

mapp

FRAMEWORK



Table of contents

mapp Rahmen	4
Allgemeine Informationen	4
Einführung	4
System Anforderungen	6
Voraussetzungen	6
Starten des Importers	6
Umgang mit Benutzerpartitionen	7
Überlegungen zur Visualisierung	7
Versionsinformation	8
Ausführung 1.00.0	8
mapp AlarmX	8
Merkmale	9
Aufgabenübersicht	9
Erforderliche Benachrichtigungen	10
Optionale Benachrichtigungen	11
Wecker hinzufügen oder löschen	12
Alarmzuordnung	12
Alarmer sperren	14
Abfragen hinzufügen	15
mapp Axis / Cockpit	15
Merkmale	16
Aufgabenübersicht	16
Erforderliche Benachrichtigungen	17
Optionale Benachrichtigungen	18
Zusätzliche Achse hinzufügen	18
Umbenennen AppAxis_1	19
mapp Backup	20
Merkmale	20
Aufgabenübersicht	20
Erforderliche Benachrichtigungen	21
Optionale Benachrichtigungen	21
mapp File	21
Merkmale	21
Aufgabenübersicht	22
Erforderliche Benachrichtigungen	22
Optionale Benachrichtigungen	22
FIFO	22
mapp Recipe	23
Merkmale	23
Design des Rezeptsystems	23
Aufgabenübersicht	24
Erforderliche Benachrichtigungen	25
Optionale Benachrichtigungen	25
Rezeptformat ändern	25
mapp UserX	26
Merkmale	26
Aufgabenübersicht	27

Erforderliche Benachrichtigungen	27
Optionale Benachrichtigungen	27

mapp Rahmen

Um mit dem mapp Framework zu beginnen, fahren Sie mit dem Abschnitt [Allgemeine Informationen](#) fort.



WICHTIG: Jeder Abschnitt der mapp Framework-Hilfe hat eine Seite mit dem Titel "Erforderliche Änderungen". Die Schritte auf dieser Seite müssen ausgeführt werden, um das Framework in einen funktionsfähigen Zustand zu bringen!

mapp Rahmen - Allgemeine Informationen

Dieser Abschnitt enthält Informationen, die für alle mapp Frameworks relevant sind.

WICHTIG: Jeder Abschnitt der mapp Framework-Hilfe hat eine Seite mit dem Titel "Erforderliche Änderungen". Die Schritte auf dieser Seite müssen ausgeführt werden, um das Framework in einen funktionsfähigen Zustand zu bringen!

Themen in diesem Abschnitt:

- [Introduction](#)
- [System Requirements](#)
- [Prerequisites](#)
- [Launching the Importer](#)
- [User Partition Handling](#)
- [Visualization Considerations](#)
- [Version Information](#)

mapp Rahmen - Einführung

Überblick

mapp Technology ist der Oberbegriff für die vorgefertigten, modularen Softwareprodukte bei B&R. Diese

Technologie ermöglicht es Ihnen, komplexe oder langwierige Funktionen (z. B. ein Rezeptsystem) mit nur wenigen mapp-Funktionsblöcken und Konfigurationseinstellungen zu implementieren, anstatt sie mit PLCopen von Grund auf neu zu erstellen. Durch den Einsatz von mapp Technology können Sie Ihre Anwendungsentwicklung mit deutlich weniger Code bis zu dreimal schneller abschließen, als wenn Sie sie vollständig mit PLCopen geschrieben hätten.

Das mapp Framework geht die mapp Technology einen Schritt weiter, um dem Anwender einen universellen Einstiegspunkt für die mapp Technology zu bieten. Dadurch wird die Menge an Anwendungscode, die vom Anwendungsingenieur geschrieben werden muss, noch weiter reduziert. Das Framework umfasst Programmieraufgaben und unterstützende Konfigurationsdateien mit integrierten Best Practices und Anwendungs-Know-how. Es ist modular aufgebaut, sodass der Anwender die spezifischen Teile, die für die Maschine relevant sind, einfach zu einem bestehenden Projekt hinzufügen kann.

Motivation und Ziele

Motivation und Ziele des mapp Frameworks sind:

- Qualität
 - Das Framework wurde unter Berücksichtigung von Best Practices entwickelt, die von mehreren erfahrenen Anwendungsingenieuren geprüft wurden
 - Ziel ist es, jeden mapp-Anwender auf Erfolgskurs zu bringen
- Zeitersparnis
 - Indem wir den Benutzern einen zuverlässigen Ausgangspunkt geben, senken wir die Lernkurve der mapp-Technologie
 - Neue Ingenieure können schneller hochfahren
 - Kürzere Time-to-Market
- Einfachheit
 - Das mapp Framework ist modular aufgebaut, ohne übermäßig komplex zu sein
 - Mit einem standardisierten Ansatz für die mapp-Technologie werden Anwendungsunterstützung/-übergabe und Codewartung einfacher
 - Das Framework ist entsprechend den Anforderungen der Anwendung skalierbar
- Kosteneinsparungen
 - Die Verwendung des Frameworks führt zu Kosteneinsparungen für die Maschine, da die erforderliche Anwendungsentwicklungszeit reduziert wird

Verfügbarkeit

Ein mapp Framework ist aktuell für folgende mapp Technologien verfügbar:

- mapp AlarmX
- mapp Rezept
- mapp UserX
- mapp Datei
- mapp Sicherung
- mapp Achse / Cockpit

Die Entwicklung weiterer mapp Services ist in Arbeit.

Framework-Inhalte

Jedes mapp Framework enthält Folgendes:

- Aufgabe(n) der logischen Ansicht
- Konfigurationsdatei(en)
- Hilfedateien

Die unterstützenden Hilfeseiten sind für jedes mapp Framework individualisiert. Auf diesen Seiten wird angegeben, was im Framework enthalten ist und welche Änderungen erforderlich sind, um das Framework ordnungsgemäß in eine vorhandene Anwendung einzubetten. Es ist wichtig zu beachten, dass sich die Dokumentation auf das Framework selbst konzentriert und nicht auf die Grundlagen von mapp Services / mapp Motion. Einzelheiten zu den Grundlagen der mapp Technologie finden Sie in den jeweiligen Abschnitten der Hilfe („Dienste“ „mapp Dienste“ bzw. „Motion Control“ „mapp Motion“).

WICHTIG: Jeder Abschnitt der mapp Framework-Hilfe hat eine Seite mit dem Titel "Erforderliche Änderungen". Die Schritte auf dieser Seite müssen ausgeführt werden, um das Framework in einen funktionsfähigen Zustand zu bringen!

mapp Rahmen - System Anforderungen

Das mapp Framework wird in folgenden Versionen unterstützt:

mapp Technologien	Automation Studio	Automation Runtime
5.16 oder später	4.8.2 oder später	M4.34, F4.45, C4.53, C4.63, A4.72 oder später

mapp Rahmen - Voraussetzungen

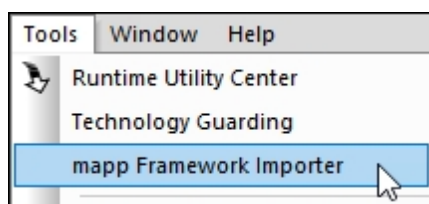
Es wird erwartet, dass der Benutzer über Grundkenntnisse in Automation Studio und der mapp-Technologie verfügt, bevor er das mapp-Framework verwendet. Daher werden folgende Schulungen als Voraussetzungen empfohlen:

- SEM210 – Automatisierungsstudio
- SEM270 – mapp-Dienste
- SEM611 – Mapp-Ansicht
- SEM415 – Mapp-Achse

Das mapp Framework ist für die Verwendung durch Anwendungsingenieure von B&R/Partnern und Kundenanwendungsingenieuren gleichermaßen konzipiert und vorgesehen. Mit anderen Worten, jeder Nutzer von mapp Technology kann das mapp Framework nutzen.

mapp Rahmen - Starten des Importers

Navigieren Sie zum Starten des mapp Framework Import Tools im Automation Studio zu Tools mapp Framework Importer:



Wenn Sie diesen Menüpunkt nach der Installation des Frameworks nicht sehen, dann schließen Sie bitte Automation Studio und öffnen Sie es erneut.

mapp Rahmen - Umgang mit Benutzerpartitionen

Standardbenutzerpartitionsdateien

Das Framework enthält einige Standarddateien (z. B. Rezeptdateien), die über eine Offline-Installation oder einen FTP-Zugriff auf die Benutzerpartition übertragen werden müssen. Diese Dateien befinden sich in der logischen Ansicht innerhalb des Pakets "UserPartition".

(Wenn der Benutzer die mapp Framework-Dateigeräte so geändert hat, dass sie einem anderen Speicherort als der Benutzerpartition entsprechen, müssen diese Dateien stattdessen an diesem neuen Speicherort abgelegt werden.)

Umschalten zwischen ARsim und Hardware

Standardmäßig verweisen alle Dateigeräte im Framework auf die Benutzerpartition (F:\).

Gehen Sie für einfache Übergänge zwischen ARsim und einer physischen SPS wie folgt vor:

1. Erstellen Sie eine neue Datei mit der Endung ".bat" (Batch-Datei)
2. Fügen Sie der Batchdatei die folgende Zeile hinzu: subst F: C:\UserPartition
3. Ersetzen Sie C:\UserPartition durch den Speicherort auf dem Laufwerk C:\ Ihres Entwicklungs-PCs, an dem Sie die Benutzerpartitionsdateien in der Simulation speichern möchten
4. Führen Sie die Batchdatei aus, bevor Sie das Projekt in der Simulation ausführen. Alternativ können Sie den Windows-Taskplaner verwenden, um die Batchdatei bei jedem Start Ihres PCs automatisch auszuführen.

Dadurch wird auf Ihrem PC ein virtuelles Laufwerk mit dem Namen F: erstellt, das Ihren

Benutzerpartitionsordner auf dem Laufwerk C funktional verwendet.

Auf diese Weise kann die Dateigerätedefinition immer F:\ verwenden, egal ob Sie in der Simulation oder auf echter Hardware laufen.

mapp Rahmen - Überlegungen zur Visualisierung

Schnittstelle zum HMI

Jedes mapp Framework hat eine Strukturvariable für Befehle, Parameter und Statusinformationen aus dem HMI. Dieser Variablenname beginnt immer mit „Hmi“ gefolgt von der mapp Technology (z. B. HmiRecipe).

Ebenso hat jedes mapp-Framework eine Aktionsdatei namens HMIActions.st, die die gesamte Programmierung im Zusammenhang mit der HMI-Schnittstelle enthält.

Wenn Sie das mapp View Frontend in das Framework importieren, dann sind die Verbindungen zu dieser Hmi-Struktur bereits für Sie vorhanden. Wenn Sie VC4 verwenden oder Ihr mapp View HMI lieber selbst gestalten, dann müssen Sie die Variablen innerhalb dieser Struktur selbst mit Ihrem HMI verbinden. Die Strukturelemente werden in der Beschreibungsspalte jeder .typ-Datei erläutert, damit Sie wissen, wie Sie sie mit Ihrem HMI verbinden. Beachten Sie jedoch, dass nicht alle Variablen/Strukturen mit einem VC4-Setup kompatibel sind (daher können zusätzliche Nacharbeiten erforderlich sein).

Demo-Seite

Damit Sie schnell und einfach durch alle importierten Inhalte in Chrome navigieren können, ist im mapp Framework eine Demoseite von mapp View enthalten. Diese Seite dient nur zu Demonstrationszwecken. Es ist nicht für die Verwendung in der endgültigen Anwendung vorgesehen.

Wenn Sie beim Import des Frameworks noch keine mapp View-Visualisierung haben, wird diese Demo-Seite als Startseite zugewiesen. Wenn Sie eine bestehende mapp View-Anwendung haben, in die das Framework integriert wird, wird die Demo-Seite zur Seitenliste hinzugefügt und Sie müssen diese Seite selbst mit einer Navigation versehen.

Administratorrechte

Die folgenden Funktionen des mapp View-Frontends sind auf den administrativen Zugriff beschränkt:

- Sicherung
 - Die Möglichkeit, ein Backup wiederherzustellen oder zu löschen und die automatischen Backup-Einstellungen zu ändern
- Datei
 - Die Möglichkeit, eine Datei auszuschneiden/zu löschen und den FIFO zu konfigurieren
- Rezept
 - Die Möglichkeit, ein Maschineneinstellungsrezept zu erstellen/löschen/laden/bearbeiten.
- BenutzerX
 - Die Möglichkeit, die Benutzer im UserList-Widget anzuzeigen und zu bearbeiten/hinzuzufügen/löschen/importieren/exportieren

mapp Rahmen - Versionsinformation

Dieser Abschnitt beschreibt die neuen Features in jeder Version des mapp Frameworks.

Themen in diesem Abschnitt:

- [Version 1.00.0](#)

mapp Rahmen - Ausführung 1.00.0

Framework	Description
mapp AlarmX	First release
mapp Axis / Cockpit	First release
mapp Backup	First release
mapp File	First release
mapp Recipe	First release
mapp UserX	First release

mapp AlarmX

Dieser Abschnitt beschreibt das mapp AlarmX Framework.

WICHTIG: Die Schritte auf der Seite „Erforderliche Modifikationen“ müssen ausgeführt werden, um das Framework in einen funktionsfähigen Zustand zu bringen!

Themen in diesem Abschnitt:

- [Merkmale](#)
- [Aufgabenübersicht](#)
- [Erforderliche Benachrichtigungen](#)
- [Optionale Benachrichtigungen](#)

mapp AlarmX Rahmen - Merkmale

Folgende Features und Funktionalitäten sind im mapp AlarmX Framework enthalten:

- 100 vorgefertigte diskrete Überwachungsalarms und ein boolesches Array zum Auslösen jedes Alarms
- Lokalisierbarer Text für jeden Alarm
- Alarmzuordnung nach Schweregrad mit Reaktionen
- Eine Abfrage zusammen mit der unterstützenden Zustandsmaschine, um große Datenmengen abfragen zu können
- Ein mapp View-Inhalt zur Anzeige der aktuellen Alarms, der Alarmhistorie und der Alarmabfrage
- Die Möglichkeit, die Alarms von der HMI aus zu bestätigen und zu exportieren

Die folgenden Beispiele sind in das Framework eingebettet:

- Beispiele für jede Art von Überwachungsalarm. Details stehen in den Kommentaren in der Aktionsdatei AlarmSamples.st (ab Zeile 5).
 - Alarmnamen: LevelMonitoringExample, DeviationMonitoringExample, RateOfChangeExample
- Ein Beispiel für die Einbindung eines Snippets in den Alarm. Dieses Beispiel ist verfügbar, damit Sie die Syntax zum Verweisen auf ein Snippet in der mapp AlarmX-Konfiguration einfach kopieren/einfügen können.
 - Alarmname: SnippetExample
- Ein Beispiel für die Verwendung von MpAlarmXAlarmControl() zum manuellen Setzen/Zurücksetzen eines Alarms aus dem Code
 - Alarmname: MpAlarmXControlExample
 - Der unterstützende Code wird in der Aktionsdatei AlarmSamples.st angezeigt (Zeile 24-32).

mapp AlarmX Rahmen - Aufgabenübersicht

Die AlarmMgr-Task enthält den gesamten bereitgestellten Framework-Code für mapp AlarmX. Diese Aufgabe befindet sich in der Logischen Ansicht innerhalb des Infrastrukturpakets. Das folgende Diagramm enthält eine Beschreibung für die Hauptaufgabe und jede Aktionsdatei.

Filename	Type	Description
AlarmMgr.st	Hauptaufgabencode	Allgemeine Handhabung von mapp-Funktionsblöcken. Behandelt die Alarmbestätigung. Behandelt den Export des Alarmverlaufs. Überprüft, ob Reaktionen aktiv sind. Ruft alle Aktionen auf.
AlarmHandling.st	Aktion	Definiert die Bedingungen, die jeden Alarm auslösen. Die AlarmMgr-Aufgabe enthält ein boolesches Array namens „Alarms“. Jeder Index des Arrays entspricht dem überwachten PV für jeden der 100 vordefinierten Alarms in der AlarmX-Konfiguration. Dies ist auch der vorgesehene Ort, um die Bedingungen zu definieren, unter denen Alarms gesperrt werden sollen, wenn die Sperrung in der Anwendung erforderlich ist.
HMIActions.st	Aktion	Zeigt die Alarmrückverfolgungsinformationen auf dem HMI an. Normalerweise muss die Backtrace-Aktion (GetBacktraceInformation) nicht geändert werden. Richtet auch die Tabellenkonfiguration für die Abfragetabelle ein.

ExecuteQuery.st	Aktion	Führt eine Abfrage aus. Enthält die unterstützende Zustandsmaschine zum Abfragen großer Datenmengen.
AlarmSamples.st	Aktion	Ruft die Variablen für die in das Framework integrierten Beispiele auf. Enthält Kommentare zur Erläuterung jedes Beispiels.

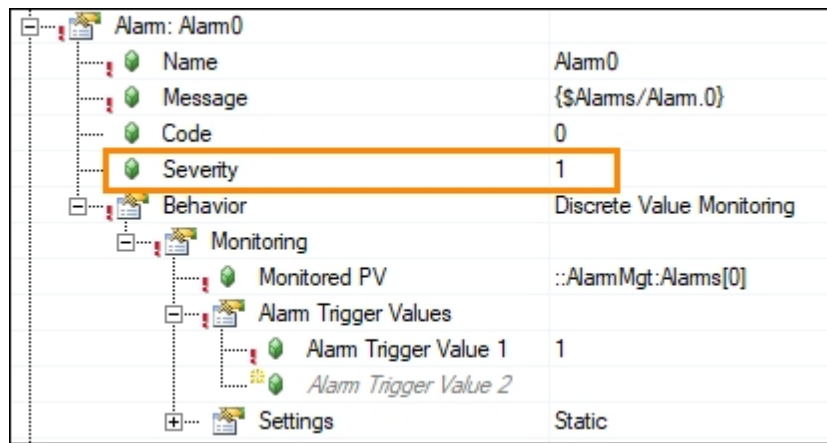
mapp AlarmX Rahmen - Erforderliche Benachrichtigungen

Das Framework bietet standardmäßig eine solide Grundlage, aber um es vollständig in Ihre Anwendung zu integrieren, müssen einige Änderungen vorgenommen werden.

Die folgende Liste von Änderungen ist erforderlich, um das Framework innerhalb der Anwendung in einen funktionsfähigen Zustand zu versetzen.

WICHTIG: Die Schritte auf dieser Seite müssen ausgeführt werden, um das Framework in einen funktionsfähigen Zustand zu bringen!

1. Definieren Sie die Bedingung zum Auslösen jedes Alarms
 - Die AlarmMgr-Aufgabe enthält ein boolesches Array namens „Alarms“. Jeder Index des Arrays entspricht dem überwachten PV für jeden der 100 vordefinierten Alarmer in der AlarmX-Konfiguration.
 - Setzen Sie für jeden dieser 100 Alarmer das Alarms[]-Bit gleich der Alarmbedingung, die für die Anwendung relevant ist. Dies geschieht in der Aktionsdatei AlarmHandling.st.
 - Wenn beispielsweise Alarme[0] ausgelöst werden sollen, wenn der Lichtvorhang unterbrochen wird und Sie sich nicht im Wartungsmodus befinden, dann:
Alarme[0] := LightCurtainInterrupted AND NOT MaintenanceMode;
2. Definieren Sie den eindeutigen Alarmtext für jeden Alarm in der Datei Alarms.tmx
 - Alarms.tmx befindet sich in der Logischen Ansicht Infrastrukturpaket AlarmX-Paket
 - Text-ID Alarm.0 enthält den Alarmtext für Alarm0 (Alarme[0]), Text-ID Alarm.1 enthält den Alarmtext für Alarm1 (Alarme[1]) usw
 - Denken Sie daran, den Text für alle Sprachen zu definieren, die für die Anwendung relevant sind
3. Weisen Sie jedem Alarm in der Alarmliste gemäß der Alarmzuordnung einen geeigneten Schweregrad zu
 - Dies erfolgt in der Konfigurationsdatei AlarmXCfg.mpalarmxcore, die sich in der Konfigurationsansicht Paket {CPU-Name} Paket mapp Services Paket AlarmX befindet
 - Standardmäßig ist der Schweregrad aller Alarmer „1“, was der „Info“-Reaktion entspricht
 - Der ausgewählte Schweregrad beeinflusst, welche Alarmreaktion ausgelöst wird, wenn der Alarm ausgelöst wird



4. Definieren Sie, wie die Anwendung auf jede Alarmreaktion reagieren soll
 - Dies geschieht über die Funktionsaufrufe MpAlarmXCheckReaction() in AlarmMgr.st (ab Zeile 62)
 - Fügen Sie innerhalb jeder Bedingung der IF-Anweisung Code hinzu, um zu definieren, wie die Maschine auf die Reaktion reagieren soll
 - Wenn zum Beispiel die Reaktion "Fehler" wahr ist, dann stoppen Sie alle Achsen; wenn die Reaktion „Warnung“ wahr ist, stoppen Sie die Maschine nach dem nächsten Zyklus; Wenn die „Info“-Reaktion wahr ist, zeige ein Pop-up auf dem HMI mit den Informationen.
 - Weitere Einzelheiten zu optionalen Änderungen bezüglich Alarmreaktionen / Alarmzuordnung finden Sie hier.
5. Wenn Sie das mapp-View-Frontend mit dem Framework importiert haben, weisen Sie den bereitgestellten mapp-View-Content (Content-ID = AlarmX_content) einem Bereich auf einer Seite innerhalb Ihrer Visualisierung zu. Wenn Sie das Frontend mapp View nicht importiert haben, dann verbinden Sie die HmiAlarmX-Strukturelemente entsprechend mit Ihrer Visualisierung (weitere Details siehe hier).

mapp AlarmX Rahmen - Optionale Benachrichtigungen

Das Framework kann bei Bedarf für die Anwendung in jeder notwendigen Weise angepasst werden. Dieser Abschnitt fasst einige optionale Änderungen zusammen, die häufig am Framework vorgenommen werden.

- Passen Sie die Quellbedingungen für die Abfrage (ActiveAlarms) („SELECT“ und „WHERE“) in AlarmXCfg.mpalarmxcore an.
- Entscheiden Sie, wie Sie mit der Situation umgehen möchten, in der das Abfrageergebnis mehr als 20 Alarmer enthält. Siehe Kommentare in ExecuteQuery.st ab Zeile 38.
- Wenn Sie Alarmer über MpAlarmXControl() oder MpAlarmXSet() auslösen, erwägen Sie das Setzen/Zurücksetzen der Alarmer in der gesamten Anwendung und nicht nur innerhalb der AlarmMgr-Aufgabe.
 - Wenn Sie beispielsweise einen Fehler im Rezeptsystem haben, können Sie diesen direkt in der Rezeptaufgabe auslösen.
 - Diese Dezentralisierung ist ein wesentlicher Vorteil von mapp AlarmX.
- Derzeit wird die bereitgestellte Abfrage nur ausgeführt, wenn auf der HMI auf eine Schaltfläche "Abfrage ausführen" geklickt wird. Wenn Sie es vorziehen, dass die Abfrage immer dann ausgeführt wird, wenn neue Daten verfügbar sind, lesen Sie die Kommentare im Status ACTIVE_ALARM_WAIT der Aktion ExecuteQuery.
- Ändern Sie in der CPU-Konfiguration das Dateigerät „mappAlarmXFiles“ auf das gewünschte Speichermedium. Standardmäßig entspricht dies der Benutzerpartition (F:\AlarmX).
 - Aktualisieren oder löschen Sie in diesem Fall die Zeilen 10-16 des INIT-Programms AlarmMgr.st, das das Verzeichnis F:\AlarmX erstellt, falls es noch nicht vorhanden ist.

- Fügen Sie weitere Alarme hinzu / löschen Sie unbenutzte Alarme (siehe hier für weitere Details)
- Passen Sie die Alarmzuordnung an (weitere Informationen finden Sie hier)
- Bestimmte Alarme unter bestimmten Bedingungen sperren (weitere Einzelheiten finden Sie hier)
- Fügen Sie zusätzliche Abfragen hinzu (weitere Informationen finden Sie hier)

Themen in diesem Abschnitt:

- [Wecker hinzufügen oder löschen](#)
- [Alarmzuordnung](#)
- [Alarme sperren](#)
- [Abfragen hinzufügen](#)

mapp AlarmX Rahmen - Wecker hinzufügen oder löschen

Wecker hinzufügen

Das Framework wird mit 100 Überwachungsalarmen für diskrete Werte geliefert. Wenn Sie mehr Einzelwertüberwachungsalarme oder andere Alarmtypen benötigen, fügen Sie diese entsprechend hinzu. Dazu:

1. Erhöhen Sie die Größe des Alarms[]-Arrays in AlarmMgr.var, je nachdem, wie viele Alarme Sie hinzufügen müssen.
2. Fügen Sie die neuen Alarme zur AlarmList in der Konfigurationsdatei AlarmXCfg.mpalarmxcore hinzu. Legen Sie die überwachten PV(s) auf die neu hinzugefügten Elemente des Arrays Alarms[] aus Schritt 1 fest.
3. Definieren Sie die Bedingung zum Auslösen jedes neuen Alarms in der Aktionsdatei AlarmHandling.st. Beachten Sie, dass es zulässig ist, verschiedene Arten von Alarmen zu mischen und abzugleichen (z. B. Überwachungsalarme mit Flanken-/Daueralarmen, die über MpAlarmXAlarmControl() oder MpAlarmXSet() ausgelöst werden).

Löschen von Alarmen

Das Framework wird mit 100 Überwachungsalarmen für diskrete Werte geliefert. Wenn Sie nicht alle 100 dieser Alarme benötigen, löschen Sie sie nach Bedarf. Dazu:

1. Verringern Sie optional die Größe des Alarms[]-Arrays in AlarmMgr.var, je nachdem, wie viele Alarme Sie entfernen möchten.
2. Löschen Sie die nicht verwendeten Alarme aus der Alarmliste in der Konfigurationsdatei AlarmXCfg.mpalarmxcore.
3. Löschen Sie die Alarmzuweisungen für die nicht verwendeten Alarme in der Aktionsdatei AlarmHandling.st. Beachten Sie, dass diese möglicherweise aus dem anfänglichen Framework-Import auskommentiert werden.

Löschen der Beispielalarme

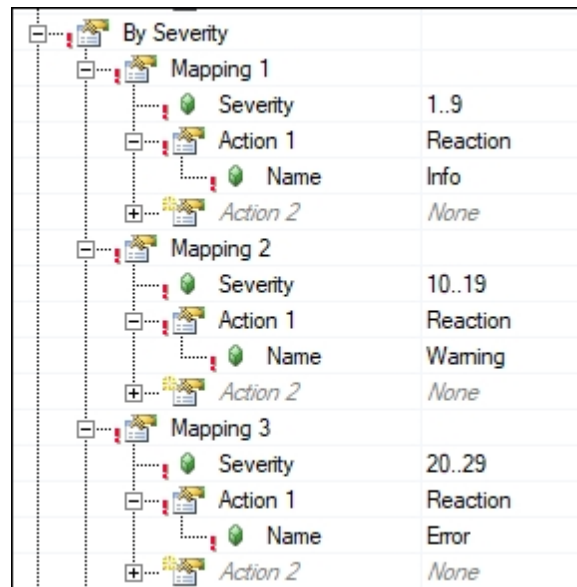
Das Framework enthält auch eine Reihe von Beispielalarmen. Wenn Sie einige (oder alle) dieser Beispiele nicht benötigen, gehen Sie wie folgt vor:

1. Entfernen Sie Mitglieder aus AlarmExamples_typ oder löschen Sie die AlarmExamples-Variable in AlarmMgr.var vollständig.
2. Löschen Sie die Beispielalarme oben in der Alarmliste in der Konfigurationsdatei AlarmXCfg.mpalarmxcore.
3. Bearbeiten Sie die Datei AlarmSamples.st oder löschen Sie sie vollständig, je nachdem, welche Alarme Sie nicht mehr benötigen. Wenn Sie diese Datei löschen, löschen Sie auch die Zeilen 19 und 60 von AlarmMgr.st, wo die Aktionen AlarmSampleInit und AlarmSampleFub aufgerufen werden.

mapp AlarmX Rahmen - Alarmzuordnung

Das Framework legt standardmäßig drei Alarmreaktionen innerhalb des Alarm-Mappings fest:

1. Info (Schweregrad 1 – 9)
2. Warnung (Schweregrad 10 – 19)
3. Fehler (Schweregrad 20 – 29)



Die Funktion MpAlarmXCheckReaction() wird für jede Reaktion innerhalb der AlarmMgr-Aufgabe aufgerufen.

```

62 // Check if any reactions are active.
63 // Typically the MpAlarmXCheckReaction() function is called from other tasks within the application.
64 // For example, the axis control task might check for the 'Error' reaction to determine whether to send a stop command to the axes.
65 // Therefore, copy/paste these IF statements to other places in the application as needed.
66 IF MpAlarmXCheckReaction(gMpLinkAlarmXCore, 'Error') THEN
67     // Error is active. Add code here to respond to error situation.
68 ELSEIF MpAlarmXCheckReaction(gMpLinkAlarmXCore, 'Warning') THEN
69     // Warning is active. Add code here to respond to warning situation.
70 ELSEIF MpAlarmXCheckReaction(gMpLinkAlarmXCore, 'Info') THEN
71     // Info is active. Add code here to respond to info situation.
72 END_IF

```


Die folgenden optionalen Änderungen sollten berücksichtigt werden, um das Framework an die Anwendungsanforderungen anzupassen:

- Passen Sie die Schweregrade an
 - Dies geschieht in der Konfigurationsdatei AlarmXCfg.mpalarmxcore
- Benennen Sie die Reaktionen um
 - Dies geschieht in der Konfigurationsdatei AlarmXCfg.mpalarmxcore
 - Denken Sie in diesem Fall daran, den Namen in den Funktionsaufrufen MpAlarmXCheckReaction() in AlarmMgr.st auf den neuen Namen zu ändern (ab Zeile 66).
 - Beispiele für alternative Reaktionsnamen:
 - SlowDownConveyor
 - Hydraulischer MotorAus
 - StopAllMotion
 - Gelbe Lampe
- Reaktionen hinzufügen / entfernen
 - Dies geschieht in der Konfigurationsdatei AlarmXCfg.mpalarmxcore
 - Wenn Sie dies tun, denken Sie daran, Funktionsaufrufe von MpAlarmXCheckReaction() in AlarmMgr.st entsprechend hinzuzufügen / zu entfernen (ab Zeile 66).
- Überprüfen Sie die Reaktionen an anderer Stelle im Code
 - Typischerweise wird die Funktion MpAlarmXCheckReaction() von anderen Tasks innerhalb der Anwendung aufgerufen. Beispielsweise könnte die Achsensteuerungstask die „Fehler“-Reaktion prüfen, um zu bestimmen, ob ein Stoppbefehl an die Achsen gesendet werden soll. Kopieren Sie daher die IF-Anweisungen, die MpAlarmXCheckReaction() enthalten, aus AlarmMgr.st nach

Bedarf um die Anwendung herum.

mapp AlarmX Rahmen - Alarmer sperren

Entscheiden Sie, ob jeder Überwachungsalarm zu irgendeinem Zeitpunkt gesperrt werden soll. Wenn ja:

1. Weisen Sie dem Alarm in der Alarmliste der Konfigurationsdatei AlarmXCfg.mpalarmxcore einen Sperr-PV zu.
 - Dies ist ein erweiterter Parameter, denken Sie also daran, auf das Symbol zu klicken .
2. Schreiben Sie in der Aktion AlarmHandling.st eine IF-Anweisung, um den Inhibit PV gemäß einer bestimmten Bedingung festzulegen.
 - Es ist üblich, denselben PV sperren zu verwenden, um mehrere Alarmer zu sperren. Beispielsweise könnten Sie ein „Wartungsmodus“-Bit haben, das, wenn es wahr ist, gleichzeitig alle Alarmer sperren würde, die während der Maschinenwartung nicht überwacht werden sollten.

Ein Beispiel für die Verwendung von Inhibit PV ist in Alarm0 integriert. Die Bedingung zum Setzen von CommissioningModeActive auf True ist noch nicht in AlarmHandling.st definiert (dies muss auf der Anwendung basieren).

Alarm: Alarm0	
Name	Alarm0
Message	{ \$Alarms/Alarm.0 }
Code	0
Severity	1
Behavior	Discrete Value Monitoring
Auto Reset	FALSE
Acknowledge	Required
Multiple Instances	TRUE
Reaction Until Acknowledged	TRUE
History Report	
Inactive to Active	TRUE
Active to Inactive	FALSE
Unacknowledged to Ackno...	TRUE
Acknowledged to Unackno...	TRUE
Update	FALSE
Monitoring	
Monitored PV	::AlarmMgt:Alarms[0]
Inhibit PV	::AlarmMgt:CommissioningModeActive
Alarm Trigger Values	
Alarm Trigger Value 1	1
Alarm Trigger Value 2	
Settings	Static

```

1
2 ACTION AlarmHandling:
3
4 // CommissioningModeActive = TRUE will inhibit Alarms[0]. The alarm will not trigger even if the alarm condition is active.
5 // CommissioningModeActive = FALSE will NOT inhibit Alarms[0]. The alarm will trigger when the alarm condition is active.
6 // See Alarm.mpalarmxcore (must enable Advanced Features)
7 CommissioningModeActive;
8
9 // Define the condition which triggers each alarm below.
10 Alarms[0];
11 // Alarms[1] := ;
12 // Alarms[2] := ;
13 // Alarms[3] := ;

```

mapp AlarmX Rahmen - Abfragen hinzufügen

Die Aktionsdatei ExecuteQuery.st enthält die Programmierung, die zum Ausführen der Abfrage erforderlich ist, die mit dem Framework geliefert wird (die „ActivateAlarms“-Abfrage).

Es enthält auch die unterstützende Zustandsmaschine, die zum Abfragen großer Datenmengen verwendet wird.

Wenn Sie eine zusätzliche Abfrage hinzufügen möchten, sind die folgenden Schritte erforderlich:

1. In AlarmMgr.var:
 1. Kopieren/fügen Sie die Variablendeklaration für QueryActiveAlarms_0 ein und geben Sie ihr einen eindeutigen Namen. (Jede Abfrage benötigt eine dedizierte Funktionsblockinstanz.)
 2. Kopieren/fügen Sie die Variablendeklaration für AlarmQuery ein und geben Sie ihr einen eindeutigen Namen.
2. In der Konfigurationsdatei AlarmXCfg.mpalarmxcore:
 1. Definieren Sie die neue Abfrage unten im Abschnitt "Datenabfragen". Geben Sie ihm einen eindeutigen Namen.
 2. Verwenden Sie die neue Abfragevariable, die Sie in Schritt 1.2 für die Prozessvariablenverbindungen und die Aktualisierungsanzahl erstellt haben.
3. Kopieren Sie die Zeilen 3-47 von ExecuteQuery.st und fügen Sie sie ein, um diesen Code zu duplizieren. Dann innerhalb des kopierten Codes:
 1. Ersetzen Sie jede Instanz von „QueryActiveAlarms_0“ durch Ihren neuen Funktionsblocknamen aus Schritt 1.1.
 2. Ersetzen Sie die Zuordnung des Abfragenamens durch den Namen Ihrer neuen Abfrage aus Schritt 2.1.
 3. NewQueryFUB_0.Name := ADR('NewQueryNameFromStep2.1');
 4. Ersetzen Sie jede Instanz von „AlarmQuery“ durch Ihren neuen Variablennamen aus Schritt 1.2.
 5. Wenn Sie möchten, dass die neue Abfrage nur basierend auf einem Tastendruck ausgeführt wird, fügen Sie einen neuen booleschen Wert in HmiAlarmX.Commands für den neuen Trigger hinzu. Aktualisieren Sie dann die IF-Anweisung innerhalb des Zustands ACTIVE_ALARM_WAIT entsprechend.

Wiederholen Sie diese Schritte für alle weiteren Abfragen.

mapp Axis Rahmen - mapp Axis / Cockpit

Dieser Abschnitt beschreibt das mapp Axis Framework.

WICHTIG: Die Schritte auf der Seite "Erforderliche Modifikationen" müssen ausgeführt werden, um das Framework in einen funktionsfähigen Zustand zu bringen!

Themen in diesem Abschnitt:

- [Merkmale](#)
- [Aufgabenübersicht](#)
- [Erforderliche Benachrichtigungen](#)
- [Optionale Benachrichtigungen](#)

mapp Axis Rahmen - Merkmale

Folgende Features und Funktionen sind im mapp Axis Framework enthalten:

- Eine einachsige Implementierung, die für mehrere Achsen wiederholt werden kann. Enthält grundlegende Befehle, manuelle/automatische Modi und eine übergreifende Zustandsmaschine für die Achsensteuerung
- Zyklischer Datenaustausch für Strom, Schleppfehler und Motortemperatur
- Detaillierte Alarmintegration mit mapp AlarmX
- Automatische Einrichtung von mapp Cockpit
- Ein Mapp-View-Inhalt zum Anzeigen des Achsen-Faceplates

Das mapp Axis-Framework ist so konzipiert, dass der Steuercode der Hauptachse innerhalb des AxisTemplate-Pakets für alle hinzugefügten Achsen wiederverwendet wird. Dadurch ist es einfach, den Gesamtprozess für alle Achsen gleichzeitig zu aktualisieren. Alle Änderungen, die an den Dateien AxisMgr.var, AxisStateMachine.st, ChangeConfiguration.st oder Recipe.st in diesem Paket vorgenommen werden, gelten für alle Achsen. Das Framework enthält dieses AxisTemplate-Paket sowie eine Anwendungsschse, die für Sie im AppAxis_1-Paket eingerichtet wurde. Informationen zum Hinzufügen zusätzlicher Achsen finden Sie hier.

Achsenspezifischer Code wird in einem dedizierten Paket für diese Achse geschrieben. Weitere Einzelheiten finden Sie hier.

mapp Axis Rahmen - Aufgabenübersicht

Die AxisMgr-Aufgabe enthält den gesamten bereitgestellten Framework-Code für mapp Axis. Diese Aufgabe befindet sich in der logischen Ansicht im MachineControl-Paket. Das folgende Diagramm enthält eine Beschreibung für die Hauptaufgabe und jede Aktionsdatei.

Filename	Type	Reference vs Unique	Description
AxisMgr.st	Hauptaufgabencode	Einzigartige / dedizierte Datei pro Achse	Allgemeine Handhabung von mapp-Funktionsblöcken. Ruft alle Aktionen auf.
AxisStateMachine.st	Aktion	Referenzierte Datei im AxisTemplate-Paket	Generische Zustandsmaschine, die für alle Achsen verwendet wird. Enthält Zustände wie Einschalten, Referenzfahrt, Hand-/Automatikbetrieb, Stoppen usw.
SimulationControl.st	Aktion	Einzigartige / dedizierte Datei pro Achse	Unterstützender Code für die Simulation. Ändern Sie s nach Bedarf für Simulationsanforderungen.
AxisControlModes.st	Aktion	Einzigartige / dedizierte Datei pro Achse	Achsspezifische Programmierung. Enthält Zustandsmaschinen für manuellen und automatischen Modus. Der manuelle Modus ist für einfaches Joggen

			eingrichtet. Der Automatikmodus ist standardmäßig leer (muss je nach Anwendung ausgefüllt werden).
Recipe.st	Aktion	Referenzierte Datei im AxisTemplate-Paket	Registriert Achsvariablen im Rezeptsystem. Fügen Sie bei Bedarf optional weitere Variablen hinzu.
ManualCommand.st	Function	Einzigartige / dedizierte Datei pro Achse	Gibt True/False zurück, je nachdem, ob ein Befehl für den manuellen Modus ausgegeben wurde. Wird in AxisStateMachine.st verwendet, um den Übergang zwischen manuellem und automatischem Modus zu handhaben. Ändern Sie die ManualCommand-Zuweisung nach Bedarf. Diese Funktion ermöglicht es der AxisStateMachine, generisch zu bleiben.
AutomaticCommand.st	Function	Einzigartige / dedizierte Datei pro Achse	Gibt True/False zurück, je nachdem, ob ein Befehl für den automatischen Modus ausgegeben wurde. Wird in AxisStateMachine.st verwendet, um den Übergang zwischen manuellem und automatischem Modus zu handhaben. Ändern Sie die AutomaticCommand-Zuweisung nach Bedarf. Diese Funktion ermöglicht es der AxisStateMachine, generisch zu bleiben.
ChangeConfiguration.st	Aktion	Referenzierte Datei im AxisTemplate-Paket	Richtet die Möglichkeit ein, Konfigurationseinstellungen zur Laufzeit zu ändern. Änderungen an dieser Aktion sind normalerweise nicht erforderlich.

mapp Axis Rahmen - Erforderliche Benachrichtigungen

Das Framework bietet standardmäßig eine solide Grundlage, aber um es vollständig in Ihre Anwendung zu integrieren, müssen einige Änderungen vorgenommen werden.

Die folgende Liste von Änderungen ist erforderlich, um das Framework innerhalb der Anwendung in einen funktionsfähigen Zustand zu versetzen.

WICHTIG: Die Schritte auf dieser Seite müssen ausgeführt werden, um das Framework in einen funktionsfähigen Zustand zu bringen!

1. Fügen Sie in der Konfigurationsansicht eine einzelne Axis-Konfigurationsdatei hinzu. Weisen Sie einen eindeutigen Mplink zu.
2. Fügen Sie in der Physikalischen Ansicht den Antrieb hinzu, der diese neue Achse steuern wird. Stellen Sie dann in der Antriebskonfiguration den Achsbezug auf den Mplink aus Schritt 1 ein.
3. Bearbeiten Sie die Dateien SimulationControl.st, AxisMgr.st, ManualCommand.st und AutomaticCommand.st gemäß den einzigartigen Anwendungsanforderungen Ihrer Achse. Einzelheiten darüber, wofür diese Dateien gedacht sind, finden Sie hier.
4. Die Achstasks sollten in zyklisch 1 laufen, und die maximal zulässige Zykluszeit beträgt 20 ms. Passen Sie die TC1-Zykluszeit entsprechend an.
5. Wenn Sie das mapp-View-Frontend mit dem Framework importiert haben, weisen Sie den mapp-View-Content (Inhalts-ID = Axis_content) einem Bereich in einer Seite innerhalb Ihrer Visualisierung zu. Wenn Sie das Frontend mapp View nicht importiert haben, dann verbinden Sie die HmiAxis-Strukturelemente

entsprechend mit Ihrer Visualisierung (weitere Details siehe hier).

mapp Axis Rahmen - Optionale Benachrichtigungen

Das Framework kann bei Bedarf für die Anwendung in jeder notwendigen Weise angepasst werden. Dieser Abschnitt fasst einige optionale Änderungen zusammen, die häufig am Framework vorgenommen werden.

- Das Framework wird mit den Achsenvorlagendateien und einer bereits eingerichteten Achse (AppAxis_1) geliefert. Um eine zusätzliche Achse hinzuzufügen, siehe hier.
- Informationen zum Umbenennen von AppAxis_1 in etwas, das seinen Zweck genauer widerspiegelt, finden Sie hier.

Themen in diesem Abschnitt:

- [Zusätzliche Achse hinzufügen](#)
- [Umbenennen AppAxis_1](#)

mapp Axis Rahmen - Zusätzliche Achse hinzufügen

Hier sind die Schritte zum Hinzufügen einer neuen Achse zum Projekt mithilfe der bereitgestellten Achsenvorlage:

1. Kopieren Sie das AxisTemplate-Paket, das sich im Infrastructure-Paket der Logical View befindet. Fügen Sie es in das MachineControl-Paket ein.
2. Benennen Sie das kopierte Paket um. (Hinweis: Für die restlichen Schritte wird das kopierte Paket als „NewAxis“-Paket bezeichnet.)
3. Benennen Sie die enthaltene AxisMgr-Aufgabe so um, dass sie mit dem neuen Paketnamen übereinstimmt. Stellen Sie diese Aufgabe in der Softwarekonfiguration bereit.
4. Löschen Sie die folgenden Dateien aus dem NewAxis-Paket:
 - AxisMgr.var
 - AxisStateMachine.st
 - ChangeConfiguration.st
 - Rezept.st
5. Fügen Sie die Dateien aus Schritt 2 oben wieder in das NewAxis-Paket als Verweis auf die Dateien im AxisTemplate-Paket ein. Mit anderen Worten: Fügen Sie aus der Toolbox viermal eine „referenzierte Datei“ hinzu (eine für jede Datei in Schritt 2). Legen Sie den Verweis so fest, dass er auf die entsprechende Datei im AxisTemplate-Paket zurückverweist.
6. In AxisInfo.tmx des NewAxis-Pakets:
 1. Ändern Sie den Namespace in etwas Einzigartiges.
 2. Ändern Sie den Wert der Text-ID „Name“, um die Achse zu beschreiben.
7. In AxisAlarms.tmx des NewAxis-Pakets:
 1. Ändern Sie den Namespace in etwas Einzigartiges.
 2. Löschen Sie alle vorhandenen Einträge. Generische Achsenalarme werden vom AxisTemplate-Paket behandelt.
 3. Fügen Sie nach Bedarf Einträge hinzu, die den Alarmmeldungen entsprechen.
8. Fügen Sie in der Konfigurationsansicht TextSystem TC.textconfig die beiden Textdateien aus Schritt 6 und 7 zur Liste „Tmx-Dateien für Ziel“ hinzu.
9. Kopieren Sie in der Konfigurationsansicht die Datei AppAxis_1.axis im mappMotion-Paket und fügen Sie sie ein. Benennen Sie die kopierte Datei um. (Beachten Sie, dass die umbenannte Datei nicht genau mit dem Namen der entsprechenden Aufgabe in der logischen Ansicht übereinstimmen kann.) Dann innerhalb der kopierten Datei:

1. Benennen Sie den MpLink wie gewünscht um
2. Aktualisieren Sie die Namespace-Verweise im Abschnitt „Alarmer“ auf die Namespaces, die Sie in den Schritten 6.1 und 7.1 ausgewählt haben. Alternativ können Sie die .axis-Datei in einem Texteditor öffnen und suchen und ersetzen. Der Namensraum wird durch das „\$“ innerhalb der ersten geschweiften Klammern identifiziert. Zum Beispiel: {\$Namespace/TextID}
10. Optional: Wenn Sie die Achse in der Simulation ausführen möchten, kopieren Sie auch die Datei VAppAxis1.purevaxcfg aus dem Paket AppAxis_1 in die Konfigurationsansicht und fügen Sie sie ein. In der eingefügten Datei:
 1. Benennen Sie den MpLink um
 2. Weisen Sie dem MpLink die Achsreferenz aus Schritt 9.1 zu.
11. Aktualisieren Sie in Logical View MachineControl-Paket NewAxis-Paket AxisMgr.st die MpLink-Referenz in den Zeilen 27 und 28 auf den Namen, den Sie in Schritt 9.1 gewählt haben.
12. Kopieren Sie in der Konfigurationsansicht das Paket mappServices das Paket AppAxis_1 und fügen Sie es ein. Benennen Sie das kopierte Paket um. Dann zu den beiden enthaltenen Dateien:
 1. Benennen Sie die .mpalarmxcore-Datei entsprechend um. Benennen Sie in dieser Datei die Datei MpLink um. Fügen Sie dann in AxisMgr.st in Zeile 37 diesen neuen MpLink ein.
 2. Benennen Sie die .mpcomgroup-Datei entsprechend um. Innerhalb dieser Datei:
 1. Benennen Sie den MpLink der Gruppe entsprechend um.
 2. Ändern Sie das Element Child 1 so, dass es mit dem neuen MpLink aus Schritt 9.1 übereinstimmt.
 3. Ändern Sie das Element Child 2 so, dass es mit dem neuen MpLink-Namen übereinstimmt, den Sie in Schritt 12.1 erstellt haben.
13. Setzen Sie in der Physikalischen Ansicht innerhalb der Antriebskonfiguration, die dieser Achse entspricht, die Achsenreferenz auf den MpLink, den Sie in Schritt 9.1 benannt haben.

Beachten Sie, dass dieser Prozess in einer zukünftigen Version des Frameworks stärker automatisiert sein wird.

mapp Axis Rahmen - Umbenennen AppAxis_1

Hier sind die Schritte zum Umbenennen der bereitgestellten AppAxis_1 in etwas spezifischeres für die Anwendung:

1. Wählen Sie einen Namen mit maximal 10 Zeichen.
2. Führen Sie ein Ersetzen in Dateien durch (Bearbeiten Suchen und Ersetzen In Dateien ersetzen), um alle Instanzen von AppAxis_1 durch Ihren neuen Namen zu ersetzen. Stellen Sie sicher, dass das Kontrollkästchen "Nur ganze Wörter" NICHT aktiviert ist und der Dateitypfilter auf *.* eingestellt ist.



3. Benennen Sie die folgenden Dateien/Pakete im Logical View MachineControl-Paket manuell in Ihren neuen Namen um:
 1. AppAxis_1-Paket
 2. AppAxis_1-Aufgabe

3. AppAxis_1_Info.tmx
4. AppAxis_1_Alarms.tmx
4. Öffnen Sie in der Konfigurationsansicht Konnektivität OpcUA Axis.uad in einem Texteditor. Ersetzen Sie in Zeile 25 AppAxis_1 durch Ihren neuen Namen.
5. Öffnen Sie in der Konfigurationsansicht TextSystem TC.textconfig in einem Texteditor. Ersetzen Sie alle Instanzen von AppAxis_1 durch Ihren neuen Namen (insgesamt 4 Instanzen).

mapp Backup Rahmen - mapp Backup

This section describes the mapp Backup Framework.

WICHTIG: Die Schritte auf der Seite "Erforderliche Modifikationen" müssen ausgeführt werden, um das Framework in einen funktionsfähigen Zustand zu bringen!

Themen in diesem Abschnitt:

- [Merkmale](#)
- [Aufgabenübersicht](#)
- [Erforderliche Benachrichtigungen](#)
- [Optionale Benachrichtigungen](#)

mapp Backup Rahmen - Merkmale

Folgende Features und Funktionen sind im mapp Backup Framework enthalten:

- Einfaches Erstellen und Wiederherstellen einer Sicherung über die HMI
- Zeigen Sie eine Liste aller verfügbaren Sicherungen von der HMI an
- Wählen Sie aus, ob das Projekt automatisch in einem bestimmten Intervall gesichert werden soll

mapp Backup Rahmen - Aufgabenübersicht

Der BackupMgr-Task enthält den gesamten bereitgestellten Framework-Code für mapp Backup. Diese Aufgabe befindet sich in der Logischen Ansicht innerhalb des Infrastrukturpakets. Das folgende Diagramm enthält eine Beschreibung für die Hauptaufgabe und jede Aktionsdatei.

Filename	Type	Description
BackupMgr.st	Hauptaufgaben code	Allgemeine Handhabung von mapp-Funktionsblöcken. Diese Aufgabe enthält den gesamten erforderlichen Code zum Erstellen, Wiederherstellen und Löschen einer Sicherungskopie des Programms.
HMIActions.st	Aktion	Unterstützender Code für HMI-Funktionalität. Lädt und speichert die Sicherungskonfiguration. Handhabung des Dateimanagers.
ChangeConfiguration.st	Aktion	Enthält die Programmierung zum Ändern der Backup-Konfiguration zur Laufzeit. Änderungen an dieser Aktion sind normalerweise nicht erforderlich.

mapp Backup Rahmen - Erforderliche Benachrichtigungen

Das Framework bietet standardmäßig eine solide Grundlage, aber um es vollständig in Ihre Anwendung zu integrieren, müssen einige Änderungen vorgenommen werden.

Die folgende Liste von Änderungen ist erforderlich, um das Framework innerhalb der Anwendung in einen funktionsfähigen Zustand zu versetzen.

WICHTIG: Die Schritte auf dieser Seite müssen ausgeführt werden, um das Framework in einen funktionsfähigen Zustand zu bringen!

1. Wenn Sie das mapp-View-Frontend mit dem Framework importiert haben, weisen Sie den mapp-View-Content (Content-ID = Backup_content) einem Bereich in einer Seite innerhalb Ihrer Visualisierung zu. Wenn Sie das Mapp-View-Frontend nicht importiert haben, dann verbinden Sie die HmiBackup-Strukturelemente entsprechend mit Ihrer Visualisierung (weitere Details siehe hier).

mapp Backup Rahmen - Optionale Benachrichtigungen

Das Framework kann bei Bedarf für die Anwendung in jeder notwendigen Weise angepasst werden. Dieser Abschnitt fasst einige optionale Änderungen zusammen, die häufig am Framework vorgenommen werden.

- Ändern Sie in der CPU-Konfiguration das Dateigerät „mappBackupFiles“ auf das gewünschte Speichermedium. Standardmäßig entspricht dies der Benutzerpartition (F:\Backup).
 - Ändern oder löschen Sie in diesem Fall die Zeilen 10-16 des INIT-Programms BackupMgr.st, das das Verzeichnis F:\Backup erstellt, falls es noch nicht vorhanden ist.

mapp File Rahmen - mapp File

Dieser Abschnitt beschreibt das mapp File Framework.

WICHTIG: Die Schritte auf der Seite "Erforderliche Modifikationen" müssen ausgeführt werden, um das Framework in einen funktionsfähigen Zustand zu bringen!

Themen in diesem Abschnitt:

- [Merkmale](#)
- [Aufgabenübersicht](#)
- [Erforderliche Benachrichtigungen](#)
- [Optionale Benachrichtigungen](#)
- [FIFO](#)

mapp File Rahmen - Merkmale

Folgende Features und Funktionen sind im mapp File Framework enthalten:

- Allgemeine Datei-Explorer-Funktionalität auf dem HMI
 - Erstellen / löschen / sortieren / umbenennen / suchen / kopieren / etc

- Die Fähigkeit, einen FIFO für ein Dateigerät zu aktivieren. Weitere Einzelheiten finden Sie hier.

mapp File Rahmen - Aufgabenübersicht

Die FileMgr-Aufgabe enthält den gesamten bereitgestellten Framework-Code für die mapp-Datei. Diese Aufgabe befindet sich in der Logischen Ansicht innerhalb des Infrastrukturpakets. Das folgende Diagramm enthält eine Beschreibung für die Hauptaufgabe und jede Aktionsdatei.

Filename	Type	Description
FileMgr.st	Hauptaufgabencode	Allgemeine Handhabung von mapp-Funktionsblöcken. Ruft alle Aktionen auf.
HMIActions.st	Aktion	Behandelt die Datei-Explorer-Operationen für die HMI. Beachten Sie, dass die Verwendung dieses Datei-Explorers ausschließlich administrativen Benutzern vorbehalten ist.
FIFOOperations.st	Aktion	Behandelt die Implementierung des FIFO (First-In-First-Out) für das ausgewählte Dateigerät.

mapp File Rahmen - Erforderliche Benachrichtigungen

Das Framework bietet standardmäßig eine solide Grundlage, aber um es vollständig in Ihre Anwendung zu integrieren, müssen einige Änderungen vorgenommen werden.

Die folgende Liste von Änderungen ist erforderlich, um das Framework innerhalb der Anwendung in einen funktionsfähigen Zustand zu versetzen.

WICHTIG: Die Schritte auf dieser Seite müssen ausgeführt werden, um das Framework in einen funktionsfähigen Zustand zu bringen!

1. Wenn Sie das mapp-View-Frontend mit dem Framework importiert haben, weisen Sie den mapp-View-Content (Inhalts-ID = File_content) einem Bereich in einer Seite innerhalb Ihrer Visualisierung zu. Wenn Sie das Frontend mapp View nicht importiert haben, dann verbinden Sie die HmiFile-Strukturelemente entsprechend mit Ihrer Visualisierung (weitere Details siehe hier).

mapp File Rahmen - Optionale Benachrichtigungen

Das Framework kann bei Bedarf für die Anwendung in jeder notwendigen Weise angepasst werden. Dieser Abschnitt fasst einige optionale Änderungen zusammen, die häufig am Framework vorgenommen werden.

- N/A

mapp File Rahmen - FIFO

Das mapp File Framework bietet die Möglichkeit, ein FIFO (first-in-first-out) für ein Dateigerät Ihrer Wahl zu aktivieren. Es gibt ein paar wichtige Details zu dieser Funktion:

- Der FIFO löscht die ältesten Dateien, sobald sich das Dateigerät zu füllen beginnt. Es stehen Ihnen zwei Konfigurationsmöglichkeiten zur Verfügung, um festzulegen, was „Tanken“ für Ihre Anwendung bedeutet. Über ein Dialogfeld auf dem HMI können Sie zwischen Folgendem wählen:
 1. Definieren Sie eine maximale Speichergröße und löschen Sie die älteste Datei, sobald der gesamte Dateigeräteinhalt diese Größe überschreitet
 2. Definieren Sie eine maximale Anzahl von Dateien und löschen Sie die älteste Datei, sobald die Anzahl der Dateien auf dem Dateigerät diesen Wert überschreitet
- Der Benutzer muss ein Dateigerät auswählen, auf dem der FIFO aktiv sein soll.
- Sobald es aktiviert ist, überprüft der FIFO dieses Dateigerät periodisch auf Größe/Anzahl von Dateien.
 - Beachten Sie, dass der FIFO Dateien nicht automatisch löscht, wenn File_content aktiv ist, da Sie zu diesem Zeitpunkt möglicherweise ein anderes Dateigerät anzeigen und der FIFO Ihre Anzeige unterbrechen würde.
- Der FIFO überwacht alle Dateien im Stammverzeichnis des ausgewählten Dateigeräts. Dateien, die in Unterordnern enthalten sind, werden nicht geprüft!

mapp Recipe Rahmen - mapp Recipe

This section describes the mapp Recipe Framework.

WICHTIG: Die Schritte auf der Seite "Erforderliche Modifikationen" müssen ausgeführt werden, um das Framework in einen funktionsfähigen Zustand zu bringen!

Themen in diesem Abschnitt:

- [Merkmale](#)
- [Design des Rezeptsystems](#)
- [Aufgabenübersicht](#)
- [Erforderliche Benachrichtigungen](#)
- [Optionale Benachrichtigungen](#)

mapp Recipe Rahmen - Merkmale







Folgende Features und Funktionen sind im mapp Recipe Framework enthalten:

- Die Infrastruktur zum Einrichten von zwei unterschiedlichen Rezeptdateien mit jeweils einer registrierten Strukturvariablen
- Die Möglichkeit, Rezepte auf einem USB-Laufwerk zu speichern
- Die Möglichkeit, ein Rezept auf dem HMI in der Vorschau anzuzeigen und zu bearbeiten, bevor es geladen wird
- Das Rezeptsystem ist im XML-Format aufgebaut. (Um in ein CSV-Rezeptsystem zu konvertieren, siehe hier.)
 - Sowohl die Konfigurationsdateien RecipeXML.mprecipexml als auch RecipeCSV.mprecipecsv sind bereits im Framework enthalten, um ein einfaches Hin- und Herwechseln zu ermöglichen. Es wird jeweils nur eine dieser Konfigurationsdateien verwendet (d. h. der Mplink von nur einer dieser Dateien wird in der Anwendung referenziert).

mapp Recipe Rahmen - Design des Rezeptsystems

- In diesem Rezeptsystem gibt es zwei Rezeptkategorien: „Parameter“ und „Maschinenkonfiguration“.
- Jede Kategorie wird in einer eigenen Rezeptdatei gespeichert.

- Für jede Kategorie wird eine Strukturvariable registriert.
 - Parameters_type enthält die Produktdaten.
 - MachineSettings_type enthält die Maschinendaten.
- Drei Variablen jedes Datentyps werden verwendet, um die Vorschaufunktionalität und die Möglichkeit zu erreichen, ein Rezept zu bearbeiten, ohne es formal zu laden. Konzentrieren wir uns beispielsweise auf die Parameterstruktur:
 - Die Variable Parameter ist die eigentliche Variablenstruktur, die das aktive Rezept enthält, das in der Anwendung verwendet werden sollte. Diese Variable wird nicht direkt im Rezept registriert.
 - Die Variable ParametersPreview ist die einzige Variable, die in der Produktkategorie des Rezepts registriert ist. Auf diese Weise können Sie Rezepte laden und in der Vorschau anzeigen, ohne sie tatsächlich in der Anwendung zu aktivieren. Wenn ein Rezept in die Anwendung geladen werden soll, wird die Parameters-Variablenstruktur vom Framework gleich der ParametersPreview-Struktur gesetzt.
 - Die Variable ParametersEdit wird als Zwischenstruktur verwendet, um das Rezept bearbeiten zu können, ohne es in die Anwendung zu laden. Es fungiert auch als Puffer zwischen den registrierten Variablen, sodass Sie Änderungen während der Bearbeitung bei Bedarf einfach verwerfen können.

Name	Type
 Parameters	Parameters_type
 ParametersEdit	Parameters_type
 ParametersPreview	Parameters_type
 MachSettings	MachineSettings_type
 MachSettingsEdit	MachineSettings_type
 MachSettingsPreview	MachineSettings_type

Beachten Sie, dass das Rezept-Framework einige While-Schleifen im Initialisierungsprogramm hat, während dies bei den anderen Frameworks nicht der Fall ist. Der Grund dafür ist, dass das Framework die Standardrezepte in das Initialisierungsprogramm lädt. Daher müssen die unterstützenden Funktionsblöcke aktiv sein, bevor der Ladebefehl erfolgreich ausgelöst werden kann. Die Standardrezepte werden eher in das Initialisierungsprogramm als in das Cyclic-Programm geladen, um die Notwendigkeit einer globalen Variablen über die Frameworks hinweg zu vermeiden, um anzuzeigen, wann das Laden der Rezepte beendet ist.

mapp Recipe Rahmen - Aufgabenübersicht

Die RecipeMgr-Aufgabe enthält den gesamten bereitgestellten Framework-Code für mapp Recipe. Diese Aufgabe befindet sich in der Logischen Ansicht innerhalb des Infrastrukturpakets. Das folgende Diagramm enthält eine Beschreibung für die Hauptaufgabe und jede Aktionsdatei.

Filename	Type	Description
RecipeMgr.st	Hauptaufgabe ncode	Allgemeine Handhabung von mapp-Funktionsblöcken. Registriert Strukturvariablen für die beiden Rezepte. Lädt die beiden Standardrezepte. Ruft alle Aktionen auf.
HMIActions.st	Aktion	Unterstützender Code für HMI-Funktionalität, wie z. B. die Rezeptvorschaufunktion. Behandelt alle vom HMI kommenden Rezeptbefehle (Laden, Speichern usw.).
FileOperations.st	Aktion	Dateihandling zum Kopieren von Rezepten zwischen der User-Partition und dem USB-Stick.

mapp Recipe Rahmen - Erforderliche Benachrichtigungen

Das Framework bietet standardmäßig eine solide Grundlage, aber um es vollständig in Ihre Anwendung zu integrieren, müssen einige Änderungen vorgenommen werden.

Die folgende Liste von Änderungen ist erforderlich, um das Framework innerhalb der Anwendung in einen funktionsfähigen Zustand zu versetzen.

WICHTIG: Die Schritte auf dieser Seite müssen ausgeführt werden, um das Framework in einen funktionsfähigen Zustand zu bringen!

1. In der Datei „mappRecipe“ müssen beim Start zwei Standardrezepte („Machine.mcfg“ für Maschineneinstellungen und „Default.par“ für Produktdaten) vorhanden sein. Erste Versionen dieser Dateien werden in der Logical View (UserPartition Recipe CSVformat und UserPartition Recipe XMLformat) zur Referenz bereitgestellt. Nur die Dateien, die sich direkt in der UserPartition Recipe-Paket befinden, werden vom Rezeptsystem geladen (standardmäßig das XML-Dateiformat).
2. Ändern Sie die Strukturtypen in RecipeMgr.typ für jede registrierte Variable, um sie den Anforderungen der Anwendung anzupassen.
 - Parameters_type wird verwendet, um die Produktdaten zu speichern
 - MachineSettings_type wird verwendet, um die Maschineneinstellungen / Inbetriebnahmedaten zu speichernWeitere Informationen zum Aufbau des Rezeptsystems finden Sie hier.
3. Wenn Sie das mapp View Frontend mit dem Framework importiert haben:
 - Weisen Sie den mapp View-Inhalt (Content-ID = Recipe_content) einem Bereich auf einer Seite innerhalb Ihrer Visualisierung zu.
 - Ändern Sie auf RecipePreviewPars.content und RecipePreviewMachConfig.content den Text auf den Beschriftungen und die Bindungen auf den NumericOutputs/TextOutputs entsprechend den Parametern, die Sie aus Ihrem Rezept anzeigen möchten.
 - Diese Inhalte werden je nach Kategorieauswahl in das Vorschaufenster auf Recipe.content geladen
 - Die Texte befinden sich in RecipePageTexts.tmx
 - Wiederholen Sie dies für ContentEditRecipe.content und ContentCreateNewRecipe.content
4. Wenn Sie das Mapp-View-Frontend nicht importiert haben, dann verbinden Sie die HmiRecipe-Strukturelemente entsprechend mit Ihrer Visualisierung (weitere Details siehe hier).

mapp Recipe Rahmen - Optionale Benachrichtigungen

Das Framework kann bei Bedarf für die Anwendung in jeder notwendigen Weise angepasst werden. Dieser Abschnitt fasst einige optionale Änderungen zusammen, die häufig am Framework vorgenommen werden.

- Umstellung auf CSV-Format (siehe hier)
- Ändern Sie in der CPU-Konfiguration das Dateigerät „mappRecipeFiles“ auf das gewünschte Speichermedium. Standardmäßig entspricht dies der Benutzerpartition (F:\Recipe).
 - Ändern oder löschen Sie in diesem Fall die Zeilen 10-16 des INIT-Programms RecipeMgr.st, das das Verzeichnis F:\Recipe erstellt, falls es noch nicht vorhanden ist.

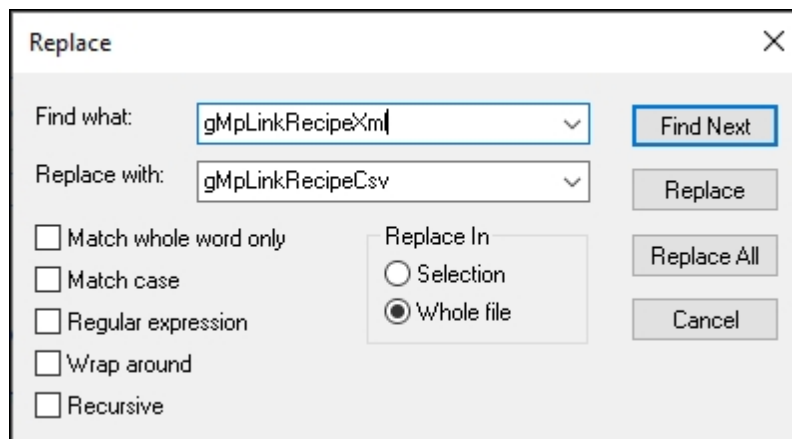
Themen in diesem Abschnitt:

- [Rezeptformat ändern](#)

mapp Recipe Rahmen - Rezeptformat ändern

Standardmäßig richtet das Framework das Rezeptsystem im XML-Format ein. Das Framework enthält Rezeptkonfigurationen sowohl für das XML- als auch für das CSV-Format, um die Umstellung zu vereinfachen. Wenn Sie zu CSV wechseln möchten, führen Sie die folgenden Schritte aus:

1. In RecipeMgr.var:
 1. Ändern Sie den Datentyp der Variablen MpRecipe_0 in MpRecipeCsv
 2. Ändern Sie den Datentyp der Variable Header in MpRecipeCsvHeaderType
2. In RecipeMgr.st:
 1. Gehen Sie zu Bearbeiten Suchen und Ersetzen Ersetzen
 2. Ersetzen Sie alle Instanzen von gMpLinkRecipeXml durch gMpLinkRecipeCsv in der gesamten Datei. Es gibt insgesamt 8 Vorkommnisse.



3. Löschen oder verschieben Sie alle vorhandenen .mcfg- und .par-Dateien aus dem Rezeptdateigerät (standardmäßig ist dies F:\Recipe), da diese im XML-Format vorliegen.
4. Kopieren Sie die Standard-Rezeptdateien, die im CSV-Format bereitgestellt werden (Logical View UserPartition Recipe CSVformat), in das Stammverzeichnis des Rezeptdateigeräts. Jetzt lädt das Rezeptsystem standardmäßig die Versionen der Dateien im CSV-Format.

mapp UserX Rahmen - mapp UserX

Dieser Abschnitt beschreibt das mapp UserX Framework.

WICHTIG: Die Schritte auf der Seite "Erforderliche Modifikationen" müssen ausgeführt werden, um das Framework in einen funktionsfähigen Zustand zu bringen!

Themen in diesem Abschnitt:

- [Merkmale](#)
- [Aufgabenübersicht](#)
- [Erforderliche Benachrichtigungen](#)
- [Optionale Benachrichtigungen](#)

mapp UserX Rahmen - Merkmale

Folgende Features und Funktionalitäten sind im mapp UserX Framework enthalten:

- Lokale Benutzerverwaltung (im Gegensatz zu Active Directory)
- Möglichkeit zum Importieren/Exportieren der Benutzerinformationen über die HMI
- Die folgenden vordefinierten Rollen: Jeder, Operator, Service, Admin
- Die folgenden vordefinierten Benutzer: Admin, Operator, ServiceTech, Anonymous

mapp UserX Rahmen - Aufgabenübersicht

Die UserXMgr-Aufgabe enthält den gesamten bereitgestellten Framework-Code für mapp UserX. Diese Aufgabe befindet sich in der Logischen Ansicht innerhalb des Infrastrukturpakets. Das folgende Diagramm enthält eine Beschreibung für die Hauptaufgabe und jede Aktionsdatei.

Filename	Type	Description
UserXMgr.st	Hauptaufgabencode	Behandelt die Interaktion mit der Benutzeroberfläche.

mapp UserX Rahmen - Erforderliche Benachrichtigungen

Das Framework bietet standardmäßig eine solide Grundlage, aber um es vollständig in Ihre Anwendung zu integrieren, müssen einige Änderungen vorgenommen werden.

Die folgende Liste von Änderungen ist erforderlich, um das Framework innerhalb der Anwendung in einen funktionsfähigen Zustand zu versetzen.

WICHTIG: Die Schritte auf dieser Seite müssen ausgeführt werden, um das Framework in einen funktionsfähigen Zustand zu bringen!

1. Wenn Sie das mapp-View-Frontend mit dem Framework importiert haben, weisen Sie den mapp-View-Content (Inhalts-ID = UserX_content) einem Bereich in einer Seite innerhalb Ihrer Visualisierung zu. Wenn Sie das Frontend mapp View nicht importiert haben, dann verbinden Sie die HmiUserX-Strukturelemente entsprechend mit Ihrer Visualisierung (weitere Details siehe hier).

mapp UserX Rahmen - Optionale Benachrichtigungen

Das Framework kann bei Bedarf für die Anwendung in jeder notwendigen Weise angepasst werden. Dieser Abschnitt fasst einige optionale Änderungen zusammen, die häufig am Framework vorgenommen werden.

- Ändern Sie bei Bedarf die verfügbaren Rollen und Benutzer.
 - Dies geschieht in den Dateien Role.role und User.user der Konfigurationsansicht
 - Wenn Sie Änderungen vornehmen, aktualisieren Sie unbedingt auch die Konfigurationsdatei UserXCfg.mpuserx
 - Beachten Sie, dass Benutzer nur dann zur Konfigurationsdatei UserXCfg.mpuserx hinzugefügt werden müssen, wenn Sie die Eigenschaften „Sprache“, „Maßsystem“ oder

„Zusätzliche Daten“ für diesen Benutzer verwenden möchten. mapp UserX hat automatisch Zugriff auf alle Benutzer, die in User.user aufgeführt sind.

- Beachten Sie auch, dass Rollen nur zur Konfigurationsdatei UserXCfg.mpuserx hinzugefügt werden müssen, wenn Sie Administrator- oder Zugriffsrechte festlegen möchten. mapp UserX hat automatisch Zugriff auf alle in Role.role aufgeführten Rollen.
- Legen Sie in der Datei User.user neue Passwörter für Admin, Operator und ServiceTech fest. Standardmäßig sind alle drei Passwörter auf 123ABc eingestellt. Legen Sie kein Kennwort für den anonymen Benutzer fest.
- Ändern Sie in der CPU-Konfiguration das Dateigerät „mappUserXFiles“ auf das gewünschte Speichermedium. Standardmäßig entspricht dies der Benutzerpartition (F:\UserX).
 - Ändern oder löschen Sie in diesem Fall die Zeilen 10-16 des INIT-Programms UserXMgr.st, das das Verzeichnis F:\UserX erstellt, falls es noch nicht vorhanden ist.