

# Wizualizacja procesu na panelu HMI

## Spis treści

1.	Wstęp .....	2
2.	Przyjęte nazewnictwo.....	2
3.	ContentInfo .....	2
	Event Binding.....	3
	Wyświetlanie stanów pracy maszyny oraz stanów PLC .....	5
	NumericOutput .....	5
4.	ContentHeader .....	6
	Event Binding.....	6
	Ustawienia Ogólne .....	6
	Login .....	7
5.	ContentNavi.....	8
	Widoczność i interaktywność.....	9
6.	ContentMainPage.....	9
	Event Binding.....	10
	NumericInput .....	13
	Świecące diody .....	13
	Widoczność i interaktywność.....	13
7.	ContentTrends.....	13
	TrendsHDA.....	14
	Widget Paper.....	14
8.	ContentVelTrends.....	15
9.	contentAlarm.....	16
	Event binding.....	17
10.	ContentHistoricAlarms .....	17
11.	contentAxisSettings.....	18
12.	System tekstów .....	18
13.	System użytkowników .....	19
	Role.....	19
	Użytkownicy .....	20

## 1. Wstęp

W niniejszej części dokumentacji opisane zostanie w jaki sposób została przygotowana wizualizacja oraz ogólny pomysł na realizowanie poszczególnych funkcjonalności w niej zawartych.

Wizualizacja została wykonana przy pomocy technologii mapView dostarczonej przez firmę B&R. Ponadto zrealizowana wizualizacja jest kompatybilna z wyświetlaczami o proporcjach rozdzielczości 16:9. Należy jednak nadmienić, że przeznaczona jest do wyświetlaczy o większej przekątnej. Wyświetlacze o mniejszej przekątnej także wyświetlą przygotowaną wizualizację jednakże niektóre jej elementy mogą nie być wystarczająco duże to płynnego nawigowania po wizualizacji.

## 2. Przyjęte nazewnictwo

W projekcie została przyjęta konwencja aby w miarę możliwości jednolicie kierować się przy nazywaniu poszczególnych elementów. Należy dodać, że przez staranie się aby ukończyć projekt w czasie nie wszędzie udało się zastosować do przedstawionej poniżej konwencji.

Konwencja opiera się na prostej metodologii nazewnictwa: Nazwa elementu składa się z dwóch członów słowo kluczowe, np. page, content oraz nazwy wyróżniającej ten element. Tak też możemy dla przykładu przytoczyć nazwę **pageHistoricAlarms**, po samej nazwie możemy domyślić się, że jest to strona dedykowana do historii alarmów.

Poniżej przedstawione zostaną główne **nazwy elementów**:

- Strony: **page<Nazwa>**
- Kontenty: **content<Nazwa>**
- Layout: **layout<Nazwa>**
- Elementy kontentów takie jak przyciski, pola do wpisywania, pola do wyświetlania tekstu, wykresy, itp.: **<Nazwa elementu><Nazwa nadana przez użytkownika>**
- Pliki Localizable Text: **<Nazwa>Translate**


Poniżej zostaną opisane kontenty zrealizowane w wizualizacji oraz funkcjonalności w nich zaimplementowane.

## 3. ContentInfo

Na opisywanym kontencie znajdują się informacje ogólne o stanach maszyny, stanach trybu automatycznego oraz stanach osi zgodnie z założeniami standardu PLCOpen. Ponadto dla każdej osi wyświetlana jest informacja o aktualnej pozycji oraz aktualnej prędkości osi, wyjątkiem jest osi wrzeczona dla, której wyświetlana jest tylko wartość prędkości osi.

Dodatkowo na tym kontencie umieszczony jest przycisk STOP-u awaryjnego oraz ikona błędu, która mruga przy aktywnym alarmie.

ContentInfo jest widoczny na każdej stronie w wizualizacji co pozwala na ciągłe monitorowanie zmian na nim zachodzących przez osobę obsługującą maszynę. Jako jeden z dwóch kontentów jest widoczny oraz interaktywny dla wszystkich użytkowników maszyny.

State of machine	DEFAULT
State of AutoMode	WAIT START
<b>Axis X</b>	
Position	0.0 mm
Velocity	0.0 mm/s
PLC Open State	AXIS_DISABLED
<b>Axis Y</b>	
Position	0.0 mm
Velocity	0.0 mm/s
PLC Open State	AXIS_DISABLED
<b>Axis Z</b>	
Position	0.0 mm
Velocity	0.0 mm/s
PLC Open State	AXIS_DISABLED
<b>Spindle</b>	
Velocity	0.0 rpm
PLC Open State	AXIS_DISABLED
	

Rysunek 1. Wygląd contentInfo w uruchomionej symulacji. Na zamieszczonym rysunku nie zaprezentowana jest ikona alarmu.

## Event Binding

W kontencie Info występuje kilka event binding-ów są one związane z STOP-em Awaryjnym oraz występowaniem alarmów.

### STOP Awaryjny

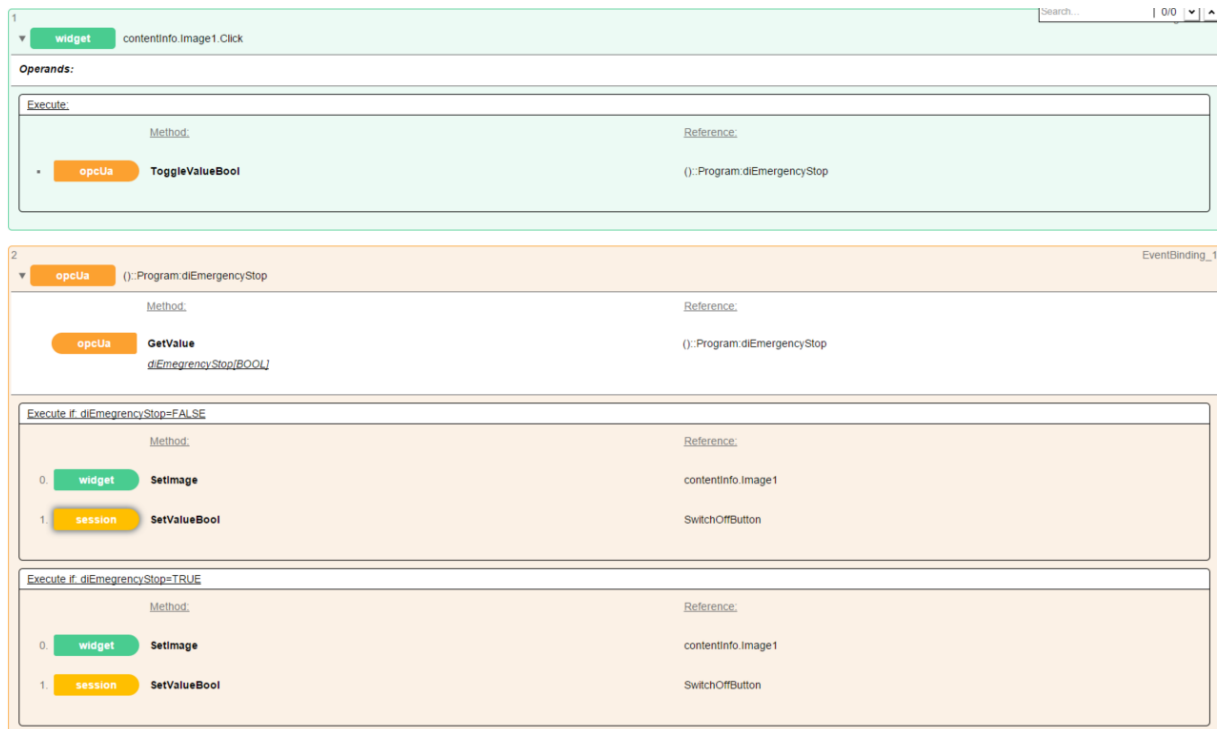
Z STOP-em Awaryjnym związane są dwa event bindingi. Pierwszym z nich związany jest z kliknięciem przycisku. Powoduje on zmianę stanu zmiennej współdzielonej po OPC UA.

Powiązany z tym zdarzeniem jest event od zmiany wartości zmiennej *diEmergencyStop*. Zmienna wartości zmiennej wywołuje 2 akcje zależne od ustalonej wartości zmiennej.

W przypadku gdy *diEmergencyStop* ustawiane jest na FALSE (przycisk jest wciśnięty) zmienia się przycisk STOP-u. Wizualnie wydaje się on wciśnięty. Realizowane jest to przez podmianę obrazu, na którym przycisk wygląda na wciśnięty.

Drugą akcją przy tym warunku jest ustawienie zmiennej sesyjnej *SwitchOffButton* na TRUE, zmienna ta jest wykorzystywana w *contentMainPage* aby ustawiać wszystkie przyciski sterujące w stan niewciśnięty.

W przypadku gdy *diEmergencyStop=TRUE* są wykonywane analogiczne akcje, jednakże ich rezultat jest przeciwny, tj. Obraz przycisku jest ustawiany na „niewciśnięty” a zmienna sesyjna otrzymuje wartość FALSE.



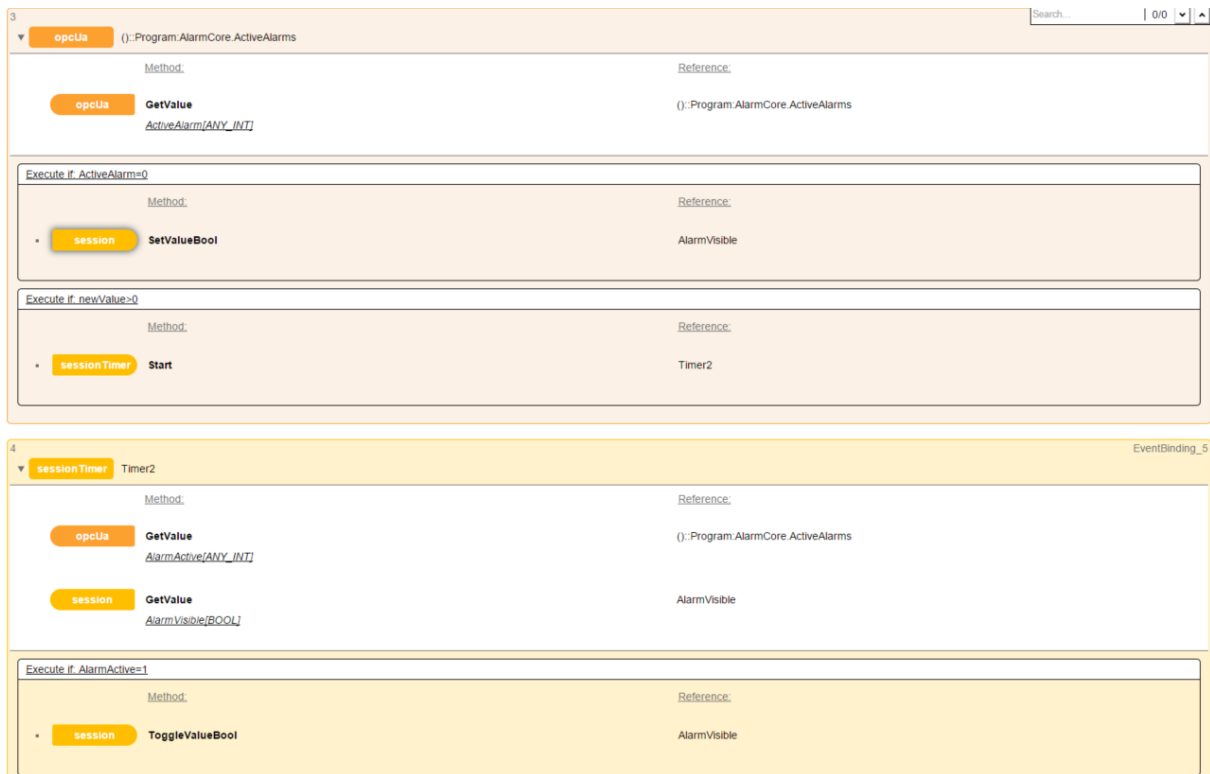
Rysunek 2. Deklaracje event bindingów związanych z przyciskiem awaryjnego STOP-u.

## Ikona błędu

Miganie ikony błędu realizowane jest z wykorzystaniem timer-a sesyjnego. W tym celu potrzebne jest zdefiniowanie w konfiguracji wizualizacji plik *Config.mappview* Timera , w naszym przypadku o okresie **1000ms**.

Aby poprawnie zaimplementować miganie należało wykorzystać 2 eventy. Pierwszym od zmiany wartości flagi *ActiveAlarms* ze zmiennej *AlarmCore*. W przypadku gdy Flaga jest nieaktywna ustawiana jest zmienna sesyjna *AlarmVisible* na FALSE, co oznacza, że ikona nie będzie wyświetlana. W przypadku gdy jest Flaga od alarmu jest aktywna uruchamiany jest timer.

Drugim zdarzeniem jest event od timer-a sesyjnego, który zmienia wartość zmiennej sesyjnej co wartość swojego okresu. Zatem okres migania ikony wynosi w naszym przypadku **2000ms**.



Rysunek 3. Deklaracje event bindingów związanych z ikoną błędu.

## Wyświetlanie stanów pracy maszyny oraz stanów PLC

Z faktu, że stany maszyny, trybu automatycznego oraz stanów PLCOpen zapisane są w formie enumeratorów nie jest możliwe wprost wyświetlanie ich przy pomocy *TextOutput*. W celu wyświetlenia ich w formie napisu zostały wykorzystane snippety.

Idea opierała się na stworzeniu pliku *.tmx* w którym zdefiniowane zostały ID stanów oraz nazwy w formie stringów. Następnie został stworzony snippet, który dostawał się do odpowiedniej nazwy stanu po numerze indeksu, który jest pobierany z enumeratora. Ostatnim elementem jest stworzenie dwóch binding-ów wpisanych na „szybko” w pliku związanych z bindingami. Jeden binding łączy zmienna przechowującą enumerator ze snippetem, natomiast drugi binding łączy snippet z wyjściem *TextOutput*. Ważne jest aby typy bindingów były jedno stronne.

```
<Binding mode="oneWay">
<Source xsi:type="opcUa" refId="::Program:MpAxisBasic_Z.Info.PLCopenState" attribute="value" />
<Target xsi:type="snippet" refId="IndexTextSnippetOpcUaZ" attribute="value" />
</Binding>
<Binding mode="oneWay">
<Source xsi:type="snippet" refId="IndexTextSnippetOpcUaZ" attribute="value" />
<Target xsi:type="brease" contentRefId="contentInfo" widgetRefId="TextOutput5" attribute="value" />
</Binding>
```

Rysunek 4. Realizacja binding-ów wykorzystywanych do wyświetlania stanów PLCOpen dla osi Z.

## NumericOutput

Widżety *NumericOutput* zostały wykorzystane w ramach tego kontentu do wyświetlania wartości pozycji 3 osi CNC oraz prędkości na wszystkich osiach.

W celu wyświetlania wartości z jednostkami oraz z wartościami dynamicznie zmieniającymi się w zależności od wybranego systemu metrycznego wszystkie bindingi do tych widżetów zostały zrealizowane jako połączenia przez **node**.

## 4. ContentHeader

Kontent ten jest wyświetlany na każdej stronie. Pozwala on na logowanie się użytkowników, zmianę ogólnych ustawień. Prezentuje się też na nim logo firmy B&R. Jako jeden z dwóch kontentów jest widoczny oraz interaktywny dla wszystkich użytkowników maszyny.

### Event Binding

W tym kontencie znajdują się dwa eventy. Oba związane są kliknięciem w odpowiednią ikonę lub przycisk oraz otwarciu okna dialogowego.



*Rysunek 5. Widok contentHeader na wizualizacji.*

### Settings

Eventem związanym z ustawieniami jest zdarzenie od kliknięcia ikony „Ustawień”. Powoduje ono otwarcie okna dialogowego z ustawieniami. Okno to zostanie opisane w dalszej części rozdziału.

### Login

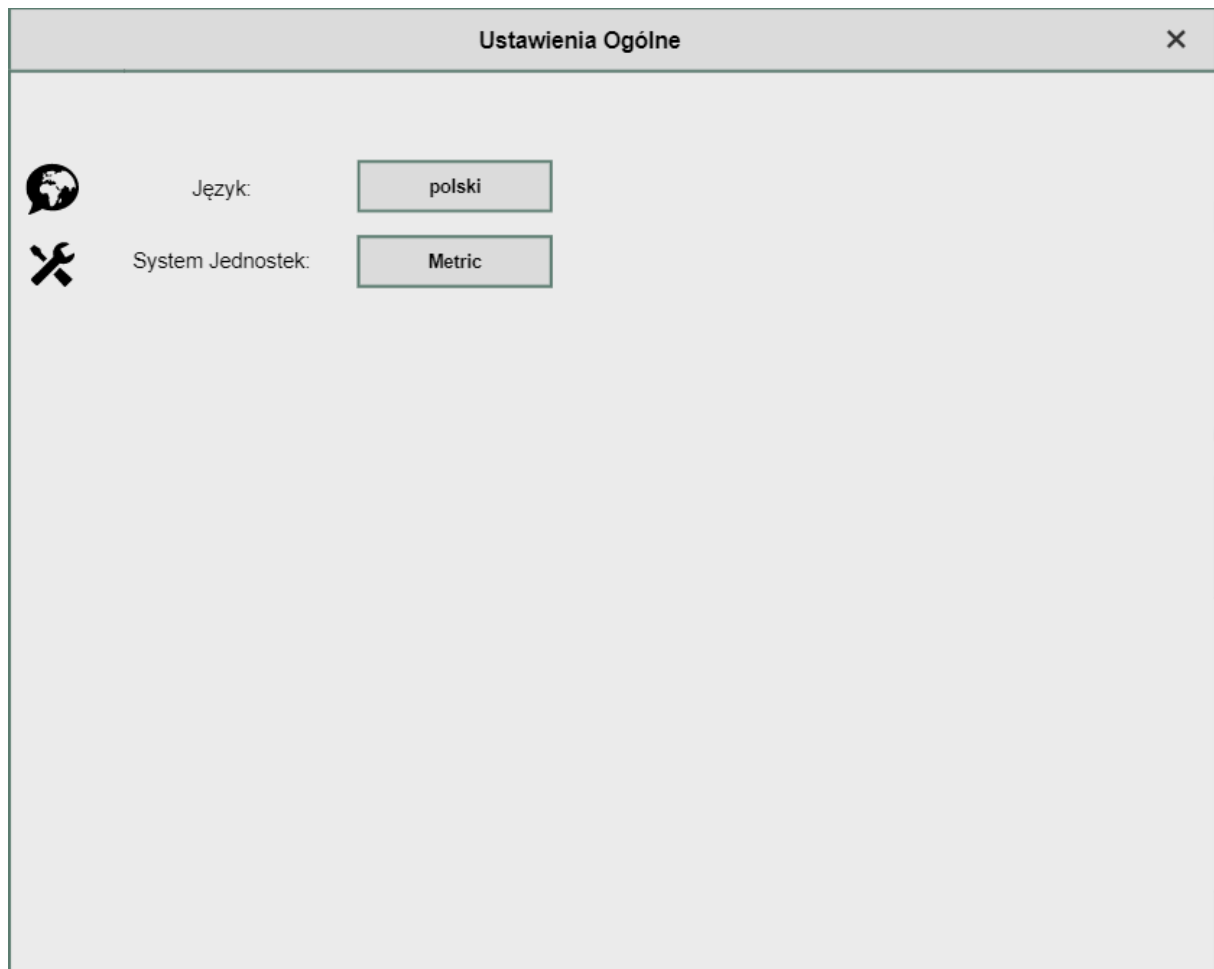
Event od kliknięcia w przycisk „Login” powoduje otwarcie okna dialogowego do Logowania użytkowników. Okno dialogowe zostanie opisane w dalszej części rozdziału.

### Ustawienia Ogólne

Ustawienia Ogólne są realizowane przez okno dialogowe, na którym są zamieszczone możliwości **zmiany języka** oraz zmiana **systemu jednostek**.

W celu wyświetlenia tego okna dialogowego wcześniej został zdefiniowany *layoutDialogSetting*, który określa wielkość wyświetlanego okna.

W oknie dialogowym zostały wykorzystane ikony z biblioteki Symbol oraz selektory do wyboru języka oraz systemu jednostek.



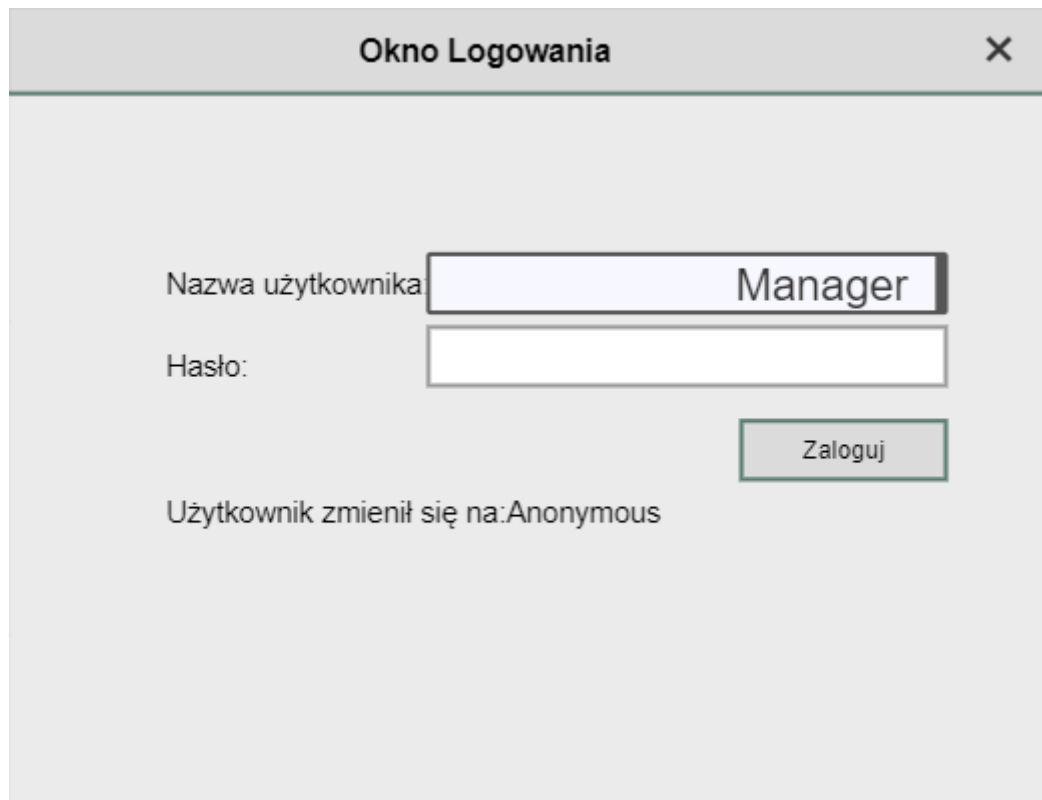
Rysunek 6. Widok okna dialogowego ustawień ogólnych.

## Login

Sekcja logowania jest najbardziej rozbudowaną sekcją w kontencie Header. Oprócz ikon pomagających rozpoznać przyciski zalogowywania oraz wylogowywania, a także ikony użytkownika zawierają wcześniej przyciski „Login”, „Logout” oraz widget do wyświetlania aktualnie zalogowanego użytkownika.

Jak zostało wspomniane w sekcji *Event binding* wciśnięcie przycisku „Login” powoduje otwarcie okna dialogowego, w którym możliwe jest wpisanie danych do logowania. Okno posiada także funkcję automatycznego zamknięcia w przypadku, gdy wprowadzone dane są poprawne/logowanie zostało zakończone sukcesem. Zrealizowane zostało to przy pomocy event binding-u.

Ponadto użyty zostały przycisk „Logout”, który w sobie ma zapisaną funkcjonalność automatycznego wylogowania użytkownika po wciśnięciu tego przycisku.



Okno Logowania

Nazwa użytkownika

Hasło:

Zaloguj

Użytkownik zmienił się na: Anonymous

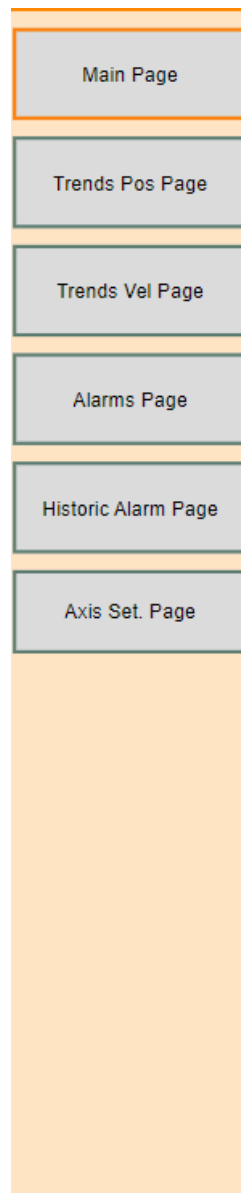
*Rysunek 7. Widok okna dialogowego do logowania użytkowników.*

## 5. ContentNavi

Jest to kontent nawigacyjny, pozwalający na przechodzenie między stronami wizualizacji. Kontent jest widoczny na wszystkich stronach. Jednakże nie wszystkie elementy z tego kontentu są widoczne lub interaktywne dla wszystkich użytkowników.

W kontencie tym znajdują się tylko przyciski nawigacyjne, które przenoszą użytkownika na wybraną stronę.





Rysunek 8. Widok contentNavi z wszystkimi widocznymi oraz aktywnymi przyciskami nawigacyjnymi.

## Widoczność i interaktywność

Widoczny i możliwy do wciśnięcia dla wszystkich użytkowników jest przycisk *Strony Główniej*. Natomiast przycisk *Strony Ustawień Osi* możliwy do interakcji jest dla **Manager-a** (osoby o najwyższych uprawnieniach).

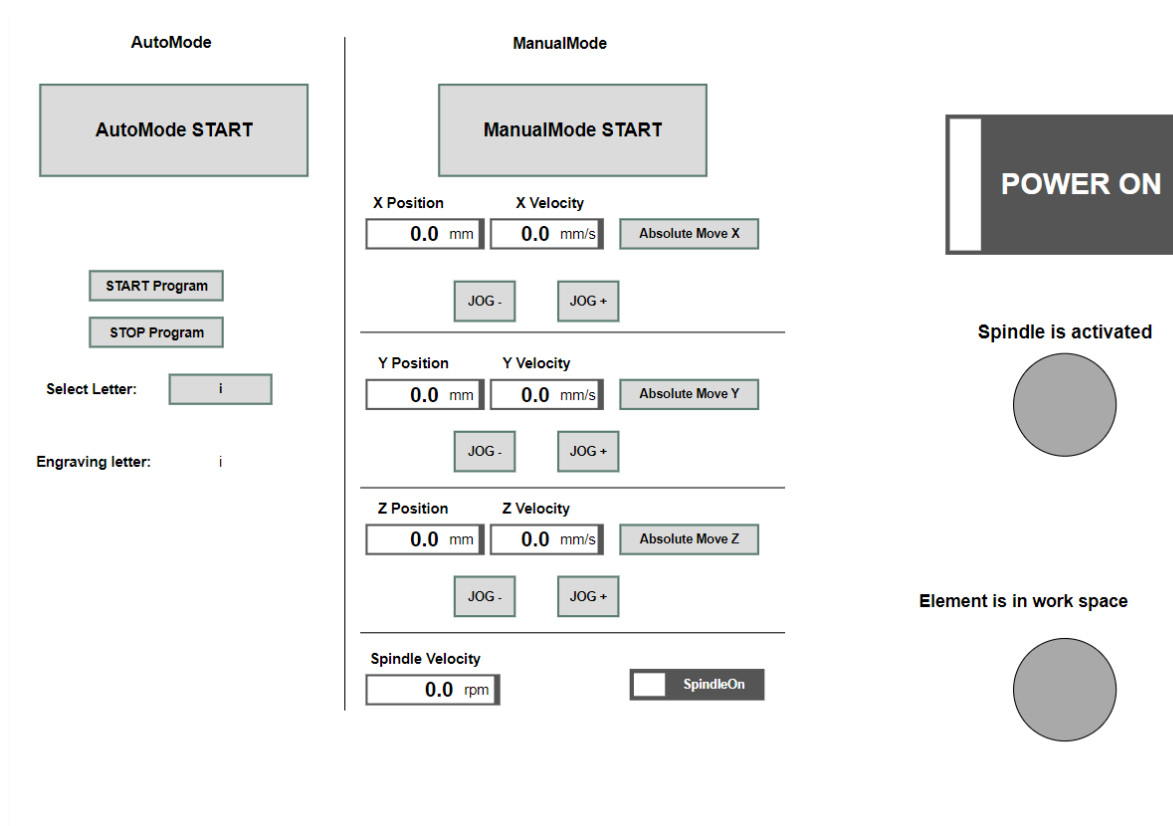
Pozostałe przyciski, tj. do strony z wykresami pozycji i prędkości oraz alarmów i historii alarmów są możliwe do kliknięcia dla **Operatora**, **Serwisanta** oraz **Manager-a**.

## 6. ContentMainPage

Jest go główny kontent wizualizacji, na którym realizowana jest główna logika związana z wykonywaniem programu. Na tym kontencie możemy wyróżnić 3 sekcje.

Sekcje związaną z trybem automatycznym, trybem manualnym oraz sekcje zasilającą i sygnalizacyjną.

Widoczność elementów tej strony oraz możliwość interakcji z nimi jest uzależniona od przypisanej roli użytkownika.



Rysunek 9. Widok *contentMainPage* z aktywnymi i widocznymi wszystkimi widgetami.

## Event Binding

W prezentowanym kontencie znajduje się najwięcej event bindingów. Jednakże mimo, że jest ich tak duża ilość możemy je zgrupować do kilku grup o wspólnych funkcjonalnościach.

### Wyłączanie przycisków po wciśnięciu przycisków

Mowa jest tu o wykluczaniu się przycisków, tj. gdy wciśnięty zostanie przycisk *AutoMode START* przycisk *ManualMode START* jest ustawiany jako niewciśnięty. Zachowanie takie zachodzi także dla pary *START Program* – *STOP Program*.

### Wyłączenie przycisków po aktywacji Emergency STOP

Realizacja tego następuje przez zmienną sesyjną *SwitchOffButton*, która jest ustawiana w *contentIfno*, przez wciśnięcie STOP-u awaryjnego.

18

EventBinding\_

session

SwitchOffButton

Operands:

Execute if: newValue=TRUE

Method

Reference

0.	widget	SetValue	contentMainPage.NumericInputXVel
1.	widget	SetValue	contentMainPage.NumericInputXPos
2.	widget	SetValueBool	contentMainPage.ToggleButtonAutoMode
3.	widget	SetValueBool	contentMainPage.ToggleButtonManualMode
4.	widget	SetValueBool	contentMainPage.ToggleButtonStartProgram
5.	widget	SetValueBool	contentMainPage.ToggleSwitch1
6.	widget	SetValueBool	contentMainPage.ToggleSwitchSpindleOnOff
7.	opcUa	SetValueBool	()::Program.ConfXPos
8.	opcUa	SetValueBool	()::Program.ConfYPos
9.	opcUa	SetValueBool	()::Program.ConfZPos
10.	widget	SetValue	contentMainPage.NumericInputYPos
11.	widget	SetValue	contentMainPage.NumericInputYVel
12.	widget	SetValue	contentMainPage.NumericInputZPos
13.	widget	SetValue	contentMainPage.NumericInputZVel
14.	widget	SetValue	contentMainPage.NumericInputSpindleSpeed

Rysunek 10. Deklaracja akcji po zdarzeniu zmiany wartości zmiennej sesyjnej na stan wysoki.

## Ustawianie wartości zmiennej przy trzymaniu przycisku

Eventy te wykorzystywane są do pracy w trybie JOG. Wciśnięcie odpowiedniego przycisku JOG oraz trzymanie go powoduje, że zmienna ustawiana jest w stan wysoki, natomiast puszczenie ustawia zmienną w stan niski. W akcji tych zdarzeń wykorzystywane jest ustawianie wartości zmiennej udostępnianej przez OPC UA.

10

▶ widget contentMainPage.PushButtonJOGYNeg.MouseUp

contentMainPage.PushButtonJOGYNeg.MouseUp

11

▼ widget contentMainPage.PushButtonJogY.MouseDown

contentMainPage.PushButtonJogY.MouseDown

Operands:

Execute:

	Method	Reference
•	opcUa SetValueBool	()::Program.JogPlusYActive

Rysunek 11. Przykład realizacji ustawiania wartości zmiennej przy zdarzeniu na trzymanie przycisku oraz resetowania tej wartości po puszczeniu przycisku.

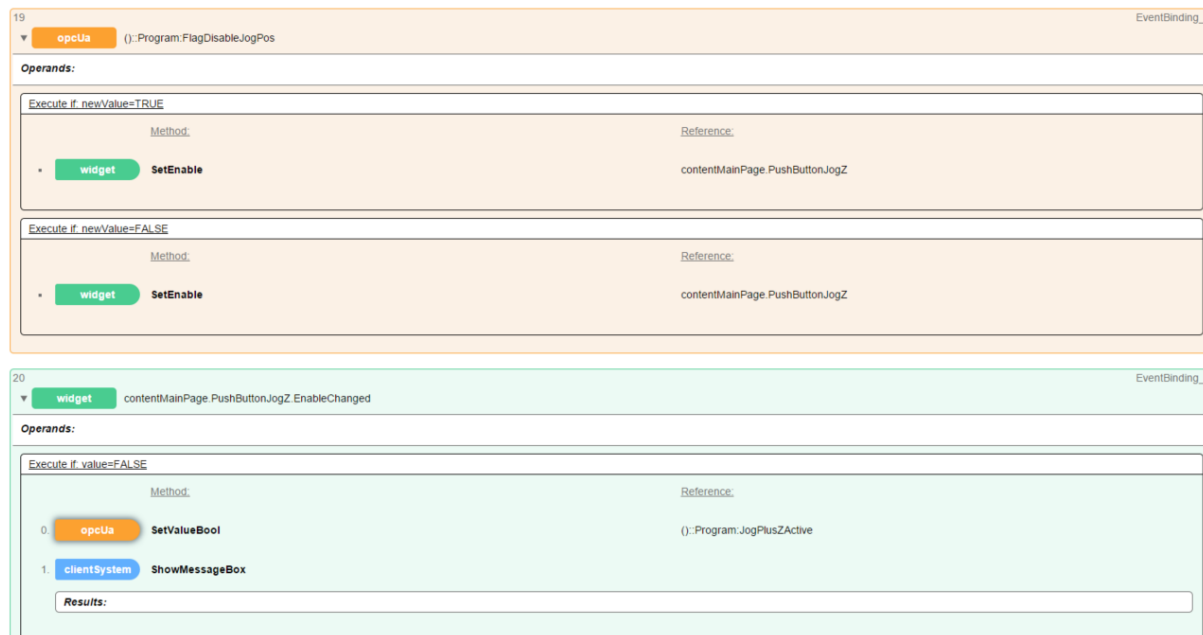
## Wyłączanie dostępu do przycisków JOG

Realizacja tego zadania pozwala na uniknięcie sytuacji nieumyślnego przekroczenia limitów osi przez zbyt długie przytrzymanie przycisku JOG.

Realizacja oparta jest na dwóch eventach. Pierwszym jest event od zmiany wartości flagi, która określa czy osź znajduje się w pozycji bliskiej limitu osi. Jeśli flaga jest w stanie wysokim dostępność przycisku jest wyłączana przez akcje na widgecie i ustawieniu *SetEnable*. Natomiast gdy jest w stanie niskim dostępność elementu ustawiana jest na TRUE.

Drugim zdarzeniem jest zdarzenie od zmiany dostępności. Gdy przycisk nie jest dostępny wyświetlany jest komunikat, że osź jest blisko limitu osi i możliwy jest tylko ruch przez zadanie wartości pozycji.

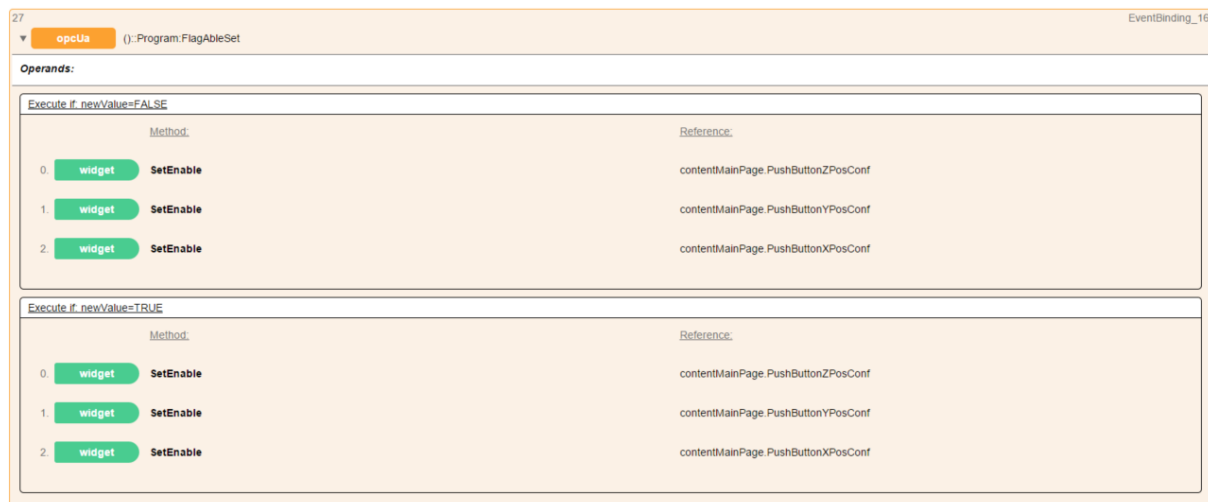
Drugą akcją w tym zdarzeniu jest zresetowanie zmiennej podpiętej pod przycisk, w celu wyłączenia ruchu w trybie JOG w danym kierunku.



Rysunek 12. Przykładowa konfiguracja zdarzeń wyłączenia dostępności przycisku JOG w pobliżu limitu osi.

### Wyłączenie dostępności przycisków w przypadku ruchu,

Zdarzenie z tym związane jest inicjalizowane przez flagę określającą czy ruch w osiach X, Y, Z jest aktywny. Jeśli tak to przyciski potwierdzania ruchu absolutnego wyżej wymienionych osi są nieaktywne. Jest to dodatkowe zabezpieczenie aby nie było możliwe wywołanie ruchów w którejs tych osi gdy inna się porusza.

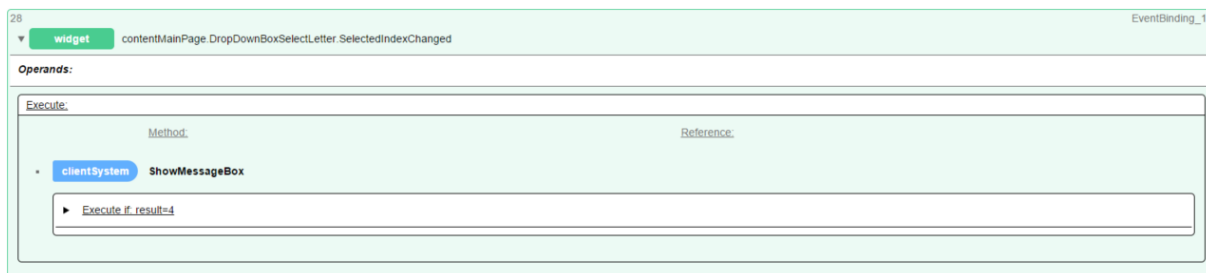


Rysunek 13. Konfiguracja event binding-u dla wyłączania dostępności w przycisków aktywacji ruchu absolutnego osi X, Y, Z w przypadku gdy któraś z tych osi jest w ruchu.

### Wyświetlanie informacji o zmianie litery

Po zmianie litery w drop boxie użytkownik otrzymuje informacje o dokonanej zmianie.

Zostało to zrealizowane na zdarzeniu od zmiany wybranego indeksu.



Rysunek 14. Konfiguracja eventu związanego ze zmianą litery.

## NumericInput

Wszystkie okna NumericInput zamieszczone na contenMainPage zostały połączone ze zmiennymi przez node, co pozwala na poprawne wyświetlanie jednostek i konwersje danych po zmianie systemu jednostek. Dodatkowo z wykorzystaniem zakresów zdefiniowanych w OPC UA wartości mogą być wpisywane w odpowiednich zakresach co pozwala nam na uniknięcie przekroczenia limitów osi.

## Świecące diody

Na kontencie znajdują się dwie lampki/diody sygnalizujące znajdujący się obiekt w przestrzeni roboczej oraz aktywne wrzeciono. W celu prezentacji tych lampek w dwóch kolorach (włączona/wyłączona) wykorzystałem dwa takie same obiekty o różnych kolorach i nałożyłem jeden na drugi. Kolor nieaktywny jest widoczny cały czas i ułożony jest pod spodem. Natomiast obiekt z kolorem aktywnym jest ustawiany na widoczny, tylko gdy występuje zdarzenie powodujące załączenie lampek.

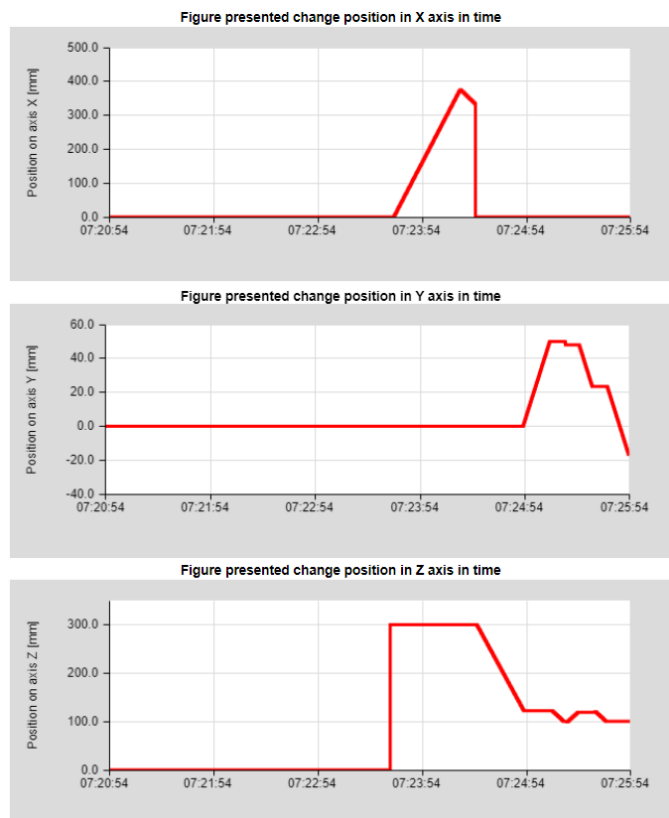
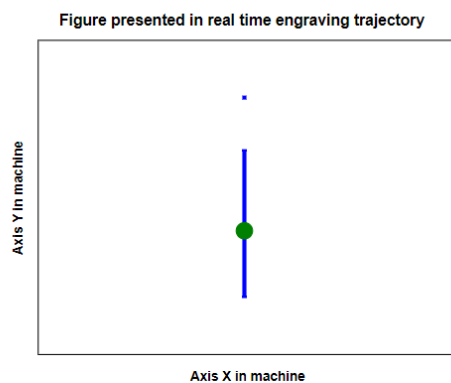
## Widoczność i interaktywność

ContentMainPage jest najbardziej zróżnicowany pod względem widoczności i interaktywności elementów ze względu na role użytkownika. Tak więc rola:

- **Everyone** – widzi tylko przycisk *PowerOn* oraz lamki sygnalizacyjne, jednakże nie może wejść w interakcje z przyciskiem
- **Operator** - Posiada widoczność i interaktywność przycisków i elementów sekcji *AutoMode* oraz przycisku *PowerOn*
- **Service** – Posiada widoczność i interaktywność wszystkich elementów kontentu
- **Manager** – tak jak Service

## 7. ContentTrends

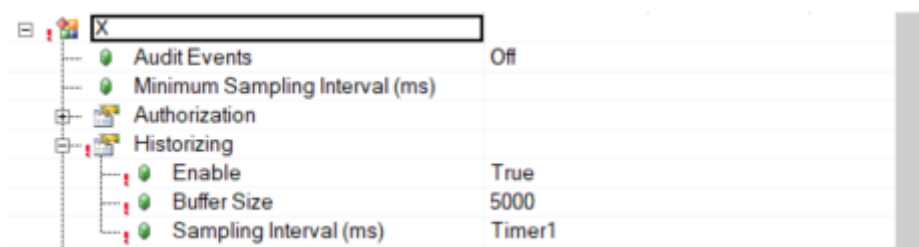
Jest to kontent zawierający wykresy prezentujące zmianę pozycji w osiach w czasie. Zawiera także widget Paper, na którym jest wyświetlany aktualnie wykonywany program CNC.



Rysunek 15. Widok contentTrends podczas wykonywania grawerunku litery "i".

## TrendsHDA

Są to widgety które pozwalają na prezentację zmian położenia osi w czasie. W celu zapisywania zmian przy nie aktywnym kontencie włączone zostało zapisywanie danych w buferze. Pozwala to na aktualizację wykresu mimo nie aktywnej strony z wykresami.



Rysunek 16. Konfiguracja zmiennej pozycji X w konfiguracji OPC UA. Włączenie buferowania zmiennej.

W celu prawidłowego wyświetlania jednostek oraz konwertowania wartości w zależności od systemu jednostek wszystkie bindingi zostały zrealizowane przez node.

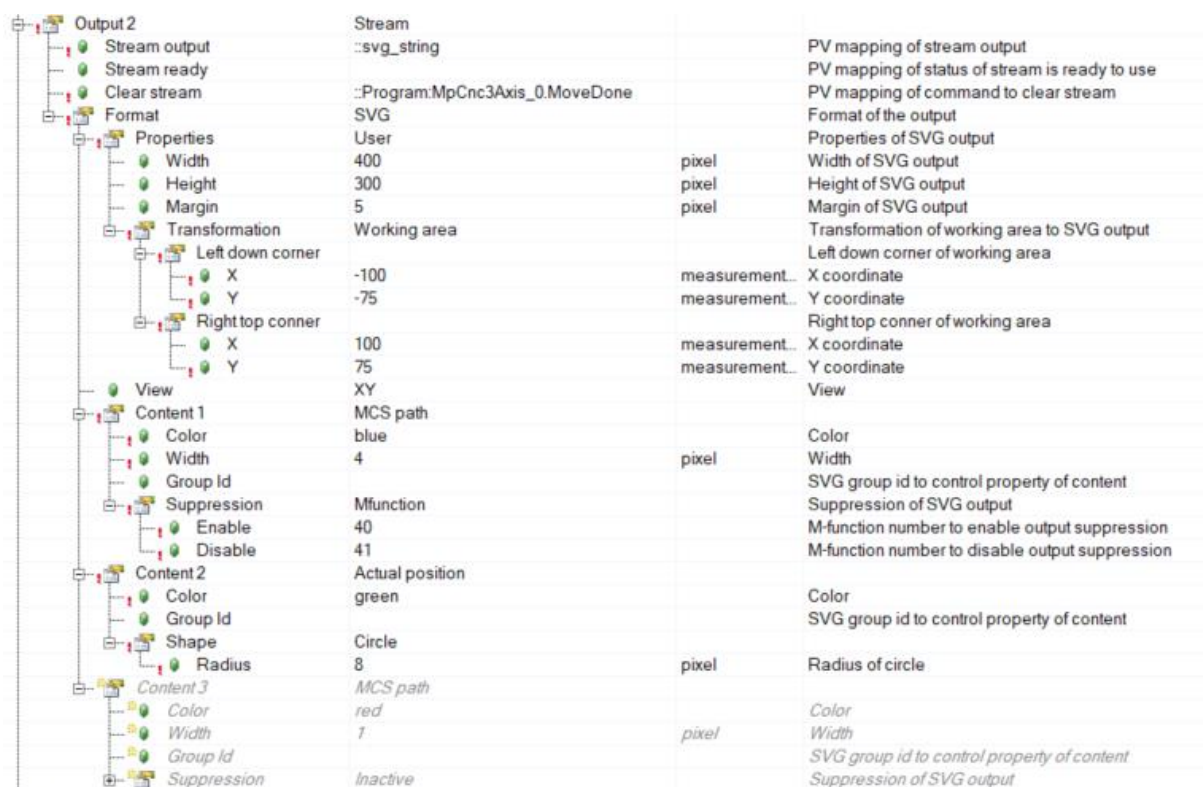
## Widget Paper

Widget pozwala na aktualne rysowanie trajektorii generowanej przez kody g-code.

W celu jego wykorzystania należy wcześniej skonfigurować w ustawieniach CNC feature PathPreview, a także stworzyć zmienną string o wystarczająco dużej ilości miejsca na zapisywanie pliku SVG.

W aplikacji został wykorzystany fragment do rysowania fragmentów g-code, które będą grawerowane z wyprzedzeniem oraz aktualna pozycja wrzeciona. Dodatkowo wykorzystane zostały

M-funkcje, które pozwalają na zapisywanie pozycji w pliku svg oraz blokują zapisywanie. W tym celu zdefiniowane zostały funkcje M40 oraz M41.

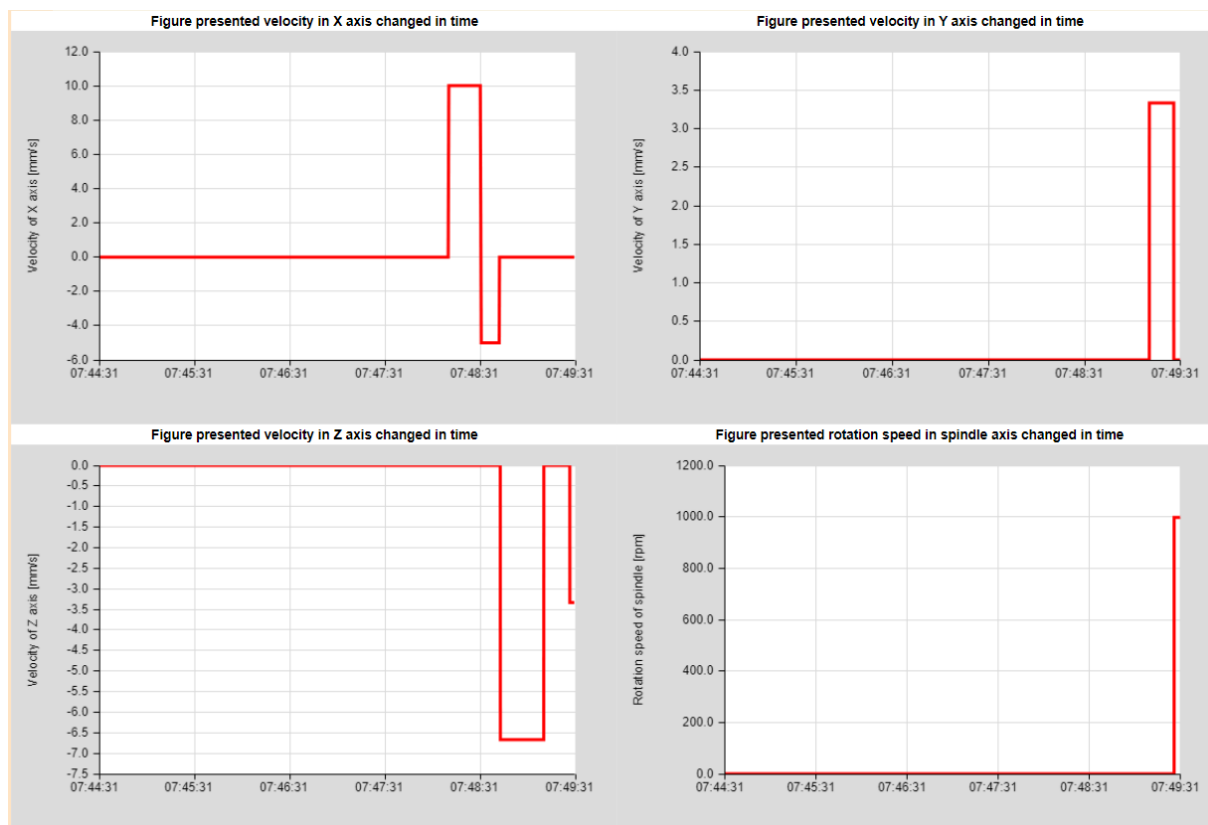


Rysunek 17. Konfiguracja PathPreview w konfiguracji Axesgroupfeature.

## 8. ContentVelTrends

Podobnie jak contentTrends składa się z samych wykresów, jednakże wykresy na tym kontekście prezentują zmianę wartości prędkości wszystkich osi.

Wszystkie połączenia zrealizowane zostały przez node, w celu poprawnej konwersji wartości przy zmianie jednostek. Zmienne wyświetlane zostały także zapisywane do bufora. Ponadto tak jak w contentTrends oś czasu została ustawiona na długość 5 minut.



Rysunek 18. Widok contentVelTrends przy aktywnym trybie automatycznym i wykonywanym programie.

## 9. contentAlarm

Kontent ten zawiera tylko kilka elementów: listę alarmów oraz dwa przyciski do potwierdzenia pojedynczych alarmów oraz potwierdzenia wszystkich zarejestrowanych alarmów.

Date/Time	Name of alarm	Alarm code	Description
Wednesday, July 27, 2022 7:53:04 AM	EmergencyStop	9999	Emergency stop is active
Wednesday, July 27, 2022 7:52:59 AM	EmergencyStop	9999	Emergency stop is active
Wednesday, July 27, 2022 7:50:23 AM	Position in Z	9990	Dangerous position The Z axis is near the axis upper limit. JOG mode is not active. To get to the exact position, enter the value in the window.
Wednesday, July 27, 2022 7:26:29 AM	Position in Z	9990	Dangerous position The Z axis is near the axis upper limit. JOG mode is not active. To get to the exact position, enter the value in the window.
Wednesday, July 27, 2022 7:23:28 AM	Position in Z	9990	Dangerous position The Z axis is near the axis upper limit. JOG mode is not active. To get to the exact position, enter the value in the window.
Wednesday, July 27, 2022 7:18:56 AM	Position in Z	9990	Dangerous position The Z axis is near the axis lower limit. JOG mode is not active. To get to the exact position, enter the value in the window.

Previous
1
Next

Acknowledge Alarm
Acknowledge All Alarm

Rysunek 19. Widok kontentu z aktywnymi alarmami.



## Event binding

W związku, że na kontencie są przyciski do potwierdzania alarmów występują zdarzenia od kliknięć tych przycisków.

Zdarzeniem od kliknięcia przycisku „*Acknowledge Alarm*” jest akcja, w której potwierdzony zostaje zaznaczony alarm. Natomiast zdarzeniem od kliknięcia przycisku „*Acknowledge All Alarm*” wywołuje akcje potwierdzenia wszystkich zarejestrowanych alarmów. W obu przypadkach Potwierdzenie następuje po kliknięciu odpowiedniego przycisku na wyskakującym oknie dialogowym.

The image displays two screenshots of a configuration tool, likely for a PLC or industrial control system, showing the configuration of event bindings for alarm confirmation buttons.

**Screenshot 1 (Top):** Shows the configuration for the event binding triggered by the click of the 'contentAlarm.Button2' widget. The configuration is set to 'Execute' when the widget is clicked. The action is 'clientSystem.ShowMessageBox'. Below this, there is a section 'Execute if result=1' which contains two actions:

	Method:	Reference:
0.	widget.AcknowledgeAll	contentAlarm.AlarmList1
1.	opcUa.SetValueBool	()::Program.FlagEstop

**Screenshot 2 (Bottom):** Shows the configuration for the event binding triggered by the click of the 'contentAlarm.Button1' widget. The configuration is set to 'Execute' when the widget is clicked. The action is 'clientSystem.ShowMessageBox'. Below this, there is a section 'Execute if result=1' which contains three actions:

	Method:	Reference:
0.	widget.Acknowledge	contentAlarm.AlarmList1
1.	opcUa.SetValueBool	()::Program.FlagEstop
2.	opcUa.SetValueBool	()::Program.FlagConfirmError

Rysunek 20. Konfiguracja zdarzeń i akcji związanych z potwierdzaniem alarmów przez kliknięcie przycisków.

## 10. ContentHistoricAlarms

Kontent ten zawiera tylko jeden element jakim jest lista historii alarmów.

Date/Time	Name of alarm	Alarm code	Old State	New State
Wednesday, July 27, 2022 8:57:55 AM	EmergencyStop	9999		
Wednesday, July 27, 2022 8:57:33 AM	EmergencyStop	9999		
Wednesday, July 27, 2022 8:57:26 AM	EmergencyStop	9999		
Wednesday, July 27, 2022 8:51:02 AM	Position in Z	9990		
Wednesday, July 27, 2022 7:53:04 AM	EmergencyStop	9999		
Wednesday, July 27, 2022 7:52:59 AM	EmergencyStop	9999		
Wednesday, July 27, 2022 7:50:23 AM	Position in Z	9990		
Wednesday, July 27, 2022 7:26:29 AM	Position in Z	9990		
Wednesday, July 27, 2022 7:23:28 AM	Position in Z	9990		
Wednesday, July 27, 2022 7:18:56 AM	Position in Z	9990		
Wednesday, July 27, 2022 7:13:35 AM	Position in Z	9990		

Previous
1
Next

Rysunek 21. Widok historii alarmów w `contentHistoricAlarms`.

## 11. `contentAxisSettings`

Kontent zawiera pola numeryczne pozwalające na wpisywanie nowych wartości pod zmienne odpowiadające za parametry osi.

Wszystkie połączenia są realizowane przez node, ma to na celu prawidłowe skalowanie wartości przy zmianie systemu jednostek.

Slow Velocity Axis X	5.0 mm/s	Velocity of Axis Y	10.0 mm/s	Velocity of Axis Z	10.0 mm/s
Fast Velocity Axis X	10.0 mm/s	Acceleration of Axis Y	25.0 mm/s <sup>2</sup>	Acceleration of Axis Z	25.0 mm/s <sup>2</sup>
Acceleration of Axis X	25.0 mm/s <sup>2</sup>	Deceleration of Axis Y	25.0 mm/s <sup>2</sup>	Deceleration of Axis Z	25.0 mm/s <sup>2</sup>
Deceleration of Axis X	25.0 mm/s <sup>2</sup>	Jog Velocity of Axis Y	2.0 mm/s	Jog Velocity of Axis Z	2.0 mm/s
Jog Velocity of Axis X	2.0 mm/s	Jog Acceleration of Axis Y	25.0 mm/s <sup>2</sup>	Jog Acceleration of Axis Z	25.0 mm/s <sup>2</sup>
Jog Acceleration of Axis X	25.0 mm/s <sup>2</sup>	Jog Deceleration of Axis Y	5.0 mm/s <sup>2</sup>	Jog Deceleration of Axis Z	5.0 mm/s <sup>2</sup>
Jog Deceleration of Axis X	5.0 mm/s <sup>2</sup>				

Rysunek 22. Widok `contentAxisSettings` z domyślnymi wartościami.

## 12. System tekstów

W projekcie został zastosowany system tłumaczenia tekstów na język angielski oraz polski. Realizowane jest to z wykorzystaniem plików `.tmx` w których zapisywane są tłumaczone frazy wraz z unikalnymi ID w danym pliku.

Przystąpię do opisanie co znajduje się w poszczególnych plikach:

- **PLCOpenStateText** – znajdują się w nim nazwy stanów PLCOpen stosowany jest do wyświetlania na TextOutput

- **MainTranslate** – znajdują się w nim wszystkie teksty zlokalizowane na stronie głównej oraz na contentNavi
- **InfoTranslate** – Zawiera teksty znajdujące się na pasku Info
- **TrendsPosTranslate** - Zawiera teksty znajdujące się na contentTrends
- **AlarmTranslate** - Zawiera teksty znajdujące się na contentAlarm
- **SettingTranslate** - Zawiera teksty znajdujące się w oknie dialogowym ustawień domyślnych
- **HeaderTranslate** - Zawiera teksty znajdujące się na contentHeader
- **LoginTranslate** - Zawiera teksty znajdujące się w oknie dialogowym Login tłumaczone są w nim także napisy na przyciskach zdefiniowanych przez środowisko
- **TrendsVelTranslate** - Zawiera teksty znajdujące się na contentVelTrends
- **PopUpWinTranslate** - Zawiera teksty znajdujące się na wyskakujących oknach
- **AxisStateTranslate** - Zawiera tłumaczenie nazw stanów maszyny wypisywanych na TextOutput
- **AutoStateTranslate** - Zawiera tłumaczenie nazw stanów trybu automatycznego wypisywanych na TextOutput
- **LetterTranslate** - Zawiera nazwy liter wypisywanych na TextOutput, na podstawie enumeratora
- **AxisSettingsTranslate** - Zawiera teksty znajdujące się na contentAxisSettings
- **AlarmNamespaceTranslate** - Zawiera teksty alarmów, które monitorują wartość zmiennej. Jako jedyny plik ma namespace inny niż IAT, wynika to z faktu aby można było wyświetlić te teksty z wykorzystaniem snippetu

## 13. System użytkowników

W wizualizacji zastosowany został system użytkowników oraz ról. Ma on na celu ograniczenie dostępu dla pewnych użytkowników.

### Role

Zastosowane zostały 4 role:

- **Everyone** – Posiada ją użytkownik niezalogowany, ma on dostęp i możliwość zobaczenia w pełni tylko contentInfo oraz contentHeader. Nie może obsługiwać maszyny. Jedynie może ją wyłączyć awaryjnie.
- **Service** – jest to rola z większością opcji dostępu. Nie posiada ona jedynie dostępu do strony z parametrami osi.
- **Operator** – Rola pozwala na obsługę maszyny tylko w trybie manualnym oraz przeglądanie zakładki wykresów oraz alarmów.
- **MainServiceman** – Rola pozwala na pełny dostęp do maszyny, włącznie ze zmianą parametrów osi.





Everyone	Role ID	2	
	Description	Role for all.	
Service	Role ID	3	
	Description	Role for servi...	
Operator	Role ID	4	
	Description	Role for oper...	
MainServiceman	Role ID	5	
	Description	Can reconfigu...	

Rysunek 23. Stworzone role.

## Użytkownicy

W systemie zostały wyróżnieni czterej użytkownicy:

- Anonymous – osoba niezalogowana, posiada role Everyone, nie ma dostępu do obsługi maszyny.
- Serviceman1 – osoba z uprawnieniami Service, posiada on role Service, Operator oraz Everyone co pozwala mu na obsługę maszyny bez zmiany parametrów osi. **Hasło: Service**
- Manager – Posiada wszystkie możliwe uprawnienia. **Hasło: Manager**
- Operator1 – Osoba mająca tylko uprawnienia operatora, może obsługiwać maszynę tylko w trybie automatycznym. **Hasło: Operator**

	Anonymous		
	User ID	1	
	Password		
	Roles		
	Assigned Rol...	Everyone	
	Assigned Rol...		
	Serviceman1		
	User ID	2	
	Password	*****	Service
	Roles		
	Assigned Rol...	Everyone	
	Assigned Rol...	Service	
	Assigned Rol...	Operator	
	Assigned Rol...		
	Manager		
	User ID	3	
	Password	*****	Manager
	Roles		
	Assigned Rol...	MainService...	
	Assigned Rol...	Everyone	
	Assigned Rol...	Service	
	Assigned Rol...	Operator	
	Assigned Rol...		
	Operator1		
	User ID	4	
	Password	*****	Operator
	Roles		
	Assigned Rol...	Administrators	
	Assigned Rol...	Operator	
	Assigned Rol...		

Rysunek 24. Zdefiniowani użytkownicy oraz przypisane do nich role.