OrangeApps

myHMI

Benutzerdefinierte HMI per XML

Version 1.2

Schnelleinstieg

Stand: 27.01.2021, Version 1.2

OrangeApps GmbH
Arnikaweg 1
87471 Durach
Deutschland
www.orangeapps.de

Gültigkeit der Dokumentation

Dokument Version	Software Version		Verfasser	Datum
	Von	Bis		
1.7	1.2		Mayer	27.01.2021

Historie der Dokumentenversionen

Version	Datum	Autor	Änderungsgrund / Bemerkung
1.0	14.10.2016	Christian Mayer	Ersterstellung
1.1	20.02.2018	Christian Mayer	Implementation des Steuerelements „Picture“
1.2	27.1.2021	Christian Mayer	Steuerelement Switch IO, Installation als KOP über WoV

Inhalt

1 Einleitung

1.1 Zielgruppe

Diese Dokumentation richtet sich an Anwender mit folgenden Kenntnissen:

- Kenntnisse über die Software-Struktur des KUKA Robotersystems

1.2 Darstellung von Hinweisen

Diese Hinweise enthalten nützliche Tipps oder besondere Informationen für das aktuelle Thema.

1.3 Verwendete Begriffe

Begriff	Beschreibung
HMI	Die Human-Machine Interface (HMI) ist eine Schnittstelle, über die ein Mensch mit einer Maschine kommuniziert.
KSS	KUKA Systemsoftware
smartPad	Roboter Bediengerät
SmartHMI	Bedienoberfläche der KRC4 Robotersteuerung
KOP	KUKA Option Package
WoV	KUKA Work Visual

2 Übersicht

2.1 Produktbeschreibung

Mit myHMI können anwenderspezifische HMI‘s auf dem SmartPad angezeigt werden. Diese diesen zur Anzeige und Manipulation aller auf dem Robotersystem bekannten KRL-Variablen werden. Im Vordergrund steht die Anzeige von BOOL, INT, REAL, ENUM und CHAR Variablen unter Verwendung von grafischen Elementen wie Textfelder, Schalter, LEDs, Dropdown-Steuerelemente und Bildsteuerelementen.

Die Erstellung einer HMI erfolgt über eine XML-Datei in der mittels eines normalen Texteditors die spezifischen Einträge erstellt werden. Ein Plugin interpretiert die Einträge in der XML-Datei und stellt die Elemente in tabellarischer Form auf einer HMI dar. Der Anwender benötigt somit keinerlei Kenntnisse in der Programmierung von Plugins und HMI‘s.

Es können beliebig viele XML-Dateien und somit beliebig viele HMI’s erzeugt werden. Der Aufruf jeder einzelnen HMI erfolgt über einen Menüeintrag im Hauptmenü des Roboters. Dabei kann zwischen einer halb- oder vollseitigen Anzeige gewählt werden. Die HMI bettet sich nahtlos in das Robotersystem ein. Alle standardmäßigen Menü- und Bedienelemente bleiben vollständig bedienbar. Zur einfachen Erstellung der Menüeinträge steht ein komfortabler Menüassistent zur Verfügung.

Um eine thematische Trennung innerhalb einer HMI zu ermöglichen, können die Inhalte auf maximal 5 Registerkarten verteilt werden. Jede Registerkarte kann 32 Elemente darstellen.

Jedes Element kann dynamisch nach angemeldeter Benutzerebene dargestellt (sichtbar) bzw. editierbar (schreibgeschützt) sein.

Die gesamte HMI unterstützt Mehrsprachigkeit, sodass eine dynamische Sprachumschaltung problemlos möglich ist.

Durch den Bediener geänderte Werte, werden im KUKA Logbuch aufgezeichnet. (Diagnose/Logbuch).

Zur Erstellung der Menüaufrufe der einzelnen HMI’s steht ein Menüassistent zur Verfügung. Somit sind keinerlei Kenntnisse zur Menüerstellung notwendig.

2.2 Merkmale

- HMI zur Anzeige und Manipulation von KRL Variablen
- Angezeigte Inhalte werden per XML definiert
- Inhalte werden thematisch mit Registerkarten dargestellt
- Die Steuerelemente werden untereinander in tabellarischer Form mit bis zu 3 Spalten dargestellt
- Steuerelemente: Schalter, Taster, Textfeld, Nummernfeld, LED, Dropdown-Liste, Schieberegler, Fortschrittsanzeige, Überschrift, Beschriftungsfeld und Bild
- Neues Steuerelement:
- Die Steuerelemente können durch Angabe optionaler Abhängigkeiten aktiviert oder deaktiviert werden (Benutzergruppe, Submit u. Programm-Status, Antriebe, Zustimmschalter, Betriebsart usw.)
- Dynamische Bildanzeige
- Bilddateien können sowohl lokal als auch auf einem Netzlaufwerk gespeichert sein

- Inhalte können unter Verwendung von KXR-Dateien mehrsprachig dargestellt werden
- Benutzereingaben werden im Logbuch gespeichert
- Die Anzahl der erstellten HMI ist theoretisch unbegrenzt
- Bis zu 5 Registerkarten können je HMI definiert werden
- Bis zu 32 Elemente können je Registerkarte definiert werden
- Jede HMI wird aus dem Hauptmenü geöffnet
- die Menüeinträge können mit einem Menüassistenten erstellt werden
- einfache Installation der Software über die Standard KUKA Installationsroutine
- Neue Funktion ab V1.2: Öffnen und Schließen der HMI‘s direkt über KRL, automatischer Start beim Start des Controllers und abhängig von der Änderung des Benutzerlevels und der Betriebsart
- Installation über WorkVisual möglich

2.3 Lieferumfang

Die Lieferung erfolgt als Technologiepaket zur direkten Installation am Roboter als Zusatzsoftware. Darin sind alle zur Installation und Betrieb notwendigen Komponenten enthalten:

- Plugin myHMI
- Anwender-Dokumentation zur Installation und Betrieb der Software
- Menü Assistent

Um dem Anwender den Einstieg zu erleichtern, ist im Setup-Paket eine Beispiel-HMI mit folgenden Dateien enthalten:

- DemoMyHMI.xml → enthält Beispiele zur Verwendung von Registerkarten und Steuerelementen
- DemoMyHMI.kxr → Beispiel zum Erstellen einer Sprachdatenbank für Mehrsprachigkeit

2.4 Einsatzgebiet / -umgebung

Die Software ist lauffähig auf allen KUKA Roboter mit KSS8.3.23 oder höher ohne CPC-Schutz.

Auskunft über den Einsatz der Software auf älteren Maschinen sind unter info@orangeapps.de erhältlich.

2.5 CPC-Zertifikat

Soll die Software auf Robotern mit KUKA.CPC eingesetzt werden, wird vor der Installation ein CPC-Zertifikat benötigt. Bitte kontaktieren Sie uns in diesem Fall.

3 Installation

Ab der Version 1.2.5 wird die Software als KUKA Optionspaket (KOP) ausgeliefert.

Die Installation kann über Work Visual (WoV) oder über die Option *Zusatzsoftware* erfolgen. Die Option *Zusatzsoftware* befindet sich im Hauptmenü unter *Inbetriebnahme.*

Ältere Version als 1.2.5 müssen vor der Installation des KOP deinstalliert werden.

Mindestanforderungen Software

- KUKA System Software 8.3.23

Schritte Installation über Zusatzsoftware

- Software auf den Roboter kopieren
- Im Hauptmenü Inbetriebnahme→Zusatzsoftware wählen
- *Neue Software* wählen und installieren

Schritte Installation über WoV

- Optionspaket in WoV als Katalogelement installieren
- Projekt von Roboter ziehen
- Option einfügen
- Am Roboter als Experte anmelden und Projekt übertragen

Die Software inklusive einer Beispiel HMI wird installiert

Folgende Dateien werden für die Beispiel-HMI installiert:

Ordner	Dateien	Funktion
C:\KRC\SmartHMI	SmartHMI.exe.DemoMyHMI.config	Menüeintrag
C:\KRC\DATA	DemoMyHMI.kxr	Sprachdatenbank für Demo HMI
C:\KRC\USER\myHMI	DemoMyHMI.xml	HMI
C:\KRC\USER\myHMI\PicsDemo	Verschiedene Bilddateien	

Die Beispiel HMI ist voll lauffähig und kann somit als Grundlage für weitere HMI's genutzt werden.

4 Lizenzierung

myHMI ist für die produktive Nutzung lizenzierungspflichtig. Die Lizenzierung erfolgt über eine Lizenzdatei. Zu Testzwecken sind kostenlose Test-Lizenzen unter www.orangeapps.de erhältlich.

Hinweis

- Für jeden Roboter ist eine Lizenz notwendig.
- Testlizenzen sind kostenlos und zeitlich begrenzt.
- **Datumsmanipulationen** am System werden erkannt, myHMI deaktiviert die Lizenz automatisch

Testlizenzen können direkt auf www.orangeapps.de bezogen werden. Laufzeitlizenzen erhalten Sie nach Eingang der Lizenzgebühr.

4.1 Roboterlizenz

Um eine gültige Lizenz zu erhalten, benötigen Sie die Seriennummer des Roboters. Diese finden Sie auf dem Typenschild des Roboters oder in der Steuerungssoftware im Menü ***Hilfe→Info→Roboter→Seriennummer***.

4.2 Lizenz für KUKA OfficePC/OfficeLite

Nach der Installation und dem Start der Software wird eine Produkt-ID angezeigt. Diese benötigen Sie um eine gültige Lizenz zu erhalten.

4.1 Lizenz installieren

Methode 1

- Stecken Sie einen USB-Stick mit darauf gespeicherter Lizenz an einem USB Port der Steuerung ein.
- Beim Hochlauf der Steuerung oder dem Öffnen einer HMI wird bei Vorhandensein einer gültigen Lizenz auf dem USB-Stick diese automatisch in den Lizenzordner kopiert und aktiviert. **Hinweis:** Eine Laufzeitlizenz im Lizenzordner wird dabei nicht durch eine Testlizenz überschrieben
- Entfernen Sie den USB-Stick

Methode 2

- Kopieren Sie die erhaltene Lizenz in den Ordner C:\KRC\TP\OrangeApps.myHMI\lic.

5 Eigene HMI „myFirstHMI“ erstellen

Um eine eigene HMI zu erstellen kann die Demo-HMI kopiert und an die eigenen Bedürfnisse angepasst werden.

Vorgehensweise zum Erstellen einer eigenen HMI

- Die Datei „DemoMyHMI.xml“ im Ordner C:\KRC\USER\myHMI nach “myFirstHMI.xml” kopieren
- Den Menüassistenten öffnen und einen Menüeintrag erzeugen
- Die Datei myFirstHMI.xml auf einem Laptop oder direkt am SmartPad (Tastatur anschließen) modifizieren (entsprechend den eigenen Wünschen)
- Die neue xml-Datei im Ordner C:\KRC\USER\myHMI speichern
- Über den Menüeintrag die neue HMI öffnen → fertig

Zur Erstellung und zur Bearbeitung von XML-Dateien, wird „Notepad++“ empfohlen. Zum einen werden die XML-Inhalte farblich gut lesbar dargestellt und zum anderen werden grundsätzliche XML-Fehler von dem in Notepad++ enthaltenen XML-Parser bereits beim Speichern der Datei erkannt.

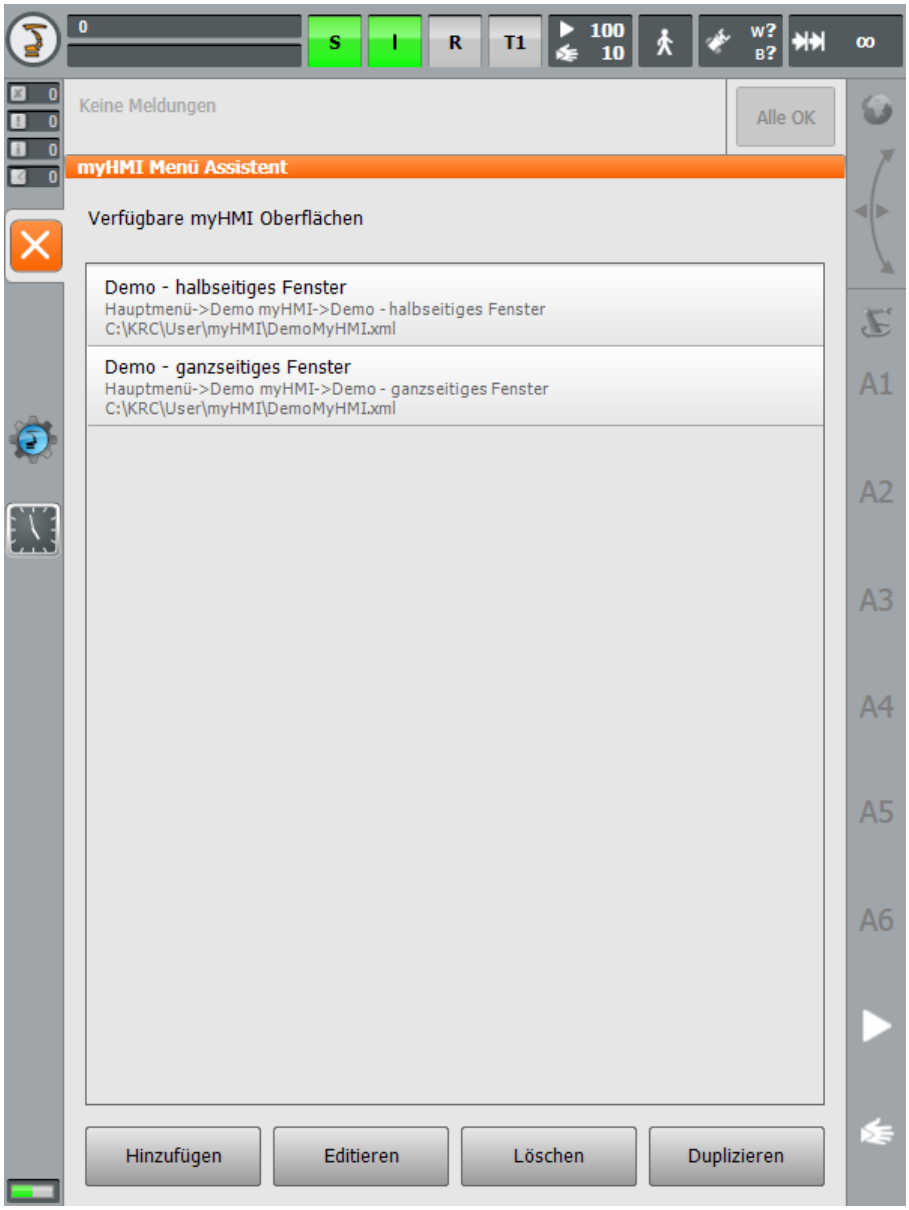
Um eine aktualisierte XML-Datei in myHMI darzustellen, ist es ausreichend das Anzeigefenster der HMI zu schließen und wieder zu öffnen. Änderungen in einer KXR-Sprachdatei erfordern einen Neustart des SmartHMI, oder alternativ einen Kaltstart der Roboter-Steuerung.

6 Menüeintrag mit dem Menü-Assistenten erzeugen

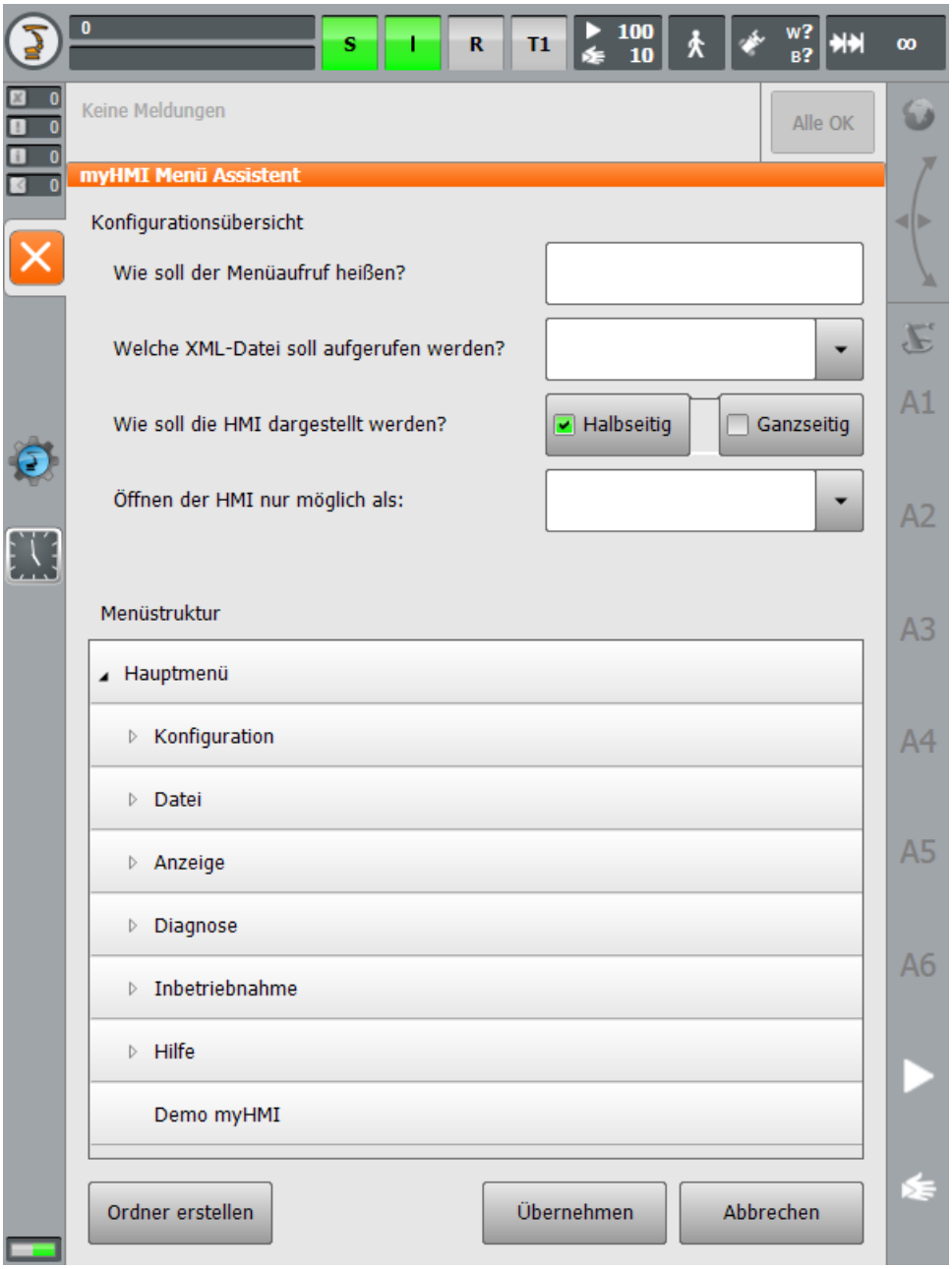
Der Menü-Assistent ermöglicht ein einfaches Erzeugen von Menüeinträgen

Vorgehensweise

- Am Roboter als ***Administrator*** einloggen
- Den Menü-Assistenten im Menü ***Konfiguration*** → ***myHMI Menü Assistent*** öffnen
- Einen neuen Eintrag mit der Schaltfläche ***Hinzufügen*** erzeugen
- Alle Eingabefelder bearbeiten und ***Übernehmen*** klicken
- Den Menü-Assistenten schliessen



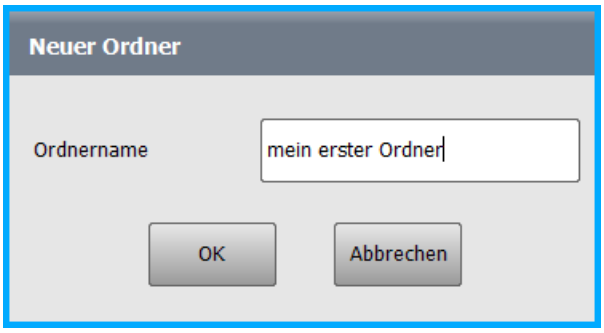
➔ ***Hinzufügen*** klicken



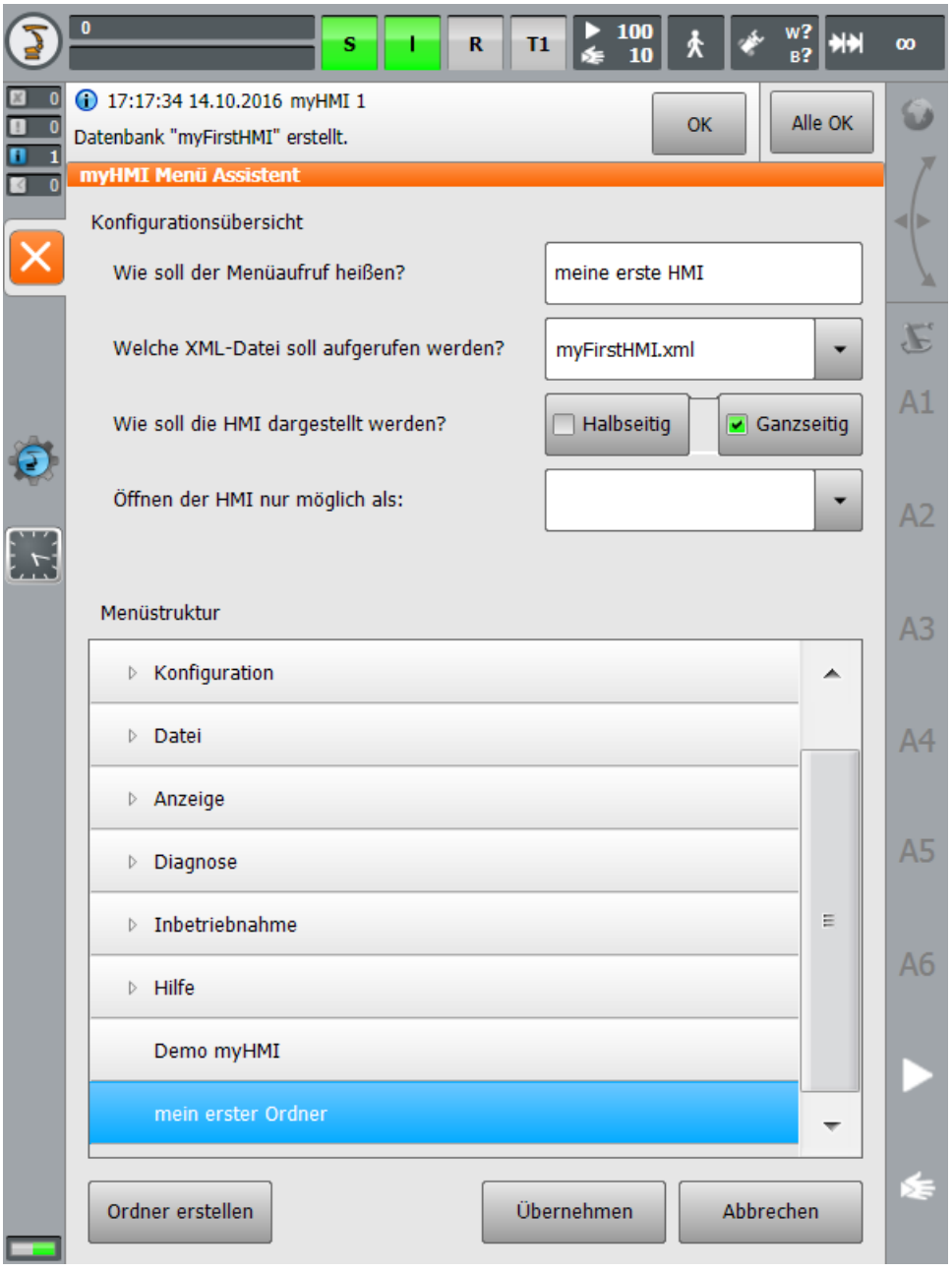
Vorgehensweise

- Den Namen für den Menüaufruf festlegen
- Die xml-Datei “myFirstHMI.xml” aus der Dropdown-Liste auswählen
- Die Größe des Anzeigefensters der HMI wählen (halb- oder ganzseitige Anzeige)
- Den Benutzerlevel zum Öffnen der HMI auswählen
- Den Ordner auswählen in dem der Menüeintrag im Hauptmenü erscheinen soll. Mit der Schaltfläche ***Ordner erstellen*** kann ein neuer Ordner dem Hauptmenü hinzugefügt werden
- Mit ***Übernehmen*** den Menü Eintrag speichern

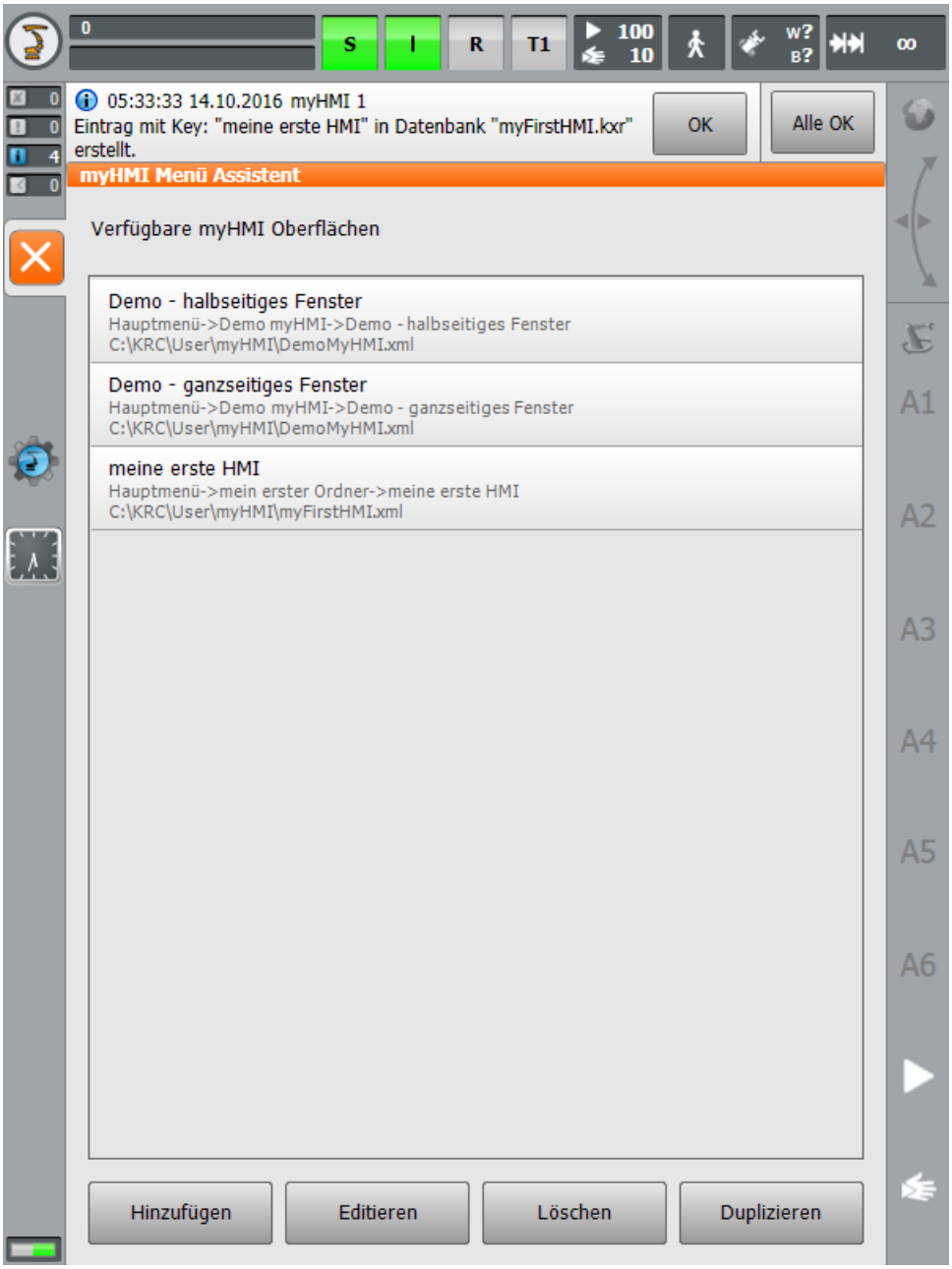
Da die Software automatisch eine Sprachdatenbank mit dem Namen der xml-Datei anlegt, sollte **bevor** ein neuer Ordner angelegt wird, die betreffende xml-Datei bereits ausgewählt sein.



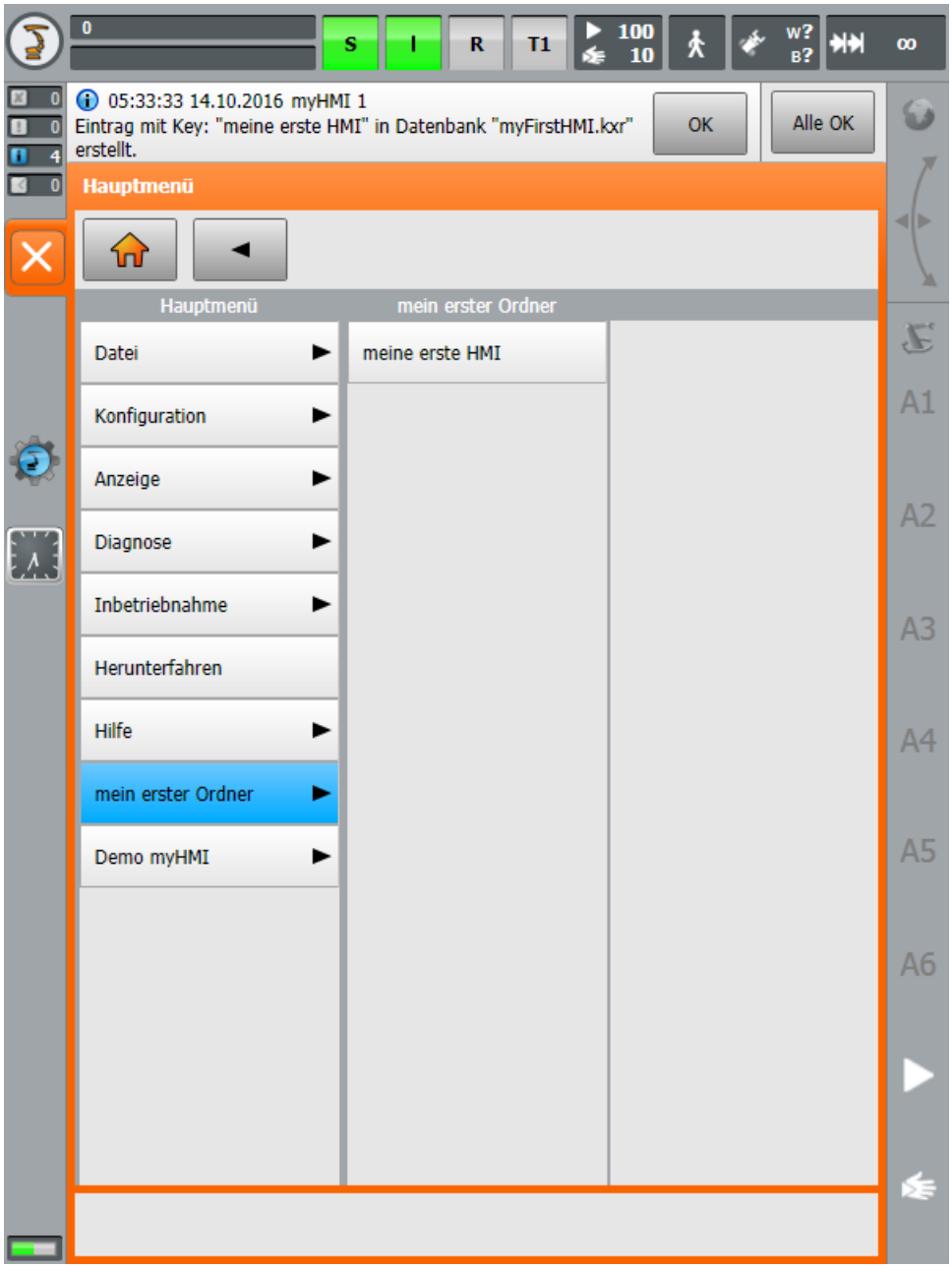
Nach dem Drücken von ***Ok*** erstellt die Software den Menüeintrag und automatisch eine Sprachdatenbank (falls noch nicht vorhanden). Diese Datenbank kann für Mehrsprachigkeit genutzt werden.



➔ ***Übernehmen*** klicken



Menü Eintrag im Hauptmenü:



7 Die HMI modifizieren

Die Modifikation der xml-Datei kann mit jedem herkömmlichen Texteditor erfolgen. Im OrangeEdit ist ein Designer zur Modifikation der xml-Datei enthalten.

Jede HMI kann bis zu 5 Registerkarten mit jeweils 32 Steuerelementen besitzen. Die Reihenfolge der Anzeige der Steuerelemente auf der HMI wird durch die Reihenfolge der Einträge in der xml-Datei bestimmt.

Um eine geänderte xml-Datei dem System bekannt zu machen reicht es die HMI neu zu öffnen oder bei bereits geöffneter HMI die Registerkarte zu wechseln.

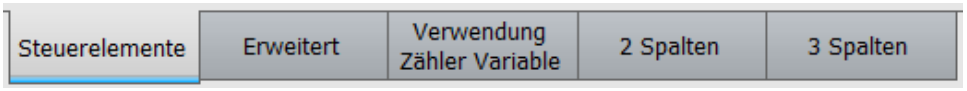
7.1 Registerkarten erstellen

Die Registerkarten werden durch das Element <Group Text="……"> festgelegt.

Beispiel XML

```
<Configuration Text="MyFirst">
<Group Text="Steuerelemente">
  . . .
</Group>
<Group Text="Erweitert">
  . . .
</Group>
  . . .
</Configuration>
```

Beispiel Darstellung



Wird innerhalb einer XML-Datei nur eine Registerkarte definiert, wird in der HMI **keine** Registerkarte dargestellt. Stattdessen kann der Platz für ein weiteres Steuerelement genutzt werden.

7.2 Steuerelemente erstellen

Die Steuerelemente werden durch das Element <Control Type"="…" Text="…" KrlVar="…"> festgelegt.

Um Steuerelemente innerhalb einer Registerkarte zu definieren, muss die Angabe innerhalb des Knotens

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

der jeweiligen Registerkarte erfolgen.

Type="…." → legt den Typ fest

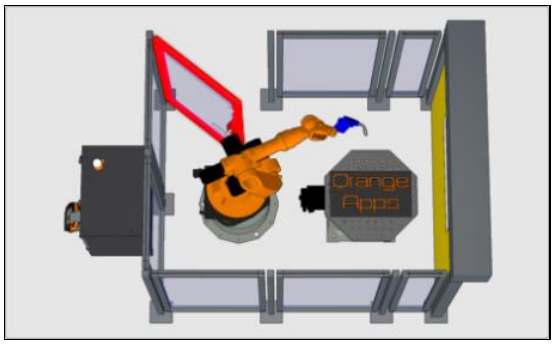
Text="…." → legt den Beschreibungstext fest

KrlVar="…." → legt die verbundene KRL-Variable fest

7.2.1 Übersicht verfügbare Steuerelemente

Folgende Steuerelemente stehen zur Verfügung:

- Picture



- Number

- LED

- Switch

- **SwitchIO (neu in V1.2)**

- Checkbox

- Button

- DropDown

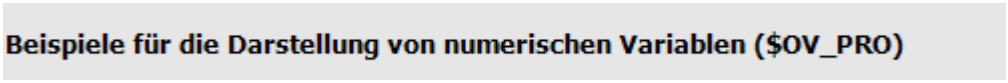
- Text

- Label

- Progressbar

- Slider

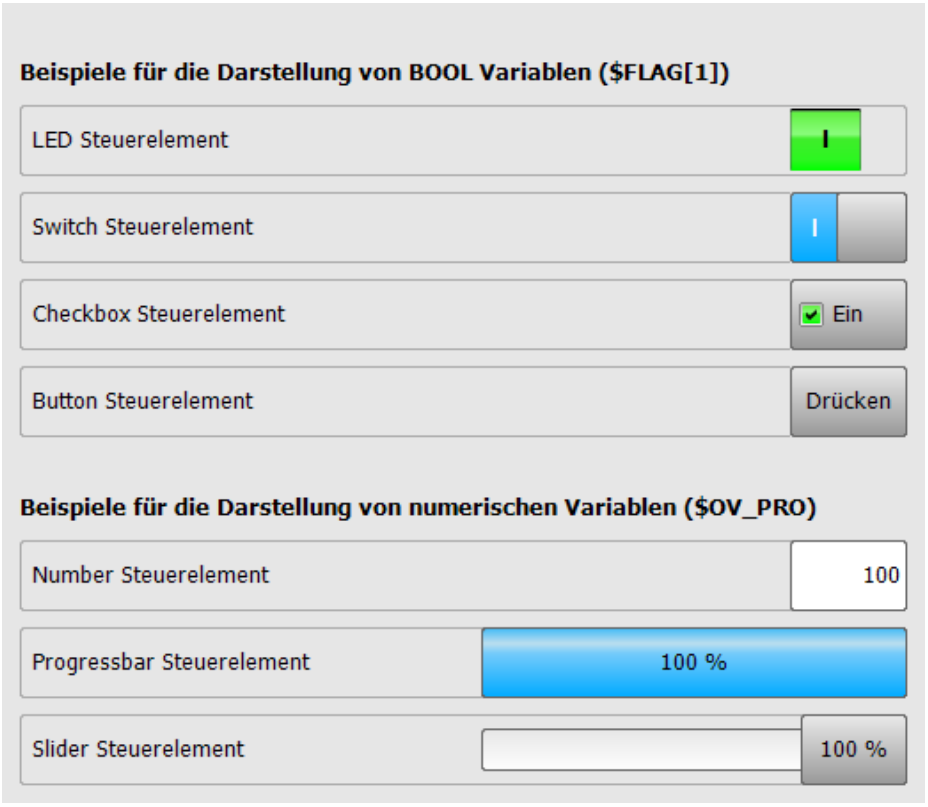
- Headline



Alle Steuerelemente können optional durch eine Vielzahl von Argumenten in Funktion und Aussehen beeinflusst werden.

Beispiele

```
<Control Type="Headline" Text="Beispiele für die Darstellung von BOOL
Variablen ($Flag[1])"/>
<Control Type="Led" Text="Led Steuerelement" KrlVar="$Flag[1]"/>
<Control Type="Switch" Text="Switch Steuerelement" KrlVar="$Flag[1]"/>
<Control Type="Checkbox" Text="Checkbox Steuerelement" KrlVar="$Flag[1]"/>
<Control Type="Button" Text="Button Steuerelement" TextButton="Press"
KrlVar="$Flag[1]"/>
<Control Type="Headline" Text=" Beispiele für die Darstellung von
numerischen Variablen ($OV_PRO)"/>
<Control Type="Number" Text="Number Steuerelement" KrlVar="$OV_PRO"/>
<Control Type="Progressbar" Text="Progressbar Steuerelement"
KrlVar="$OV_PRO" Format="0 \%" Min="0" Max="100"/>
<Control Type="Slider" Text="Slider Steuerelement" KrlVar="$OV_PRO"
Format="0 \%" Min="0" Max="100"/>
```



7.2.2 Optionale Argumente der Steuer- und Anzeige-Elemente

Für jedes Steuerelement stehen weitere Argumente zur Verfügung. Über diese kann das Aussehen und das Verhalten jedes Steuerelements beeinflusst werden.

Beschreibung aller möglichen Argumente

Argument	Beschreibung	Optional	Standardwert
Alignment (Label)	Richtet den Beschriftungstext aus (oben, mittig, unten, links, mittig, rechts)	Ja	`bottomleft`
Alignment (Picture)	Richtet das Bild aus (links, mittig, rechts)	Ja	`right`
AreYouSure	True = Sicherheitsabfrage bei Wert-Änderung einblenden (Dialog)	Ja	False
Border	Steuert ob ein Bild mit oder ohne Rahmen dargestellt wird (mit=TRUE)	Ja	TRUE
Color0	Farbe der LED des Steuerelements bei Zustand False Mögliche Werte: Grey, Green, Red, Yellow	Ja	`Gray`
Color1	Farbe der LED des Steuerelements „Led“ bei Zustand TRUE Mögliche Werte: Grey, Green, Red, Yellow	Ja	`Green`
ColSpan	Anzahl Spalten über die sich das Element erstreckt	Ja	1
Description	Wird auf die Beschriftung geklickt, so wird dieser Text als Beschreibung angezeigt	Ja	
Format	Formatierung von INT und REAL Werten	Ja	
KrlVar	Verknüpfte KRL Variable	Nein	
Max	Größter zulässiger Wert	Ja	
Min	Kleinster zulässiger Wert	Ja	
ModeOP	Betriebsart ab der das Element editierbar ist	Ja	31
Module	Modul / KXR-Datei für multilinguale Inhalte	Ja	Wert aus *Gruppe*
NeedDrivesReady	Editierbarkeit des Elements ist vom Zustand der Antriebe abhängig	Ja	False
NeedSafetySwitch	Editierbarkeit des Elements ist vom Zustand des Zustimmschalter abhängig	Ja	False
Negate	Invertiert eine boolesche Variable	Ja	`False`

Path	Legt den Pfad zu einer Bilddatei fest	Nein	
ProState0	Editierbarkeit des Elements ist vom Zustand des Submit-Interpreters abhängig	Ja	63
ProState1	Editierbarkeit des Elements ist vom Zustand des Programm-Interpreters abhängig	ja	63
Step	Schrittweite für Auf-/Abtaste	Ja	
Text	Beschriftungstext oder Key für das Element	Nein	
Text0	Beschriftung der Schaltfläche des Steuerelements "Checkbox" bei Zustand False	Ja	`Aus`
Text1	Beschriftung der Schaltfläche des Steuerelements "Checkbox" bei Zustand True	Ja	`Ein`
TextButton	Beschriftung der Schaltfläche des Steuerelements "Button"	Ja	
UserLevelEdit	Benutzergruppe ab der das Element editierbar ist	Ja	0
UserLevelVisible	Benutzergruppe ab der das Element sichtbar ist	Ja	0
Value	Wert eines Eintrags im Steuerelement "DropDown". Wird bei Auswahl des Eintrags an die Variable des Arguments "KrlVar" übergeben.	Nein	
Width	Legt die Breite eines Bildes fest (Pixel). Die Höhe wird entsprechend proportional skaliert.	Ja	Breite des Bildes

Argumente/Elemente-Matrix

Argument	<Configuration>	<Group>	Number	Led	Switch	SwitchIO	Checkbox	Button	Slider	Progressbar	Dropdown	Text	Label	Headline	Picture
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	n/a	O	O
AreYouSure	n/a	n/a	O	n/a	O	O	O	n/a	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	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	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	n/a
ColSpan	n/a	n/a	O	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	O	n/a	n/a
Format	n/a	n/a	O	n/a	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	X	n/a	O
Max	n/a	n/a	O	n/a	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	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	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	O
NeedDrivesReady	n/a	n/a	O	n/a	O	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	O	n/a	O	O	n/a	n/a	n/a
Negate	n/a	n/a	n/a	O	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	n/a	X
ProState0	n/a	n/a	O	n/a	O	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	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	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	X	O
Text0/1	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	n/a
TextButton	n/a	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	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	O

Value	n/a	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	n/a	O

X: Angabe zwingend erforderlich | O: Angabe optional | n/a: nicht verfügbar

Beispiel 1

```
<Control Type="Led" Text="Led in den Farben Rot/Grün" Description="Mit dem
optionalen Argument &quot;Color0&quot; und &quot;Color1&quot;, kann für den
jeweiligen Zustand eine der folgenden Farben vorgegeben werden: Gray,
Green, Yellow, Red" KrlVar="$FLAG[1]" Color0="Red" Color1="Green"/>
```

Beispiel 2

```
<Control Type="Number" Text="Optionale Auf-/Ab-Tasten und Min/Max"
KrlVar="$OV_PRO" Step="10" Min="0" Max="100"/>
```

7.2.3 Anordnung der Steuerelemente in Spalten

Alle Steuerelemente lassen sich in bis zu 3 Spalten pro Zeile anordnen. Standardmäßig beträgt die Spaltenzahl 1. Um mehr Spalten zu definieren werden die Attribute ***Columns*** und ***ColSpan*** verwendet. ***Columns*** kennzeichnet die Anzahl der Spalten innerhalb einer Registerkarte, ***ColSpan*** wird verwendet um ein Steuerelement über mehrere Spalten zu spannen.

Der Maximalwert für beide Attribute ist3.

Beispiel

```
<Group Text="3 Spalten" Columns="3">
      <Control Type="Headline" Text="Eingänge 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="Eingang 1" KrlVar="$FLAG[1]"/>
      <Control Type="Led" Text="Eingang 2" KrlVar="$FLAG[2]"/>
      <Control Type="Led" Text="Eingang 3" KrlVar="$FLAG[3]"/>
      <Control Type="Led" Text="Eingang 4" KrlVar="$FLAG[4]"/>
      <Control Type="Led" Text="Eingang 5" KrlVar="$FLAG[5]"/>
      <Control Type="Led" Text="Eingang 6" KrlVar="$FLAG[6]"/>
</Group>
```

➔ Ausgenommen der Steuerelemente für die das Attribut ColSpan gesetzt wurde, werden alle Elemente in drei Spalten angeordnet

- Das erste Steuerelement “Headline” wird über drei Spalten gespannt
- Das zweite Steuerelement “Led” wird über zwei Spalten gespannt

7.3 Mehrsprachigkeit

Beim Einstellen einer anderen Sprache können Beschriftungstexte automatisch übersetzt werden. Dazu werden beim Argument Text="…." sogenannte "Keys" angegeben. Damit diese "Keys" dann übersetzt werden, müssen die "Keys" und die dazugehörigen Übersetzungstexte in der KXR-Datei "myFirstHMI.kxr" eingetragen werden.

Vorgehensweise

- Die Sprachdatenbank „myFirstHMI.kxr“ öffnen
- Wie gewünscht Einträge ändern und die Datei speichern. Einträge könne aus DemomyHMI.kxr kopiert werden
- Den Roboter neu starten

Beispiel: Angabe eines Keys für das Steuerelement LED in myFirstHMI.xml

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

→ Eintrag 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="**de-DEV“ → Übersetzung für Deutsch**

➔ Angezeigte HMI, wenn die HMI Sprache Deutsch eingestellt ist

Verfügbare Sprachen

Elemente	Sprache	Elemente	Sprache
cs	Tschechisch	pl	Polnisch
da	Dänisch	pt	Portugiesisch
de	Deutsch	ro	Rumänisch
en	Englisch	sk	Slowakisch
es	Spanisch	sl	Slowenisch
el	Griechisch	sv	Schwedisch
fi	Finnisch	tr	Türkisch
fr	Französisch	ru	Russisch
it	Italienisch	ko	Koreanisch
hu	Ungarisch	zh	chinesisch

nl	Holländisch	ja	japanisch

Wird ein Key in der Datenbank gefunden, aber kein Eintrag für die aktuelle HMI-Sprache, wird der Text der Sprache Englisch dargestellt (falls vorhanden).

Wird kein Key in der Datenbank gefunden, wird der Eintrag in den Argumenten dargestellt.

Die KXR-Datei muss die Kodierung „UTF-8“ haben. Wir empfehlen die Verwendung des Editors „Notepad++“

Nach dem Ändern der kxr-Datei muss der Roboter neu gestartet werden damit die Änderungen wirksam werden.

8 HMI’s automatisiert öffnen und schließen

HMI’s können aus KRL Modulen gezielt geöffnet und geschlossen werden. Zudem können HMI’s nach dem Hochlauf der Steuerung und beim Wechsel der Betriebsart oder des Benutzers angezeigt werden.

8.1 KRL Funktionen zum Öffnen und Schließen von HMI’s

8.1.1 Funktion MyHmiOpen

Öffnet eine HMI aus KRL heraus.

Funktionsaufruf

MyHMIOpen(char „**Name der HMI“**, enum **ViewMode**, int **Tabnummer**)

Parameter

- View [char], Pfad zur MyHMI.xml Datei z.B. „MyHmiDemo.xml“
- ViewMode [enum], optional, „#Half“ für halbseitiges Fenster, „#Full“ für ganze Seite (Default=#Full)
- Tab [int], optional, Reiter der HMI, welcher geöffnet werden soll (Default=1)

Beispiel

```
MyHmiOpen("myHmiDemo.xml", #Full, 2)
```

Öffnet die HMI ganzseitig auf Reiter 2

8.1.2 Funktion MyHmiClose

Schließt eine HMI aus KRL heraus.

Funktionsaufruf

myHMIClose(char „**Name der HMI“**)

Parameter

- View [char], optional, Pfad zur MyHMI.xml Datei z.B. „MyHmiDemo.xml“ (Default= *)

Beispiel

```
MyHmiClose("myHmiDemo.xml")
```

Schließt die HMI „myHmiDemo.xml“

Beispiel

```
MyHmiClose()
```

Schließt alle geöffneten HMI‘s

8.1.3 Funktion MyHmiIsOpen

Gibt zurück, ob eine HMI geöffnet ist

Funktionsaufruf

MyHmiIsOpen(char „**Name der HMI“**)

Parameter

- View [char], Pfad zur MyHMI.xml Datei z.B. „MyHmiDemo.xml“ (optional: Default *)

Rückgabewert

TRUE oder FALSE, TRUE=angegeben HMI ist geöffnet

Beispiel

```
IF MyHmiIsOpen("myHmiDemo.xml") THEN
  ;HMI ist offen
ELSE
  ;HMI ist nicht offen
ENDIF
```

Gibt zurück ob die HMI „myHmiDemo.xml“ geöffnet ist

8.2 KRL Variablen für bedingtes Öffnen von HMI's

Mit den folgenden Variablen können HMI's zu bestimmten Bedingungen geöffnet werden. Die Variablen befinden sich in der Datei „R1\TP\myHMI_User.dat“.

8.2.1 HMI nach Hochlauf öffnen (Autostart)

Durch Beschreiben der folgende Variable, kann eine HMI nach dem Hochlauf der Steuerung geöffnet werden (Autostart)

Variable

OpenOnStartup(char **Name der HMI,** enum **ViewMode**,int **Tabnummer**)

Parameter

- View [char], Pfad zur MyHMI.xml Datei z.B. „MyHmiDemo.xml“
- ViewMode [enum], optional, „#Half“ für halbseitiges Fenster, „#Full“ für ganze Seite (Default=#Full)
- Tab [int], optional, Reiter der HMI, welcher geöffnet werden soll (Default=1)

Beispiel

Autostart deaktiviert, da keine HMI (XML-Datei) angegeben wurde

```
DECL myHmi_S OpenOnStartup={View[] " ",ViewMode #Full,Tab 0}
```

Beispiel

Autostart aktiv, nach Hochlauf wird die HMI „myHmiDemo.xml“ ganzseitig mit Reiter 3 geöffnet

```
DECL myHmi_S OpenOnStartup={View[] "myHmiDemo.xml",ViewMode #Full,Tab 3}
```

8.2.2 HMI bei Betriebsarten-Wechsel und/oder Benutzer-Wechsel öffnen

Durch Beschreiben der folgenden Variable kann eine HMI nach Benutzer-Wechsel oder Betriebsarten-Wechsel geöffnet werden.

Die Angabe der Betriebsarten und der Benutzerlevel ist optional und erfolgt als Charakter Variablen. Die Werte werden dabei kommagetrennt angegeben.

Es können bis zu 10 Bedingungen konfiguriert werden, die erste zutreffende Bedingung wird erkannt und die entsprechende HMI geöffnet.

Variable

OpenOnCondition[Nummer des Array] = (char **Name der HMI,** enum **ViewMode**, int **Tabnummer,** char **ModeOps**,char **UserModes**)

Parameter

- View [char], Pfad zur MyHMI.xml Datei z.B. „MyHmiDemo.xml“
- ViewMode [enum], optional, „#Half“ für halbseitiges Fenster, „#Full“ für ganze Seite (Default=#Full)
- Tab [int], optional, Reiter der HMI, welcher geöffnet werden soll (Default=1)
- ModeOps [char], optional, Betriebsarten T1, T2, AUT, EXT (Default= *)
- UserModes [char], optional, Benutzerlevel 5,10,15,20,29,30 (Default= *)

Beispiel

Bei jedem Benutzer- oder Betriebsarten-Wechsel wird „myHmiDemo“ geöffnet

```
OpenOnCondition[1]={View[] "myHmiDemo.xml",ViewMode #Full,Tab 0,ModeOps[] "
",UserModes[] " "}
```

Beispiel

Nur wenn Betriebsart „EXT“, oder „AUT“ und Benutzer-Level 5 oder 10 aktiv ist wird „myHmiDemo“ ganzseitig mit Reiter 2 geöffnet

```
OpenOnCondition[1]={View[] "myHmiDemo.xml",ViewMode #Full,Tab 2,ModeOps[]
"EXT,AUT",UserModes[] "5,10"}
```

Betriebsarten

- T1 – Test 1
- T2 – Test 2
- AUT – Automatik
- EXT – Extern

Benutzer-Level

- 5 – Standard
- 10 – Bediener
- 20 – Expert
- 27 – Sicherheits-Instandhalter
- 29 – Sicherheits-Inbetriebnehmer
- 30 – Administrator