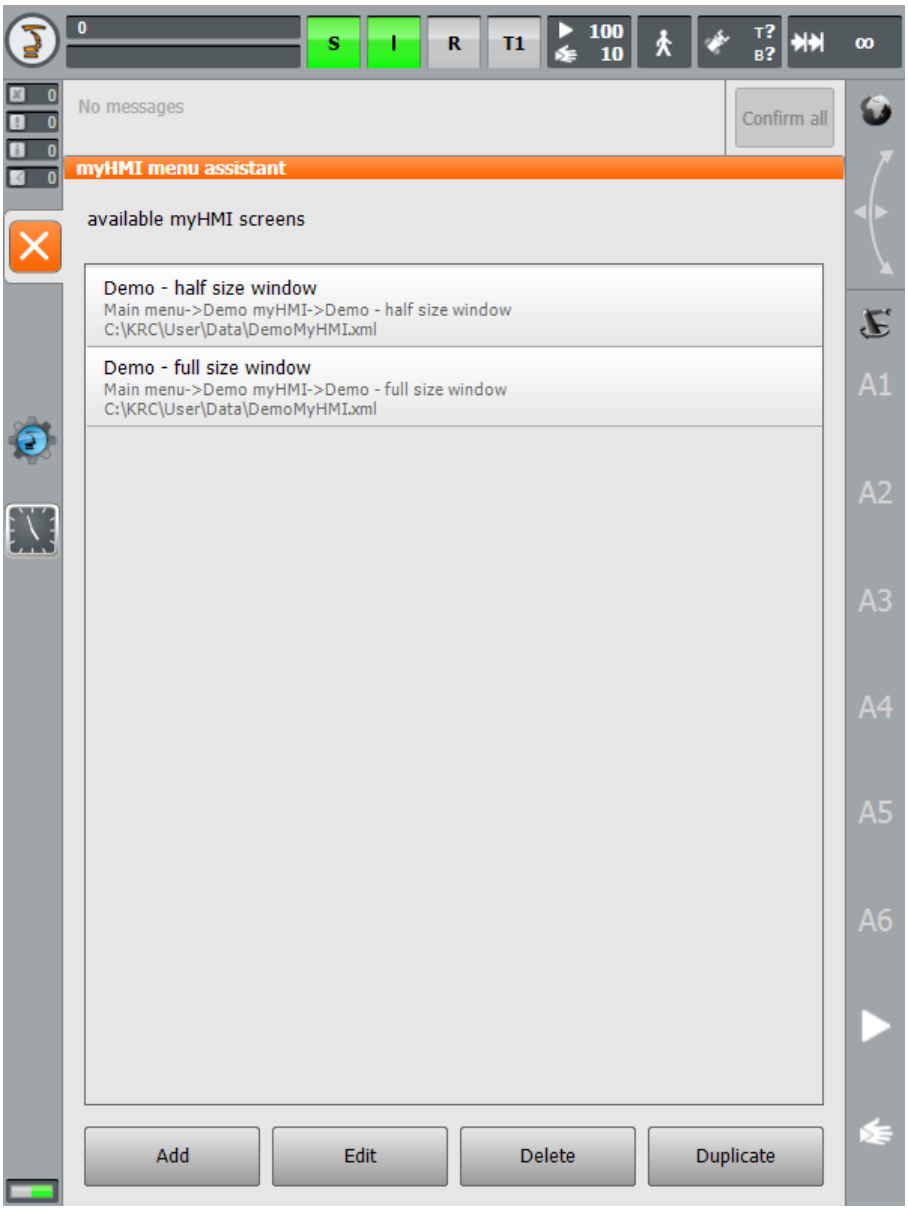
In order to present an updated XML file in myHMI, it is sufficient to close the display window of the HMI and to re-open. Changes in a KXR language file require restarting the SmartHMI, or alternatively a cold start of the robot controller.

4 Create menu entry with menu assistant

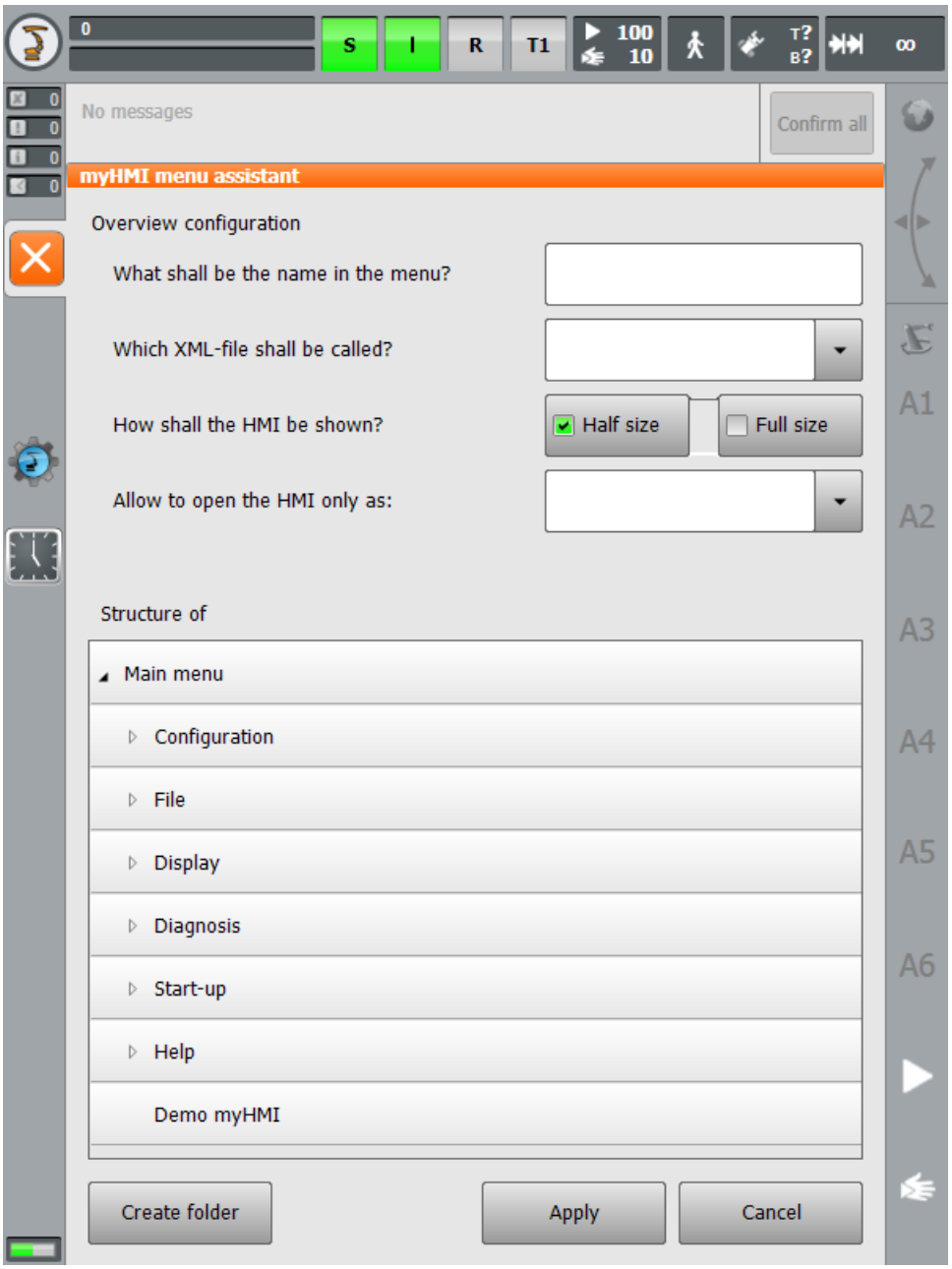
The menu assistant can be used to easily create a menu entry for your HMI.

Steps

- Log into the robot as ***Administrator***
- Open the menu assistant in the menu ***Configuration → myHMI menu assistant***
- Press ***Add*** to create a new menu entry
- Modify all entries and press ***Save***
- Close the menu assistant



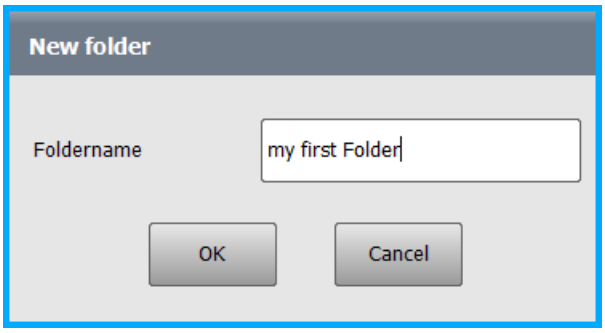
➔ Press ***Add***



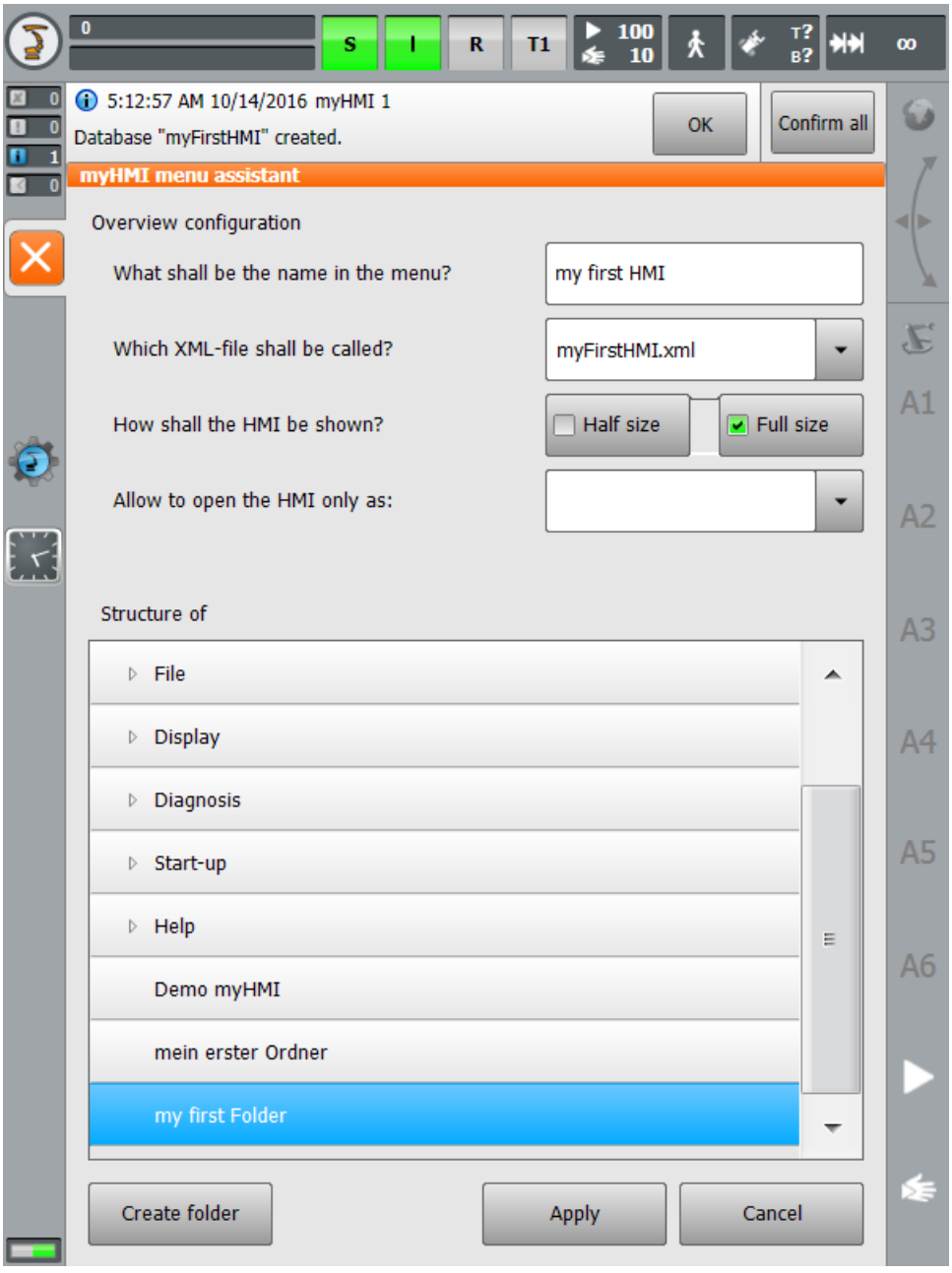
Steps to create menu entry

- Specify a name for the menu entry
- Select “myFirstHMI.xml” from the dropdown box
- Choose the height of HMI (half- or full size window)
- Choose the operator level needed to open the HMI
- Choose in which folder the menu entry shall be located. Press ***Create folder*** if you want to create a new folder in the menu
- Press ***Apply***

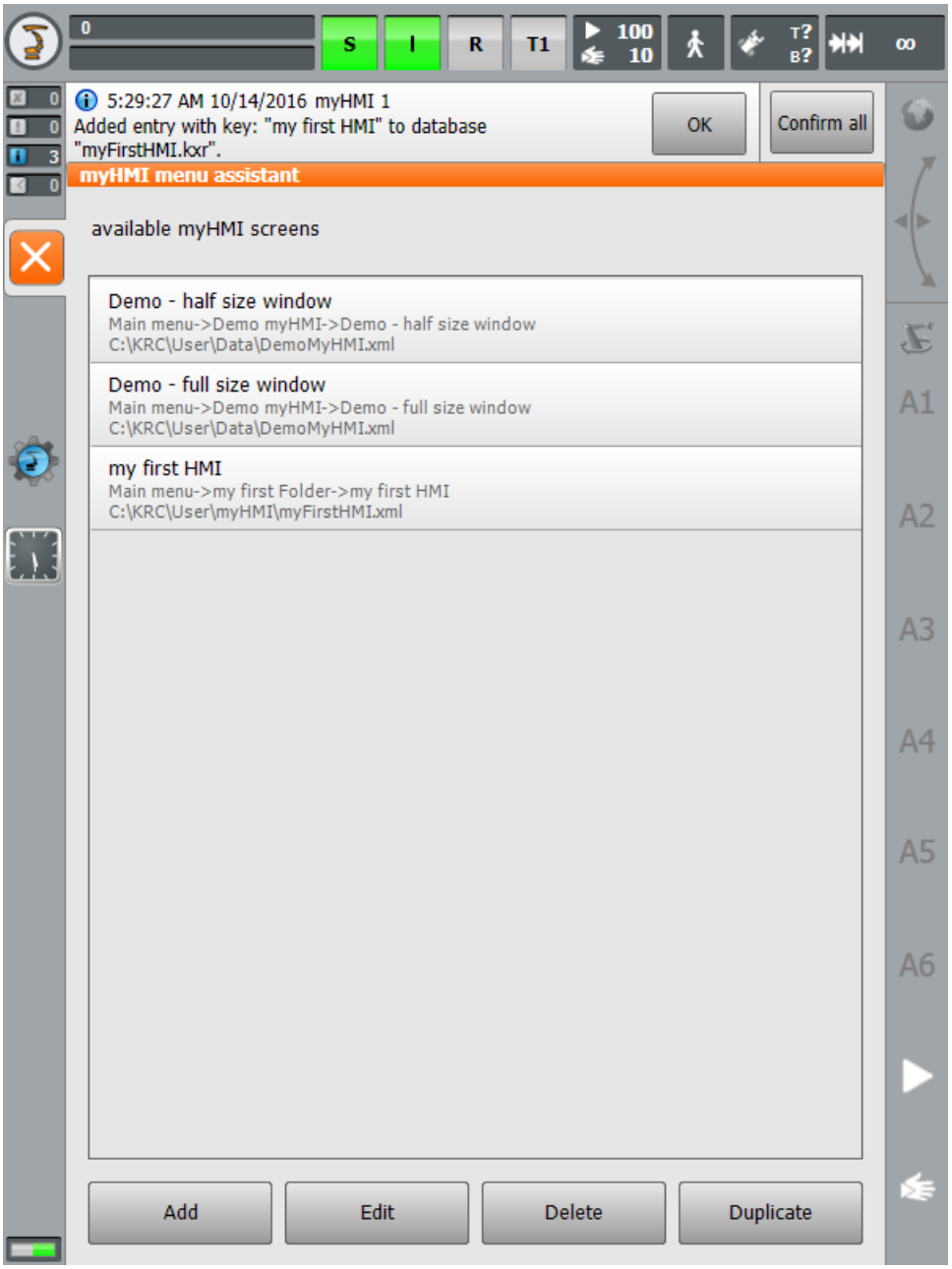
Before creating a new folder be sure that the name of the xml-file is already selected, due to the fact, that the software automatically creates a language database with a name equal to the name of the xml-file.



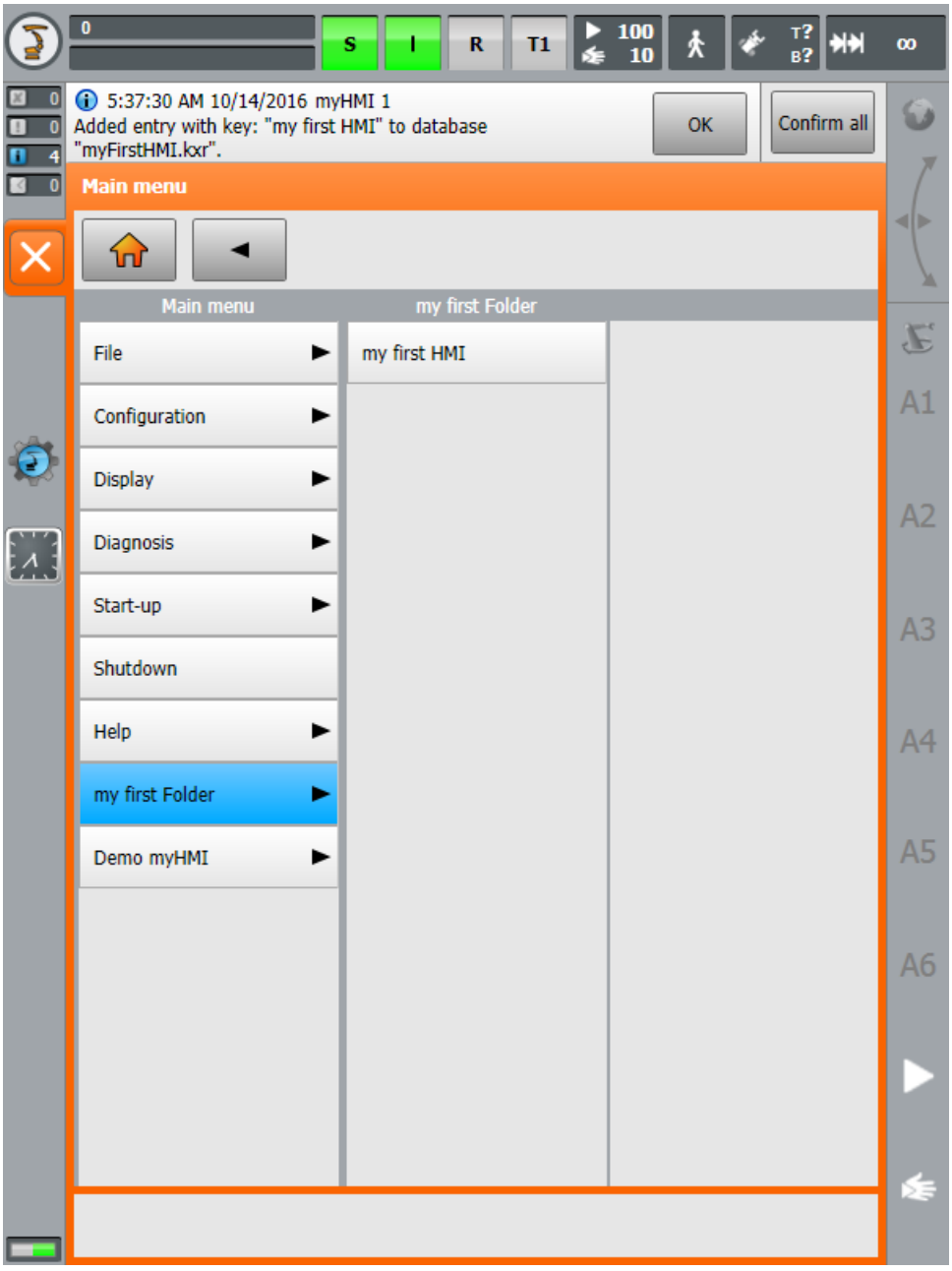
After pressing ***Ok*** the software creates the menu entry and automatically creates a language database (if not already existing). This database can be used for multilingualism.



➔ Press ***Apply***


0
S
I
R
T1
100
10
T?
B?
∞
5:29:27 AM 10/14/2016 myHMI 1
Added entry with key: "my first HMI" to database "myFirstHMI.kxr".
OK
Confirm all
myHMI menu assistant
available myHMI screens
Demo - half size window
Main menu->Demo myHMI->Demo - half size window
C:\KRC\User\Data\DemoMyHMI.xml
Demo - full size window
Main menu->Demo myHMI->Demo - full size window
C:\KRC\User\Data\DemoMyHMI.xml
my first HMI
Main menu->my first Folder->my first HMI
C:\KRC\User\myHMI\myFirstHMI.xml
Add
Edit
Delete
Duplicate
A1
A2
A3
A4
A5
A6

Entry in the main menu:



5 Modify your HMI

The HMI is represented by up to 5 tabs and up to 32 controls on each tab. The order of the tabs and the controls in the xml-file represent the order of the tab and the controls shown in the HMI.

To make a modified xml file known to the system, it is sufficient to reopen the HMI or to change the tab if HMI is already open.

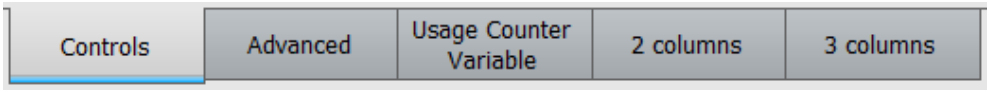
5.1 Create tab

The tab are set by the element <Group Text = "......">.

Example XML

```
<Configuration Text = "MyFirstHMI">
<Group Text = "Controls">
  . . .
</Group>
<Group Text = "Advanced">
  . . .
</Group>
   . . .
</Configuration>
```

Example Display



If only one tab is defined within an XML file no tab bar is displayed in the HMI. Instead, the space can be used for more controls.

5.2 Create controls

The controls are the set by <Control "=" Type ... "text =" ... "KrlVar =" ... ">.

To define controls within a tab, the entry must be located within the respective node

```
<Group Text = "Controls">
  <Control Type="Led" Text="Led1" KrlVar="$Flag[1]"/>
</Group>
```

Type = "...." → specifies the type

Text = "...." → specifies the description text

KrlVar = "...." → sets the associated variable KRL

5.2.1 Overview Available Controls

The following controls are available:

- Picture
- Number
- LED
- Switch
- Checkbox
- Button
- DropDown
- Text
- Label
- Progressbar
- Slider
- Headline

These controls can optionally be influenced by a variety of arguments in function and appearance

Example for a control of the type "Number" to display INT or REAL variables

```
<Control Type="Number"  Text="Number control" KrlVar="$OV_PRO"/>
```

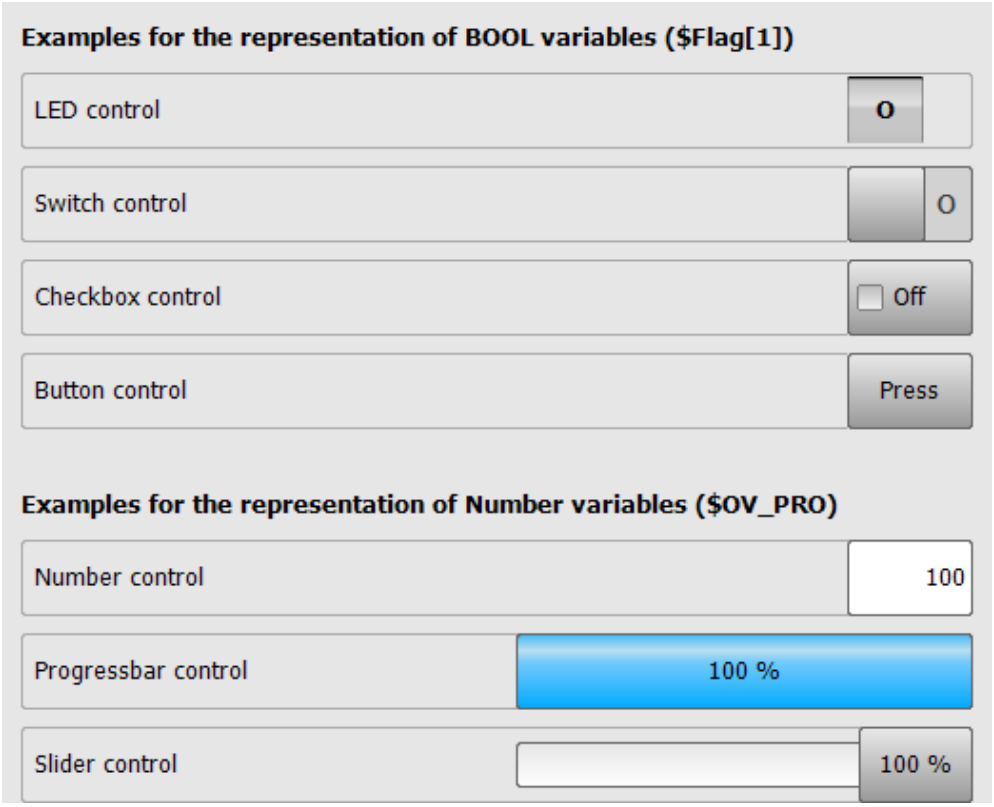
Example for a control of the type "LED" to display the status of an Input

```
<Control Type="LED" Text="LED control" KrlVar="$IN[1]"/>
```

More examples

```
            <Control Type="Headline" Text="Examples for the representation of BOOL
variables ($Flag[1])"/>
            <Control Type="Led" Text="Led control" KrlVar="$Flag[1]"/>
            <Control Type="Switch" Text="Switch control" KrlVar="$Flag[1]"/>
            <Control Type="Checkbox" Text="Checkbox control" KrlVar="$Flag[1]"/>
            <Control Type="Button" Text="Button control" TextButton="Press"
KrlVar="$Flag[1]"/>
            <Control Type="Headline" Text=" Examples for the representation of Number
variables ($OV_PRO)"/>
            <Control Type="Number" Text="Number control" KrlVar="$OV_PRO"/>
            <Control Type="Progressbar" Text="Progressbar control" KrlVar="$OV_PRO"
Format="0 \%" Min="0" Max="100"/>
```

```
            <Control Type="Slider" Text="Slider control" KrlVar="$OV_PRO" Format="0 \%"
Min="0" Max="100"/>
```



5.2.2 Optional arguments for control and display elements

For each control are further arguments are available. These arguments effect the appearance and behavior of each control.

Description of all possible arguments

Argument	Description	Optional	Default value
Alignment	Control Label: Sets the caption text left, center, right) Control Picture: Aligns the picture (left,center,right)	Yes	Label:left Picture: right
AreYouSure	True=Confirmation on value change (dialogue)	Yes	False
Border	Controls if a frame shall be drawn around a picture (with frame=True)	Yes	TRUE
Color0	Color of the LED in the control state False Possible values: Grey, Green, Red, Yellow	Yes	Gray
Color1	LED color of the control “Led” at the TRUE state Possible values: Grey, Green, Red, Yellow	Yes	Green
ColSpan	Number of columns spanned by the element	Yes	1
Description	When clicking on the control additional description text is displayed	Yes	
Format	Formatting of INT and REAL values	Yes	
KrlVar	Linked KRL variable	No	
Max	Maximum allowed value	Yes	
Min	Minimum allowed value	Yes	
ModeOP	Operation mode from which the element is editable	Yes	31
Module	Module / KXR file for multilingual content	Yes	value from *group*
NeedDrivesReady	Editability of the element is dependent on the state of the drives	Yes	False
NeedSafetySwitch	Editability of the element is dependent of the state of the enabling switch	Yes	False
Negate	Invert a 16oolean variable	Yes	False

Path	Specifies the name and path of a picture	No	
ProState0	Editability of the element is dependent of the state of the submit interpreter	Yes	63
ProState1	Editability of the element is dependent of the state of the program interpreter dependent	yes	63
Step	Increment for up / down button	Yes	
Text	Label text or key for the element	No	
Text0	Labeling of the control button "checkbox" at state False	Yes	From
Text1	Labeling of the control button "checkbox" state at True	Yes	An
TextButton	Labeling of the control "button"	Yes	
UserLevelEdit	User level from which the element is editable	Yes	0
User Level Visible	User level from which the element is visible	Yes	0
Value	Value of an entry in control "DropDown". The selected value is given to the variable of the argument "KrlVar".	No	
Width	Specifies the width of a picture (pixel). The height is scaled proportial	Yes	

Arguments / element array

Argument	<Configuration>	<Group>	Number	Led	Switch	Checkbox	Button	Slider	Progressbar	Drop	Text	Label	Headline	
Alignment	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	O	O
AreYouSure	n/a	n/a	O	n/a	O	O	O	O	n/a	O	O	n/a	n/a	n/a
Border	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	O
Color 0/1	n/a	n/a	n/a	O	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a
Columns	n/a	O	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a
ColSpan	n/a	n/a	O	O	O	O	O	O	O	O	O	O	O	O
Description	n/a	n/a	O	O	O	O	O	O	O	O	O	O	n/a	n/a
Format	n/a	n/a	O	n/a	n/a	n/a	n/a	O	O	n/a	n/a	n/a	n/a	n/a
KrlVar	n/a	n/a	X	X	X	X	X	X	X	X	X	X	n/a	O
Max	n/a	n/a	O	n/a	n/a	n/a	n/a	O	O	n/a	n/a	n/a	n/a	n/a
Min	n/a	n/a	O	n/a	n/a	n/a	n/a	O	O	n/a	n/a	n/a	n/a	n/a
ModeOP	n/a	n/a	O	n/a	O	O	O	O	n/a	O	O	n/a	n/a	n/a
Module	O	O	O	O	O	O	O	O	O	O	O	O	O	O
NeedDrivesReady	n/a	n/a	O	n/a	O	O	O	O	n/a	O	O	n/a	n/a	n/a
NeedSafetySwitch	n/a	n/a	O	n/a	O	O	O	O	n/a	O	O	n/a	n/a	n/a
Negate	n/a	n/a	n/a	O	O	O	O	n/a	n/a	n/a	n/a	n/a	n/a	n/a
Path	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	X
ProState0	n/a	n/a	O	n/a	O	O	O	O	n/a	O	O	n/a	n/a	n/a
ProState1	n/a	n/a	O	n/a	O	O	O	O	n/a	O	O	n/a	n/a	n/a
Step	n/a	n/a	O	n/a	n/a	n/a	n/a	O	n/a	n/a	n/a	n/a	n/a	n/a
Text	X	X	X	X	X	X	X	X	X	X	X	X	X	O
Text0/1	n/a	n/a	n/a	n/a	n/a	O	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a
TextButton	n/a	n/a	n/a	n/a	n/a	n/a	O	n/a	n/a	n/a	n/a	n/a	n/a	n/a
UserLevelEdit	n/a	n/a	O	n/a	O	O	O	O	n/a	O	O	n/a	n/a	n/a
UserLevelVisible	n/a	n/a	O	O	O	O	O	O	O	O	O	O	O	O

Value	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	X	n/a	n/a	n/a	n/a
Width	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	n/a	O

X: Specification mandatory | O: Optional specification | n/a:not available

Example 1

```
<Control Type="Led" Text="Led with colours Red and Green" Description="choose between
these colours: Gray, Green, Yellow, Red" KrlVar="$FLAG[1]" Color0="Red"
Color1="Green"/>
```

Example 2

```
<Control Type="Number" Text="optional up/down buttons and min/max Value"
KrlVar="$OV_PRO" Step="10" Min="0" Max="100"/>
```

5.2.3 Arrange Controls in columns

All controls can be arranged in up to three columns each line. By default the number of columns is 1. In order to specify more columns the attributes ***Columns*** and ***ColSpan*** can be used. ***Columns*** is used within the attribute ***Group*** and is therefore valid for the whole tab. The attribute ***ColSpan*** is used for single controls in order to stretch a control across the given number of columns.

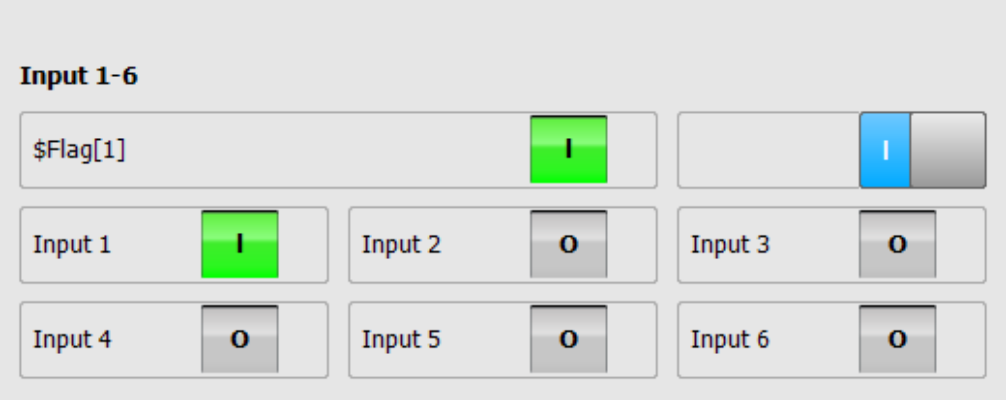
Maximum value for both attributes is 3.

Example

```
<Group Text="3Columns" Columns="3">
      <Control Type="Headline" Text="Input 1-6" ColSpan="3"/>
      <Control Type="Led" Text="Flag1" KrlVar="$FLAG[1]" ColSpan="2"/>
      <Control Type="Switch" Text="" KrlVar="$Flag[1]"/>
      <Control Type="Led" Text="Input 1" KrlVar="$FLAG[1]"/>
      <Control Type="Led" Text="Input 2" KrlVar="$FLAG[2]"/>
      <Control Type="Led" Text="Input 3" KrlVar="$FLAG[3]"/>
      <Control Type="Led" Text="Input 4" KrlVar="$FLAG[4]"/>
      <Control Type="Led" Text="Input 5" KrlVar="$FLAG[5]"/>
      <Control Type="Led" Text="Input 6" KrlVar="$FLAG[6]"/>
</Group>
```

➔ All controls in the tab will be arranged in three columns except the ones with specified attribute column span

- The 1’st control “Headline” is stretched across 3 columns
- The 2’nd control “Led” is stretched across 2 columns



5.3 Multilingualism

When selecting another HMI language, all text can be translated automatically. For this purpose, so-called "Keys" must be given within the argument text = "....". In order for these "Keys" being translated, the "Keys" and the corresponding translation texts must be entered in the KXR file "myFirstHMI.kxr".

Steps

- Edit the language database myfirstHMI.kxr file and replace all entries "Demo" to "myFirstHMI"
- Optionally add or change entries and save the file. Entries can be copied from DemoMyHMI.kxr
- Restart the robot

Example 1, specify a translation file in the XML file for a specific control

```
<Control Type="Led" Text="LED1" KrlVar="$FLAG[1]"
```

→ Entry in myFirstHMI.kxr

```
    <uiText key="LED1">
      <text xml:lang="de-DEV">Motor läuft</text>
      <text xml:lang="en-DEV">motor is running</text>
    </uiText>
```

xml:lang="**en-DEV“ → translation for the language english**

➔ Shown HMI when language English is set

Available languages

Elements	Language	Elements	Language
cs	Czech	pl	Polish
da	Danish	pt	Portuguese
de	German	ro	Romanian
en	English	sk	Slovak
es	Spanish	sl	Slovenian
el	Greek	sv	Swedish
fi	Finnish	tr	Turkish
fr	French	ru	Russian
it	Italian	ko	Korean
hu	Hungarian	zh	Chinese

nl	Dutch	ja	Japanese

If a key is found in the database,but no entry for the current HMI language is given, the text of the English language is shown (if given).

If no key is found in the database, the entry in the argument is displayed.

The kxr-file must have the encoding “UTF-8”. We recommend to use Notepad++ as editor.

In order to take changes in the kxr-Databese into affect, the robot or the SmartHMI has to be restarted