CS LAB s.c.
ElectronicLaboratory

Instrukcja Użytkownika

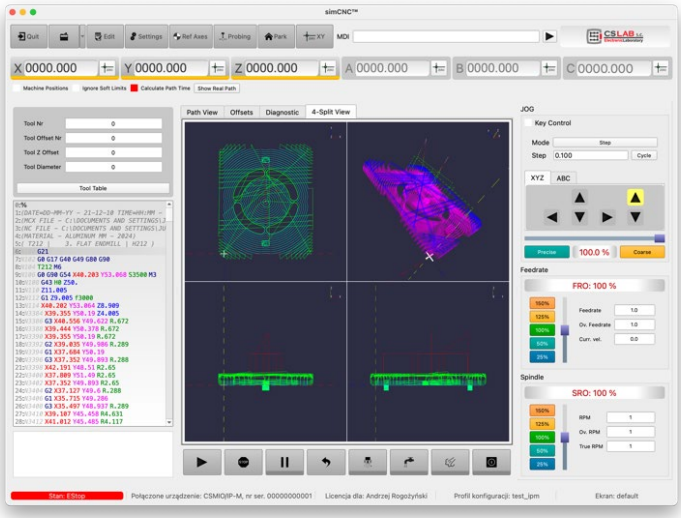
simCNC
oprogramowanie sterujące

Edytor interfejsu operatora

Spis treści

1. Informacje ogólne

Oprogramowanie simCNC wyposażono w nowoczesny edytor graficzny, który umożliwia tworzenie własnych, oryginalnych interfejsów operatorskich, precyzyjnie dostosowanych do wymagań klienta.
W połączeniu z językiem skryptowym Python i stylizowaniem z użyciem popularnego CSS, edytor umożliwia tworzenie funkcjonalnych, oraz atrakcyjnych wizualnie interfejsów.
Kod programu został zoptymalizowany pod kątem wydajności, by zapewnić responsywność i wygodę użytkowania. Zastosowano również system auto-rozmieszczenia i skalowania elementów graficznych, który sprawia, że stworzony interfejs jest bardziej dynamiczny i potrafi się dostosować w szerokim zakresie do różnych wielkości i rozdzielczości wyświetlaczy. Dzięki temu, że oprogramowanie simCNC jest wieloplatformowe, projekty ekranów mogą być używane bez modyfikacji na systemach Windows, macOS oraz Linux.



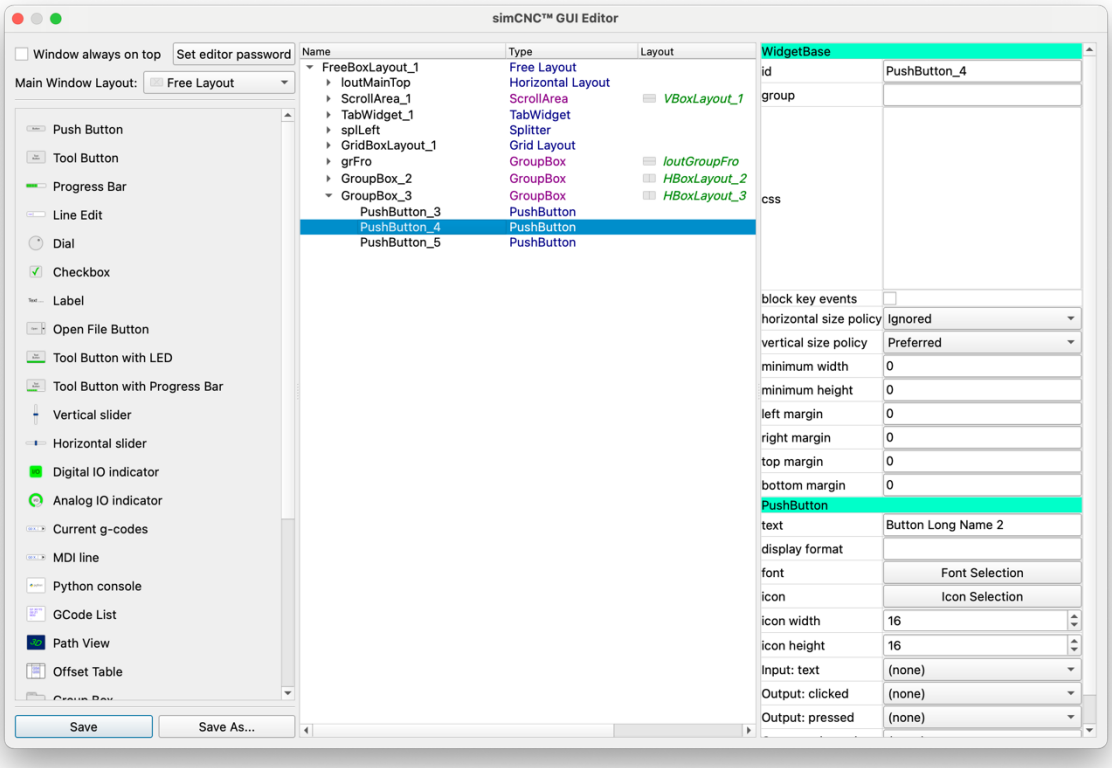
1.1. Zalecenia i wymagania systemowe

Oprogramowanie simCNC i zintegrowany edytor graficzny nie mają dużych wymagań sprzętowych.
Program zadziała nawet na RaspberryPI4 z 4GB RAM. Jednak do wygodnej pracy, szczególnie jeśli tworzymy interfejs „od zera”, dobrze jest wyposażyć się w dobry monitor o większej rozdzielczości np. 27’’ 2560x1440.
Bardzo wygodne jest posiadanie dwóch monitorów oraz komputera z większą ilością pamięci, by móc jednocześnie korzystać z narzędzi w rodzaju **Affinity Designer** czy **Photoshop** do przygotowania elementów graficznych, lub **Visual Studio Code** - do edycji arkuszy stylów (CSS) i skryptów Python.

2. Ogólne zasady pracy z edytorem

2.1. Okno edytora

Okno edytora otwieramy poprzez menu „Konfiguracja→Otwórz edytor interfejsu”.



Okno edytora podzielone jest na trzy główne, pionowe panele.
W lewym panelu kolejno (od góry) znajdują się następujące elementy:

- **Window always on top** – zaznaczenie powoduje, że okno edytora będzie zawsze na wierzchu
- **Set editor password** – zabezpieczenie edycji ekranu hasłem
- **Main Window Layout** – wybór sposobu rozmieszczenia kontenera głównego okna (patrz opis systemu auto-rozmieszczenia w dalszej części instrukcji)
- **Lista widżetów** – by umieścić dany widżet w projekcie klikamy myszą i przeciągamy go na projektowany ekran simCNC
- **Save** – zachowanie zmian
- **Save As** – zachowanie projektu pod inną nazwą

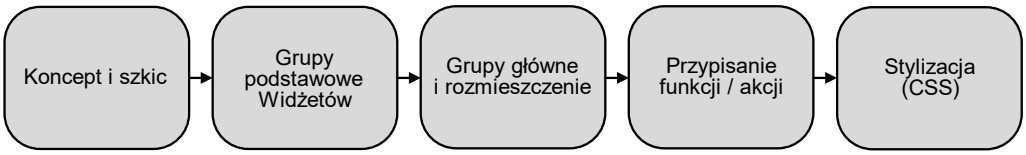
Panel środkowy pokazuje drzewo wszystkich elementów projektowanego interfejsu, natomiast panel prawy wyświetla listę właściwości zaznaczonego widżetu lub kontenera.
Właściwości są opisane w części poświęconej widżetom.

2.2. Skróty klawiszowe

Skrót	Opis
CTRL-C	Kopiuj zaznaczone elementy do schowka
CTRL-V	Wklej elementy ze schowka do zaznaczonego kontenera
CTRL-Z	Cofnij ostatnią operację (undo)
CTRL-Y	Ponów cofniętą operację (redo)
CTRL-S	Zapisz projekt

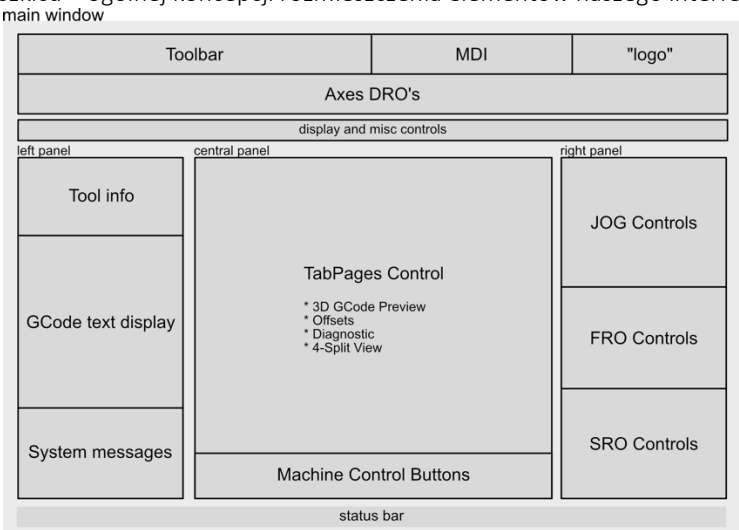
2.3. Etapy projektowania nowego interfejsu (workflow)

W tym rozdziale przedstawiono ogólny przebieg pracy przy projektowaniu nowego interfejsu. Nie należy przejmować się, jeśli część z użytych tutaj pojęć jest niejasna. Szczegółowy opis poszczególnych narzędzi znajduje się w kolejnych rozdziałach. Tutaj skupimy się na ogólnych zasadach.



2.3.1. Szkic koncepcyjny

By w pełni wykorzystać możliwości jakie oferuje system auto-rozmieszczenia i nie pogubić się w rosnącej ilości elementów, dobrze jest zacząć od przygotowania szkicu – ogólnej koncepcji rozmieszczenia elementów naszego interfejsu.



Na rysunku powyżej widać przykładowy szkic koncepcyjny domyślnego ekranu simCNC (**default**).

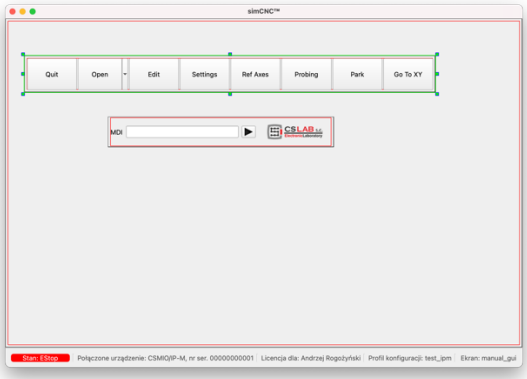
Część górna:

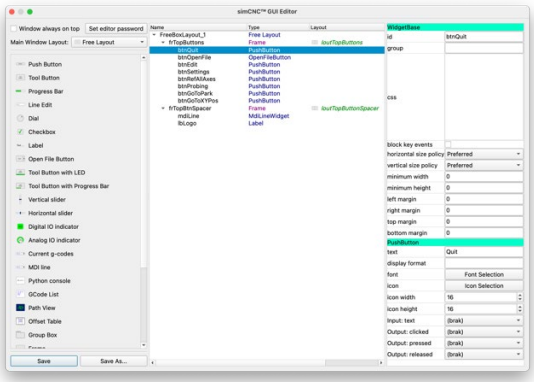
- **Toolbar** – pasek przycisków
- **MDI** – pole szybkiego wprowadzania komend maszynowych
- **„logo”** – logo firmowe
- **Axes DRO’s** – wskazania aktualnej pozycji poszczególnych osi obrabiarki
- **Display and misc controls** – kontrolki związane z wyświetlaniem koordynat i inne

Część dolna, podzielona na trzy sekcje:

- Sekcja lewa
  - **Tool Info** – informacje o aktualnym narzędziu i offsecie
  - **Gcode text display** – wyświetlanie zawartości załadowanego pliku Gcode
  - **System messages** – konsola komunikatów systemowych
- Sekcja centralna
  - **TabPages Control** – Panel zakładek
    - Podgląd 3D pliku Gcode
    - Offsety robocze
    - Diagnostyka
    - Widok 3D pliku Gcode w czterech rzutach
  - Pasek przycisków kontroli pracy
- Sekcja prawa
  - **JOG Controls** – panel kontrolek ręcznego sterowania osiami
  - **FRO Controls** – panel regulacji prędkości posuwu
  - **SRO Controls** – panel regulacji prędkości obrotowej wrzeciona

2.3.2. Przygotowanie podstawowych grup widżetów

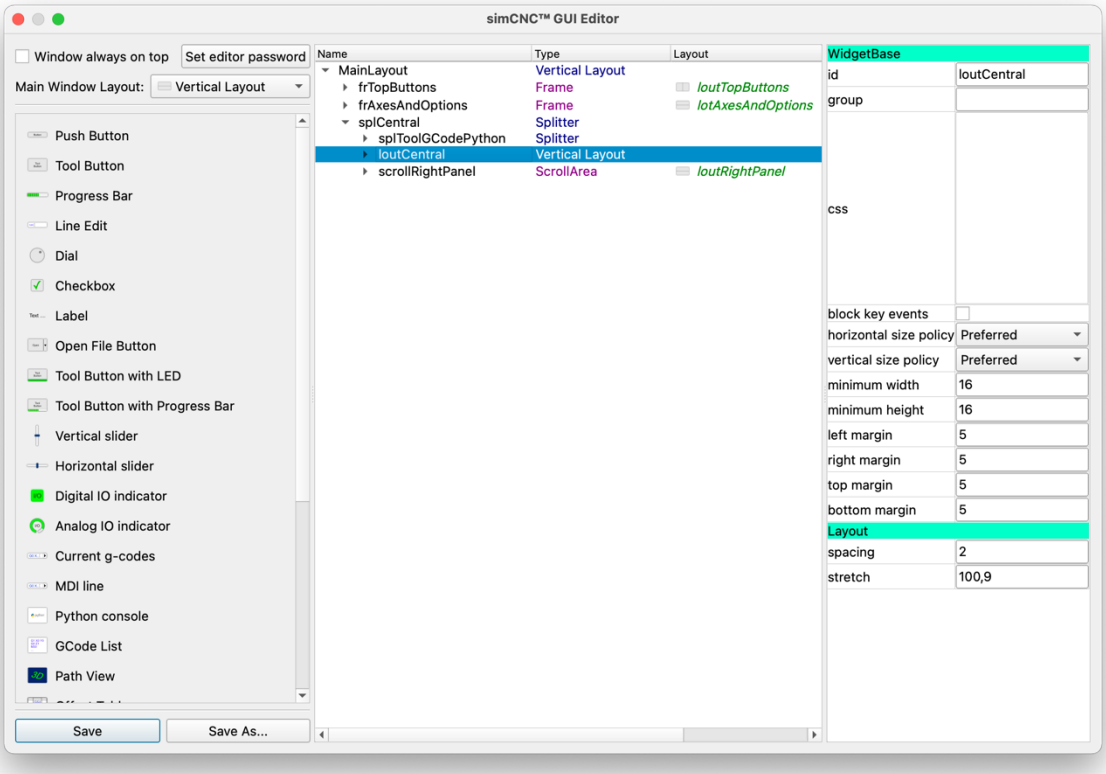




Na tym etapie możemy zacząć wstępnie tworzyć podstawowe grupy widżetów interfejsu. Na ilustracjach powyżej widzimy po lewej okno główne simCNC z grupą przycisków i MDI, po prawej - okno edytora.
Początkowo nie warto zajmować się wszystkimi szczegółami, takimi jak np. ikony dla przycisków, stylizacja itp. Na to przyjdzie czas później. Dobrze jest za to zadbać, by poszczególne kontrolki umieścić w grupach i wstępnie zdefiniować dla nich zasady rozmieszczenia. System auto-rozmieszczenia będzie szczegółowo opisany w osobnym rozdziale.
Ważne jest nadanie widżetom nazw, które pozwolą łatwo orientować się w projekcie.

2.3.3. Przygotowanie i rozmieszczenie głównych grup widżetów

Projekt interfejsu simCNC ma budowę hierarchiczną. Grupy podstawowe można łączyć w większe, a na końcu zdefiniować rozmieszczenie głównego okna programu. W hierarchii elementów projektu można się zorientować patrząc na środkowy panel okna edytora – drzewo obiektów. W przykładzie poniżej widać, że główny kontener **MainLayout** zawiera trzy elementy: **frTopButtons**, **frAxesAndOptions** oraz **splCentral**. W **splCentral** są z kolei zawarte: **splToolGCodePython**, **loutCentral** i **scrollRightPanel**.



Domyślny ekran simCNC posiada trzy główne grupy, rozmieszczone pionowo:

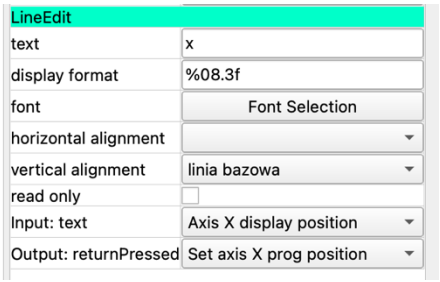
- **frTopButtons** – ramka z przyciskami, MDI oraz logo

- **frAxesAndOptions** – ramka z pozycjami osi i opcjami
- **splCentral** – grupa z lewym, centralnym i prawym panelem

Dobrze jest poświęcić nieco czasu na przemyślenie hierarchii i grup interfejsu na etapie szkicu. Ułatwia to późniejsze modyfikacje i definiowanie zasad skalowania.

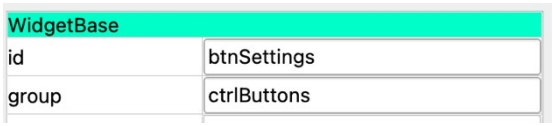
2.3.4. Przypisanie funkcji widżetom

Na tym etapie przypisujemy funkcje wejściowe i wyjściowe widżetom. Przykładowo dla widżetu wyświetlania pozycji osi definiujemy funkcję wejściową „**Axis … display position**" i wyjściową „**Set axis … prog position**".
Akcja wejściowa, aktualizuje wartość wyświetlaną, a wyjściowa wywołuje akcję ustawienia pozycji, gdy zawartość widżetu jest edytowana przez operatora. Wszystkie akcje wejściowe i wyjściowe są opisane w osobnym rozdziale. Dla przycisków jako akcję wyjściową można też ustawić **Run Script** i stworzyć makro Python, które będzie uruchomione, gdy operator kliknie przycisk. Umożliwia to realizację bardziej złożonych zadań lub akcji, których nie ma na liście standardowej.



2.3.5. Stylizacja arkuszami stylów (CSS)

System interfejsu programu simCNC posiada wbudowaną obsługę arkuszy stylów **CSS**. Stylizacja jest opcjonalna, ale zalecana, gdy chcemy stworzyć dynamiczny i atrakcyjny wizualnie interfejs. Komendy można wpisywać bezpośrednio w edytorze ekranu w polu **css,** albo (zalecane) stworzyć osobny plik z rozszerzeniem **.css** i umieścić go w katalogu ekranu. Bardzo wygodną rzeczą jest możliwość zdefiniowania grup widżetów do stylizacji. W tym celu, w trybie edycji wpisujemy w polu **group** właściwości widżetu nazwę grupy i możemy później w pliku **css** modyfikować atrybuty wizualne dla wszystkich widżetów, które mają ustawioną daną nazwę grupy. Na przykład w domyślnym ekranie wszystkie przyciski z górnego paska mają ustawioną nazwę grupy **ctrlButtons**.

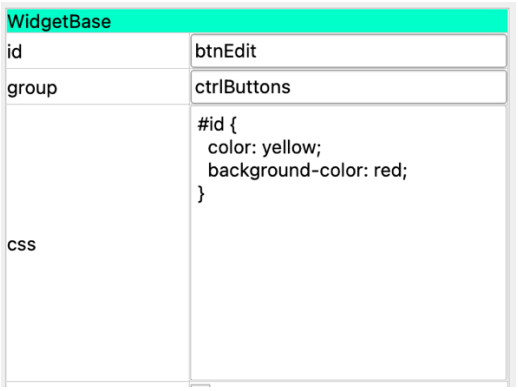


A tak wygląda fragment pliku **style.css**, który definiuje wygląd całej grupy:

```
[group="ctrlButtons"]{
  color: #404040;
  background-color: rgb(170, 170, 170);
  font-size: 12px;
}
[group="ctrlButtons"]:hover {
  color: rgb(112, 14, 14);
  background-color: lightgray;
  font-size: 13px;
}
[group="ctrlButtons"][darkTheme="true"]{
  color: #FFFFFF;
  background-color: rgb(60, 60, 60);
  font-size: 12px;
}
[group="ctrlButtons"][darkTheme="true"]:hover {
  color: rgb(182, 14, 14);
  background-color: lightgray;
  font-size: 13px;
}
```

Skrócony zapis „id" widżetu przy wprowadzaniu „css" w oknie edytora

Jeśli wprowadzamy zawartość **css'a** w oknie edytora, to możemy skorzystać z możliwości skróconego zapisu identyfikatora widżetu. W tym celu wpisujemy **#id**, tak jak poniżej.



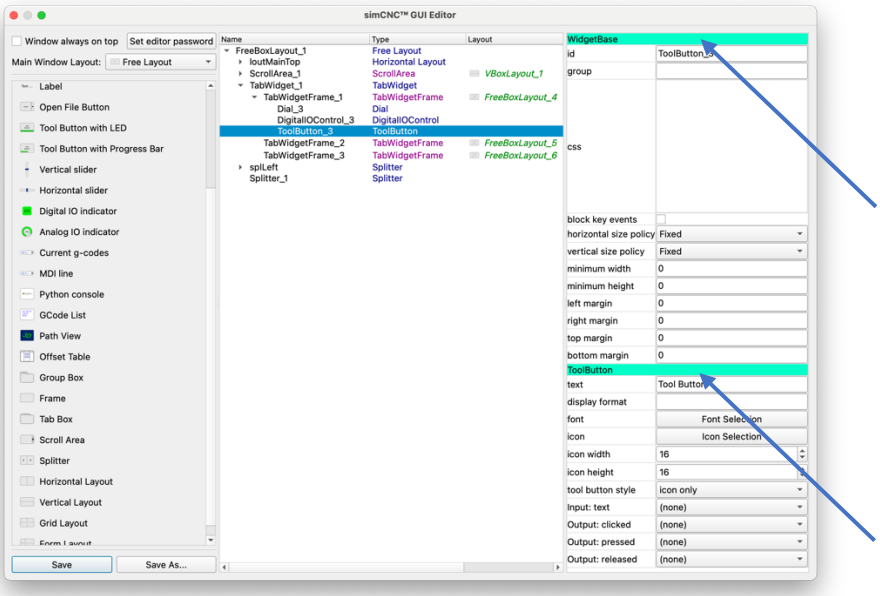
Normalnym, pełnym zapisem byłoby:

```
[id="btnEdit"] {
  ...
}
```

Oczywiście standardowy zapis jest również dozwolony.

3. Widżety

Wybór widżetu w trybie edycji wyświetla listę jego właściwości w prawym panelu okna edytora.



Na powyższym przykładzie widżetu **Tool Button** widać, że właściwości są pogrupowane. **WidgetBase** to grupa właściwości wspólnych dla wszystkich widżetów. **ToolButton** to grupa właściwości konkretnego widżetu **Tool Button**.

Poniższa tabela opisuje właściwości wspólne dla wszystkich widżetów.

Nazwa właściwości	Opis
Id	Identyfikator widżetu. Należy nadawać intuicyjne identyfikatory, szczególnie jeśli później używane są funkcje stylizacji css lub planujemy odwoływanie się do widżetów z poziomu makr Python. Na przykład: **btnStartGCode** jest o wiele lepszą nazwą niż **buton_123**.
Group	Identyfikator grupy widżetów. Bardzo przydatne do stylizacji (css). Nadając tą samą nazwę grupy wielu widżetom, można znacznie skrócić ilość kodu w arkuszu stylu css.
css	Pole szybkiego wpisywania komend do stylizacji widżetu. Przydatne szczególnie podczas eksperymentowania z różnymi właściwościami. Docelowo lepiej jest stworzyć plik (lub pliki) **.css** w katalogu tworzonego ekranu.
block key events	Zaznaczenie tej opcji powoduje, że widżet nie przekazuje dalej informacji o naciśniętych przyciskach na klawiaturze. Przydatne na przykład w polach edycji takich jak MDI. Dzięki temu zmiana pozycji kursora w polu edycji nie powoduje ruchu maszyny, gdy załączona jest kontrola JOG z klawiatury.
Horizontal size policy	Wytyczne poziomego skalowania widżetu. Dokładny opis w rozdziale poświęconym auto-rozmieszczeniu.
Vertical size policy	Wytyczne pionowego skalowania widżetu. Dokładny opis w rozdziale poświęconym auto-rozmieszczeniu.
Minimum width	Minimalna dozwolona szerokość widżetu.
Minimum height	Minimalna dozwolona wysokość widżetu.
Left margin	Lewy margines widżetu.
Right margin	Prawy margines widżetu.
Top margin	Górny margines widżetu.
Bottom margin	Dolny margines widżetu.

3.1. Push Button i Tool Button

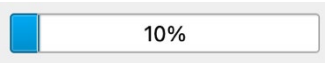
Oba te widżety przycisku mają prawie identyczną funkcjonalność. Różnią się nieco wizualnie. ToolButton posiada dodatkowo możliwość wyboru pozycji wyświetlanej ikony. Funkcjonalnie oba widżety są identyczne.

Właściwości:

Nazwa właściwości	Opis
Text	Wyświetlany na przycisku tekst
Display format	Ciąg tekstowy definiujący format wyświetlanych wartości. Zgodny ze standardem „printf”: https://en.wikipedia.org/wiki/Printf_format_string Np. **„%.3f”** powoduje wyświetlenie wartości zmiennoprzecinkowej z dokładnością do trzech miejsc po przecinku.
Font	Ustawienie czcionki dla widżetu
Icon	Wybór obrazka, który będzie użyty jako ikona przycisku
Icon width	Szerokość ikony
Icon height	Wysokość ikony
Tool buton style	Styl przycisku ToolButton. • **Icon only** – tylko ikona • **Text only** – tylko tekst • **Text beside icon** – tekst obok ikony • **Text under icon** – tekst pod ikoną • **Follow style** – tak jak zdefiniowano w arkuszu stylu
Input: text	Wejście widżetu – wybór parametru, który będzie aktualizował tekst wyświetlany na przycisku
Output: clicked	Wyjście widżetu – wybór akcji, która będzie wywołana po kliknięciu na przycisk
Output: pressed	Wyjście widżetu – wybór akcji, która będzie wywołana przy przyciśnięciu lewego klawisza myszy na przycisku
Output: released	Wyjście widżetu – wybór akcji, która będzie wywołana przy zwolnieniu lewego klawisza myszy na przycisku

3.2. Progress Bar

Pasek postępu. Może być używany do graficznej prezentacji różnych wartości, np. FRO, SRO, itp.



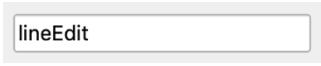
Właściwości:

Nazwa właściwości	Opis
Text	Wyświetlany na widżecie tekst
Display format	Ciąg tekstowy definiujący format wyświetlanych wartości. Zgodny ze standardem „printf”: https://en.wikipedia.org/wiki/Printf_format_string Np. **„%.3f”** powoduje wyświetlenie wartości zmiennoprzecinkowej z dokładnością do trzech miejsc po przecinku.
Font	Ustawienie czcionki dla widżetu
Horizontal alignment	Pozioma pozycja zawartości (tekstu) widżetu • **Left** – wyrównanie do lewej • **Right** – wyrównanie do prawej • **Center** – wyśrodkowanie • **Justify** – justowanie
Vertical alignment	Pionowa pozycja zawartości (tekstu) widżetu • **Top** – wyrównanie do góry • **Bottom** – wyrównanie do dołu • **Center** – wyśrodkowanie • **Baseline** – wyrównanie do linii bazowej
Read only	Zaznaczenie tej opcji oznacza, że zawartość widżetu jest tylko do odczytu i nie można jej edytować.

Input: value	Wejście widżetu – wybór parametru, który będzie aktualizował wartość paska postępu

3.3. Line Edit

Pole wyświetlania i edycji tekstu.



Właściwości:

Nazwa właściwości	Opis
Text	Wyświetlany tekst – zawartość widżetu
Display format	Ciąg tekstowy definiujący format wyświetlanych wartości. Zgodny ze standardem „printf”: https://en.wikipedia.org/wiki/Printf_format_string Np. **„%.3f**” powoduje wyświetlenie wartości zmiennoprzecinkowej z dokładnością do trzech miejsc po przecinku.
Font	Ustawienie czcionki dla widżetu
Horizontal alignment	Pozioma pozycja zawartości (tekstu) widżetu • **Left** – wyrównanie do lewej • **Right** – wyrównanie do prawej • **Center** – wyśrodkowanie • **Justify** – justowanie
Vertical alignment	Pionowa pozycja zawartości (tekstu) widżetu. • **Top** – wyrównanie do góry • **Bottom** – wyrównanie do dołu • **Center** – wyśrodkowanie • **Baseline** – wyrównanie do linii bazowej
Read only	Zaznaczenie tej opcji oznacza, że zawartość widżetu jest tylko do odczytu i nie można jej edytować.
Input: text	Wejście widżetu – wybór parametru, który będzie aktualizował tekst wyświetlany w polu edycji
Output: returnPressed	Wyjście widżetu – wybór akcji, która będzie wywołana przy wciśnięciu klawisza return, przy zakończeniu edycji zawartości widżetu

3.4. Dial

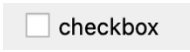
Pokrętło – zadawanie wartości liczbowych w określonym zakresie.

Właściwości:

Nazwa właściwości	Opis
Wrapping	Ustawienie zakresu obrotu na pełne 360° bez martwej strefy.
Notches visible	Włącza wyświetlanie podziałki
Notch target	Skok podziałki
Value	Aktualna wartość
Minimum	Wartość minimalna
Maximum	Wartość maksymalna
Single step	Skok regulacji zwykły (naciśnięcie strzałki w górę lub w dół)
Page step	Skok regulacji szybki (naciśnięcie page up lub page down)
Inverted appearance	Odwrócone wyświetlanie pokrętła. Zamiana minimum z maksimum.
Inverted controls	Odwrócenie kierunku regulacji
Input: value	Wejście widżetu – wybór parametru, który będzie aktualizował pozycję pokrętła
Output: valueChanged	Wyjście widżetu – wybór akcji, która będzie wywołana przy zmianie pozycji pokrętła przez użytkownika

3.5. Checkbox

Przycisk wyboru. Używany do załączania/wyłączania opcji, np. włączanie/wyłączanie limitów programowych.



Właściwości:

Nazwa właściwości	Opis
Text	Wyświetlany tekst
Checkbox state	Ustawia stan zaznaczenia
Display format	Ciąg tekstowy definiujący format wyświetlanych wartości. Zgodny ze standardem „printf”: https://en.wikipedia.org/wiki/Printf_format_string Np. **„%.3f”** powoduje wyświetlenie wartości zmiennoprzecinkowej z dokładnością do trzech miejsc po przecinku.
Font	Ustawienie czcionki dla widżetu
Input: checkbox state	Wejście widżetu – wybór parametru, który będzie aktualizował stan widżetu
Input: text	Wejście widżetu – wybór parametru, który będzie aktualizował tekst widżetu
Output: stateChanged	Wyjście widżetu – wybór akcji, która będzie wywołana przy zmianie stanu widżetu

3.6. Label

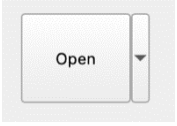
Widżet **Label** (Etykieta) służy do wyświetlania tekstu lub grafiki. Nie wywołuje żadnych akcji.

Właściwości:

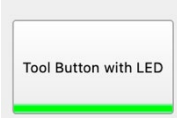
Nazwa właściwości	Opis
Text	Wyświetlany tekst
Display format	Ciąg tekstowy definiujący format wyświetlanych wartości. Zgodny ze standardem „printf”: https://en.wikipedia.org/wiki/Printf_format_string Np. „**%.3f**” powoduje wyświetlenie wartości zmiennoprzecinkowej z dokładnością do trzech miejsc po przecinku.
Font	Ustawienie czcionki dla widżetu
Word wrap	Włącza dzielenie wyrazów
Horizontal alignment	Pozioma pozycja zawartości (tekstu) widżetu • **Left** – wyrównanie do lewej • **Right** – wyrównanie do prawej • **Center** – wyśrodkowanie • **Justify** – justowanie
Vertical alignment	Pionowa pozycja zawartości (tekstu) widżetu • **Top** – wyrównanie do góry • **Bottom** – wyrównanie do dołu • **Center** – wyśrodkowanie • **Baseline** – wyrównanie do linii bazowej
Pixmap	Wybór mapy bitowej do wyświetlenia
Scale mode	Wybór typu skalowania bitmapy • **Normal** - bez zachowania proporcji • **KeepAspect** – z zachowaniem proporcji
Input: text	Wejście widżetu – wybór parametru, który będzie aktualizował tekst widżetu

3.7. Open File Button

Specjalny przycisk ładowania plików GCode. Podobny do **Tool Button**, ale dodatkowo wyświetla listę ostatnio otwieranych plików. Nie wymaga ustawiania akcji wejściowych i wyjściowych do działania. Właściwości są identyczne jak w **Tool Button**.



3.8. Tool Button with LED

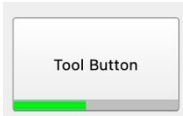


Wariant przycisku typu **Tool Button**. Dodatkowo umożliwia wyświetlanie kontrolki stanu, dla której można osobno zdefiniować akcję, która będzie aktualizować stan **LED**.

Obsługuje wszystkie właściwości Tool Button plus wymienione poniżej:

Nazwa właściwości	Opis
LED visible	Włącza wyłącza wyświetlanie kontrolki LED na przycisku
LED state	Stan LED – włączona / wyłączona
LED interval	Jeśli wartość jest większa od zera, włącza miganie LED. Wartość jest podawana w milisekundach i określa połowę okresu. Na przykład podanie 500 w tym polu oznacza *T = 500x2 = 1s ; f = 1/T = 1Hz*
Color	Wybór koloru kontrolki LED
Input: LED state	Wejście widżetu – wybór parametru, który będzie aktualizował stan wyświetlanej na przycisku diody LED
Input: LED interval	Wejście widżetu – wybór parametru, który będzie aktualizował interwał mrugania wyświetlanej diody LED

3.9. Tool Button with Progress Bar

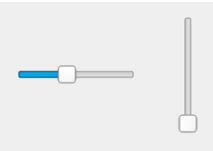


Wariant przycisku typu **Tool Button**. Dodatkowo umożliwia wyświetlanie paska postępu, dla którego można osobno zdefiniować akcję, która będzie aktualizować wartość.

Obsługuje wszystkie właściwości Tool Button plus dodatkowe poniżej:

Nazwa właściwości	Opis
Value	Aktualna wartość
Minimum	Wartość minimalna
Maximum	Wartość maksymalna
Color	Wybór koloru paska
Input: value	Wejście widżetu – wybór parametru, który będzie aktualizował wartość paska postępu

3.10. Horizontal Slider i Vertical Slider



Dwa warianty - poziomy i pionowy suwak do zadawania wartości w określonym zakresie.

Właściwości:

Nazwa właściwości	Opis
Value	Aktualna Wartość
Minimum	Wartość minimalna
Maximum	Wartość maksymalna
Single step	Skok regulacji zwykły (naciśnięcie strzałki w górę lub w dół)
Page step	Skok regulacji szybki (naciśnięcie page up lub page down)
Inverted appearance	Odwrócone wyświetlanie suwaka. Zamiana minimum z maksimum.
Inverted controls	Odwrócenie kierunku regulacji
Input: value	Wejście widżetu – wybór parametru, który będzie aktualizował pozycję suwaka
Output: valueChanged	Wyjście widżetu – wybór akcji, która będzie wywołana przy zmianie pozycji suwaka przez użytkownika

3.11. Digital IO indicator

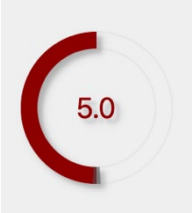
Kontrolka do wyświetlania stanu wartości logicznych (0, 1). Używana na przykład do sygnalizacji stanu sprzętowych wejść/wyjść. Może również wywoływać akcję przy kliknięciu.

Właściwości:

Nazwa właściwości	Opis
Text	Wyświetlany tekst
State	Ustawia stan kontrolki
Display format	Ciąg tekstowy definiujący format wyświetlanych wartości. Zgodny ze standardem „printf”: https://en.wikipedia.org/wiki/Printf_format_string Np. „%.3f” powoduje wyświetlenie wartości zmiennoprzecinkowej z dokładnością do trzech miejsc po przecinku.
Clickable	Zezwolenie sterowania stanem kontrolki poprzez klikanie myszą
Color	Wybór koloru kontrolki
Input: text	Wejście widżetu – wybór parametru, który będzie aktualizował tekst widżetu
Input: state	Wejście widżetu – wybór parametru, który będzie aktualizował stan widżetu
Output: stateChanged	Wyjście widżetu – wybór akcji, która będzie wywołana przy zmianie stanu widżetu przez użytkownika

3.12. Analog IO indicator

Kontrolka do wyświetlania wartości liczbowych. Używana na przykład do prezentacji wartości sprzętowych wejść/wyjść analogowych. Może również zadawać wartości.



Właściwości:

Nazwa właściwości	Opis
Text	Wyświetlany tekst
Display format	Ciąg tekstowy definiujący format wyświetlanych wartości. Zgodny ze standardem „printf”: https://en.wikipedia.org/wiki/Printf_format_string Np. „%.3f” powoduje wyświetlenie wartości zmiennoprzecinkowej z dokładnością do trzech miejsc po przecinku.
Value	Aktualna wartość
Clickable	Zezwolenie sterowania stanem kontrolki poprzez klikanie myszą
Color	Wybór koloru kontrolki
Maximum	Wartość maksymalna
Input: text	Wejście widżetu – wybór parametru, który będzie aktualizował tekst widżetu
Input: value	Wejście widżetu – wybór parametru, który będzie aktualizował wartość wskaźnika
Output: valueChanged	Wyjście widżetu – wybór akcji, która będzie wywołana przy zmianie wartości wskaźnika przez użytkownika

3.13. Current G-Codes

Lista aktualnego stanu komend modalnych (stan maszyny).

Właściwości:

Nazwa właściwości	Opis
Font	Ustawienie czcionki dla widżetu.
Horizontal alignment	Pozioma pozycja zawartości (tekstu) widżetu. • Left – wyrównanie do lewej • Right – wyrównanie do prawej • Center – wyśrodkowanie • Justify – justowanie
Vertical alignment	Pionowa pozycja zawartości (tekstu) widżetu.

	• Top – wyrównanie do góry • Bottom – wyrównanie do dołu • Center – wyśrodkowanie • Baseline – wyrównanie do linii bazowej

3.14. MDI Line

Pole szybkiego wprowadzania komend dla maszyny (MDI).

3.15. Python Console

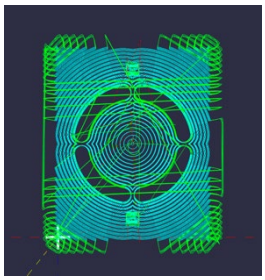
Panel tekstowy, w którym wyświetlane są komunikaty z makr Python oraz inne, systemowe komunikaty informacyjne i ostrzegawcze.

3.16. GCode List

Wyświetla zawartość aktualnie załadowanego pliku gcode w formie tekstowej oraz aktualnie wykonywanej linii podczas obróbki. Umożliwia również wybór linii, od której zacznie się obróbka – poprzez podwójne kliknięcie.

3.17. Path View

Wyświetla zawartość aktualnie załadowanego pliku gcode oraz aktualną pozycję osi maszyny w formie graficznej (3D).

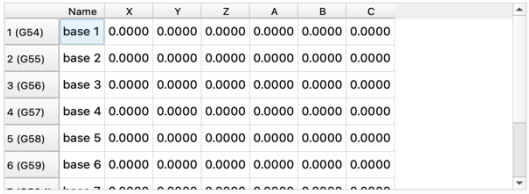


Właściwości:

Nazwa właściwości	Opis
Perspective view	Włączenie perspektywy dla widoku
Soft limits visible	Włącza wizualizację limitów programowych
Default view	Domyślny typ widoku: • **Top View** – widok z góry • **Bottom View** – widok z dołu • **Right View** – widok z prawej • **Left View** – widok z lewej • **Front View** – widok z przodu • **Rear View** – widok z tyłu • **Isometric view** – widok izometryczny
Default z height mapping	Domyślny typ mapowania wysokości (Z) kolorami: • **None** – bez kolorowania • **Color** – mapowanie kolorami • **Grayscale** – mapowanie odcieniami szarości

3.18. Offset Table

Widżet wyświetlający tabelę offsetów roboczych oraz umożliwiający ich edycję.



	Name	X	Y	Z	A	B	C
1 (G54)	base 1	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
2 (G55)	base 2	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
3 (G56)	base 3	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
4 (G57)	base 4	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
5 (G58)	base 5	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
6 (G59)	base 6	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000

3.19. Group Box

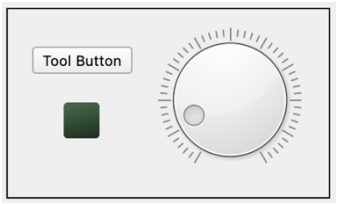
Jeden z podstawowych kontenerów służących do grupowania widżetów.

Właściwości:

Nazwa właściwości	Opis
Layout type	Typ auto-rozmieszczenia w grupie
Title	Wyświetlana nazwa grupy

3.20. Frame

Ramka - jeden z podstawowych kontenerów służących do grupowania widżetów. Ma nieco inny wygląd niż Group Box (nie wyświetla tytułu).

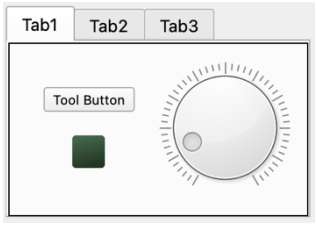


Właściwości:

Nazwa właściwości	Opis
Layout type	Typ auto-rozmieszczenia w ramce
Shadow	Typ cienia ramki • **Plain** – zwykły • **Raised** – podniesiony • **Sunken** – wpuszczony
Shape	Typ kształtu ramki • **No frame** – bez ramki • **Box** – prostokąt • **Panel** – podniesiony lub wpuszczony panel • **Styled panel** – panel, wg aktualnego stylu gui • **Horizontal line** – linia pozioma • **Vertical line** – linia pionowa • **Windows styled panel** – panel stylizowany na Windows 2000
Line width	Szerokość linii
Mid line width	Szerokość linii środkowej

3.21. Tab Box

Kontener z zakładkami, służący do grupowania widżetów.



Właściwości:

Nazwa właściwości	Opis
Tabs quantity	Ilość zakładek
Current tab	Aktualna (domyślna) zakładka

3.22. Scroll Area

Kontener z paskami przewijania. Przydatny, gdy mamy ograniczoną przestrzeń, a chcemy umieścić w interfejsie więcej elementów. Warto go również stosować, jeśli projekt interfejsu ma się poprawnie wyświetlać na ekranach o mniejszej rozdzielczości.

Właściwości:

Nazwa właściwości	Opis
Spacing	Odległość pomiędzy widżetami
Stretch	Współczynniki skalowania poszczególnych elementów w kontenerze
Layout type	Typ auto-rozmieszczenia • **Free Layout** – bez auto-rozmieszczenia • **Horizontal Layout** – rozmieszczenie poziome • **Vertical Layout** – rozmieszczenie pionowe • **Grid Layout** – rozmieszczenie w siatce • **Form Layout** – rozmieszczenie pionowe par
Horizontal scroll bar policy	Strategia wyświetlania poziomego paska przewijania zawartości • **As Needed** – tylko jeśli potrzebny • **Always Off** – zawsze wyłączony • **Always On** – zawsze widoczny
Vertical scroll bar policy	Strategia wyświetlania pionowego paska przewijania zawartości • **As Needed** – tylko jeśli potrzebny • **Always Off** – zawsze wyłączony • **Always On** – zawsze widoczny
Shadow	Typ cienia ramki • **Plain** – zwykły • **Raised** – podniesiony • **Sunken** – wpuszczony
Shape	Typ kształtu ramki • **No frame** – bez ramki • **Box** – prostokąt • **Panel** – podniesiony lub wpuszczony panel • **Styled panel** – panel, według aktualnego stylu • **Horizontal line** – linia pozioma • **Vertical line** – linia pionowa • **Windows styled panel** – panel stylizowany na Windows 2000
Line width	Szerokość linii
Mid line width	Szerokość linii środkowej

3.23. Horizontal Layout

Kontener systemu auto-rozmieszczania poziomego. Grupuje widżety i jednocześnie definiuje sposób ich rozmieszczenia i skalowania. System auto-rozmieszczania opisany jest w osobnym rozdziale.

Właściwości:

Nazwa właściwości	Opis
Spacing	Odległość pomiędzy widżetami
Stretch	Współczynniki skalowania poszczególnych elementów w kontenerze

3.24. Vertical Layout

Kontener systemu auto-rozmieszczania pionowego. Grupuje widżety i jednocześnie definiuje sposób ich rozmieszczenia i skalowania. System auto-rozmieszczania opisany jest w osobnym rozdziale.

Właściwości:

Nazwa właściwości	Opis
Spacing	Odległość pomiędzy widżetami
Stretch	Współczynniki skalowania poszczególnych elementów w kontenerze

3.25. Grid Layout

Kontener systemu auto-rozmieszczania w siatce. Grupuje widżety i jednocześnie definiuje sposób ich rozmieszczenia i skalowania. System auto-rozmieszczania opisany jest w osobnym rozdziale.

Właściwości:

Nazwa właściwości	Opis
Horizontal spacing	Odległość pozioma pomiędzy widżetami
Vertical spacing	Odległość pionowa pomiędzy widżetami
Column stretch	Współczynniki skalowania kolumn w kontenerze
Row stretch	Współczynniki skalowania rzędów w kontenerze

3.26. Form Layout

Kontener systemu auto-rozmieszczania w formularzu. Grupuje widżety i jednocześnie definiuje sposób ich rozmieszczenia i skalowania. System auto-rozmieszczania opisany jest w osobnym rozdziale.

Właściwości:

Nazwa właściwości	Opis
Horizontal spacing	Odległość pozioma pomiędzy widżetami
Vertical spacing	Odległość pionowa pomiędzy widżetami

3.27. Splitter

Kontener systemu auto-rozmieszczania z możliwością dynamicznego ustalania podziału i wielkości elementów. Grupuje widżety i jednocześnie definiuje sposób ich rozmieszczenia i skalowania. System auto-rozmieszczania opisany jest w osobnym rozdziale.

Właściwości:

Nazwa właściwości	Opis
Orientation	Sposób rozmieszczania elementów • **Horizontal** – poziomy • **Vertical** - pionowy
Shadow	Typ cienia ramki • **Plain** – zwykły • **Raised** – podniesiony • **Sunken** – wpuszczony
Shape	Typ kształtu ramki • **No frame** – bez ramki • **Box** – prostokąt • **Panel** – podniesiony lub wpuszczony panel • **Styled panel** – panel, według aktualnego stylu • **Horizontal line** – linia pozioma • **Vertical line** – linia pionowa • **Windows styled panel** – panel stylizowany na Windows 2000
Line width	Szerokość linii
Mid line width	Szerokość linii środkowej

4. System auto-rozmieszczania

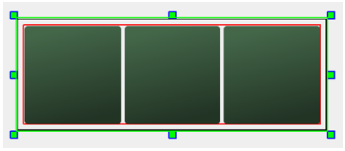
System auto-rozmieszczenia jest kluczowym elementem interfejsu użytkownika programu simCNC. Dzięki niemu stworzony interfejs jest wygodny, dynamiczny i potrafi w znacznym stopniu dostosować się automatycznie do różnych wielkości wyświetlaczy. Warto poświęcić nieco czasu na zapoznanie się z zasadami i przećwiczenie działania elementów tego systemu. Dzięki temu będzie możliwe szybkie i wygodne projektowanie atrakcyjnych i funkcjonalnych interfejsów.

4.1. Rodzaje kontenerów

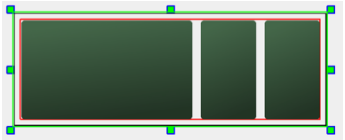
Podstawowym elementem systemu auto-rozmieszczenia jest tzw. kontener, w którym umieszczamy widżety i inne kontenery. Rodzaj kontenera ustala generalną zasadę rozmieszczenia elementów, które się w nim znajdują.

4.1.1. Horizontal Layout

Kontener z rozmieszczeniem poziomym. Jak łatwo się domyślić – służy do grupowania widżetów, układając je poziomo.



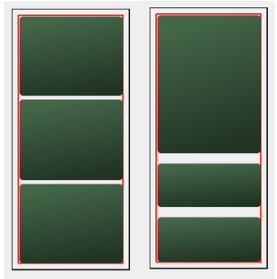
Na powyższym obrazku widać trzy widżety **Digital IO Indicator** ułożone poziomo w kontenerze.
Warto zwrócić uwagę na dwie właściwości kontenera, które kontrolują odległość między widżetami (**spacing**) oraz podział przestrzeni dla poszczególnych elementów (**stretch**).



Powyżej ten sam kontener, ale odległość (**spacing**) została zmieniona z „1" na „5" oraz podział przestrzeni (**stretch**) ustawiono na „3,1,1". Zapis „**3,1,1**" oznacza, że zalecana wielkość pierwszego widżetu będzie 3x większa od pozostałych. Proszę zwrócić uwagę na słowo **zalecana**. Na skalowanie elementów mają także wpływ parametry **size policy** opisane w osobnym podrozdziale. W powyższym przykładzie **size policy** ustawiono na **Preffered** dla wszystkich trzech elementów w kontenerze.

4.1.2. Vertical Layout

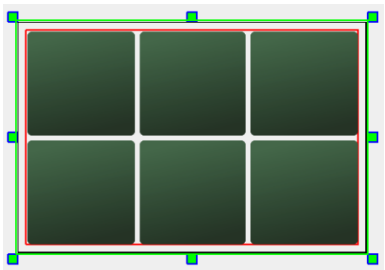
Kontener z rozmieszczeniem pionowym. Zasada działania tego kontenera jest identyczna jak **Horizontal Layout**, jedyną różnicą jest pionowe rozmieszczenie elementów, które się w nim znajdują.



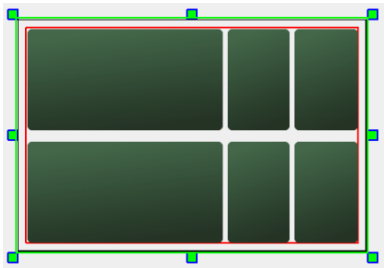
Powyżej widać dwa kontenery **Vertical Layout**, z takimi samymi widżetami i dwoma wariantami ustawień, tak jak w opisie kontenera **Horizontal Layout**.

4.1.3. Grid Layout

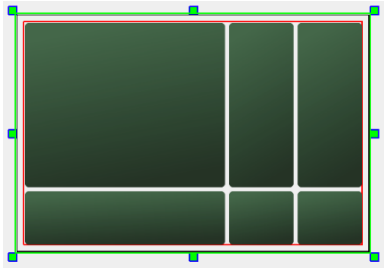
Kontener z rozmieszczeniem w siatce. W tym wypadku elementy są rozmieszczane zarówno poziomo jak i pionowo w formie wierszy i kolumn.



Na powyższym zrzucie widać sześć widżetów **Digital IO Indicator** umieszczonych w kontenerze **Grid Layout**. Jeśli chodzi o kontrolę nad odstępami pomiędzy elementami (**spacing**) oraz współczynniki skalowania (**stretch**), to zasada jest identyczna jak w poprzednio opisanych **Horizontal** i **Vertical Layout**, z tym, że mamy tutaj możliwość osobnego zdefiniowania tych właściwości dla kolumn i wierszy.



Kolejny przykład pokazuje zmianę **column spacing** na „1” i **column stretch** na „3,1,1”. Efektem tego, jak można się spodziewać, jest zmniejszenie odległości pomiędzy elementami w poziomie oraz pierwsza kolumna jest trzy razy szersza od pozostałych.

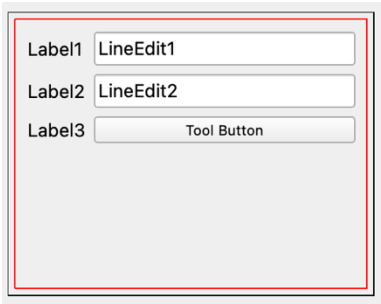


Tutaj identyczne ustawienia zastosowano dla **row spacing** (1) i **row stretch** (3,1,1). Teraz odległości w rzędach i kolumnach są takie same oraz pierwszy rząd jest trzy razy większy od pozostałych.

Często popełnianym błędem przez mniej doświadczonych użytkowników jest nadużywanie kontenera **Grid Layout** i próba umieszczenia w nim wszystkich elementów interfejsu. Takie podejście powoduje problemy, gdy liczba elementów rośnie. Gdy mamy sporo kolumn i rzędów, coraz trudniejsza staje się kontrola nad wielkością i rozmieszczeniem elementów. Dużo lepszym podejściem jest budowanie mniejszych bloków interfejsu i łączenie ich w większe. Pamiętajmy, że elementem kontenera może być również inny kontener.

4.1.4. Form Layout

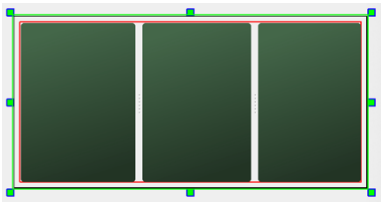
Kontener z rozmieszczeniem w formularzu. Jest to w pewnym sensie specjalna odmiana rozmieszczenia w siatce (**Grid Layout**), wyspecjalizowana do tworzenia formularzy, takich jak poniżej.



Form Layout rozmieszcza pary elementów w rzędach, czyli jest tak naprawdę siatką zbudowaną z dwóch kolumn i dowolnej liczby wierszy. Doskonale nadaje się w zastosowaniach takich jak w powyższym przykładzie, gdzie mamy powtarzające się wiersze typu „**opis→widżet**". Można w tym celu oczywiście użyć **Grid Layout**, ale Form Layout ma zoptymalizowane dla formularzy zasady skalowania elementów i często daje lepszy efekt wizualny przy mniejszym nakładzie pracy.

4.1.5. Splitter

Jest to kontener podobny do **Horizontal** i **Vertical Layout**, ale posiada istotną różnicę: O ile np. w **Horizontal Layout** definiujemy na stałe stosunek wielkości elementów, to **Splitter** pozwala na późniejszą modyfikację wielkości elementów przez operatora. **Splitter** pozwala wybrać typ rozmieszczenia (pionowy/poziomy) poprzez ustawienie właściwości „**Orientation**".



Powyżej widzimy trzy widżety „**Digital IO Indicator**" rozmieszczone poziomo w Spliterze.

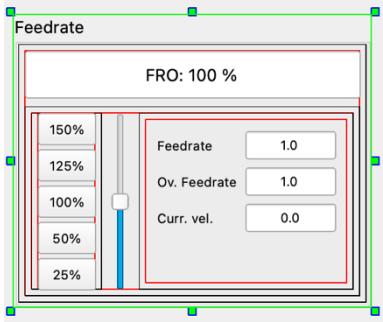
Powyżej ten sam kontener, ale po zamknięciu edytora i zmianie wielkości jego elementów poprzez kliknięcie na przestrzeń między widżetami i przesunięcie kursora myszy.

Dzięki temu kontenerowi można dać operatorowi więcej kontroli nad wielkością grup interfejsu, co często przekłada się na lepsze wrażenia i wygodniejszą pracę.

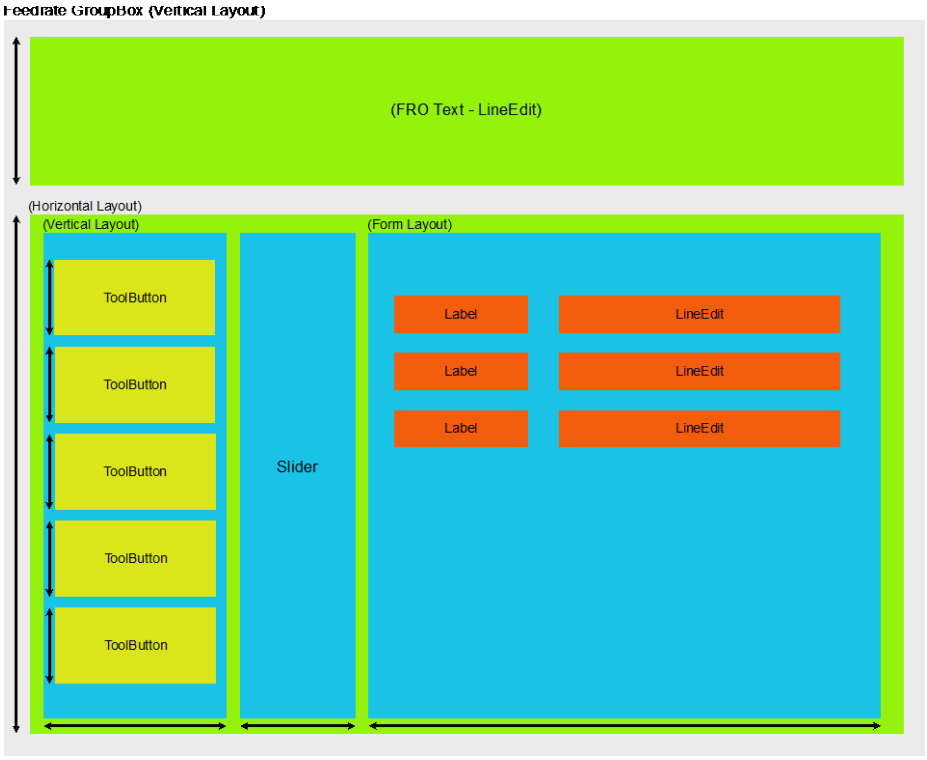
4.2. Łączenie kontenerów – budowa hierarchiczna

Łączenie ze sobą kontenerów jest podstawową drogą do budowy pełnoprawnego interfejsu oraz daje większą kontrolę nad wielkością elementów w systemie auto-rozmieszczania.

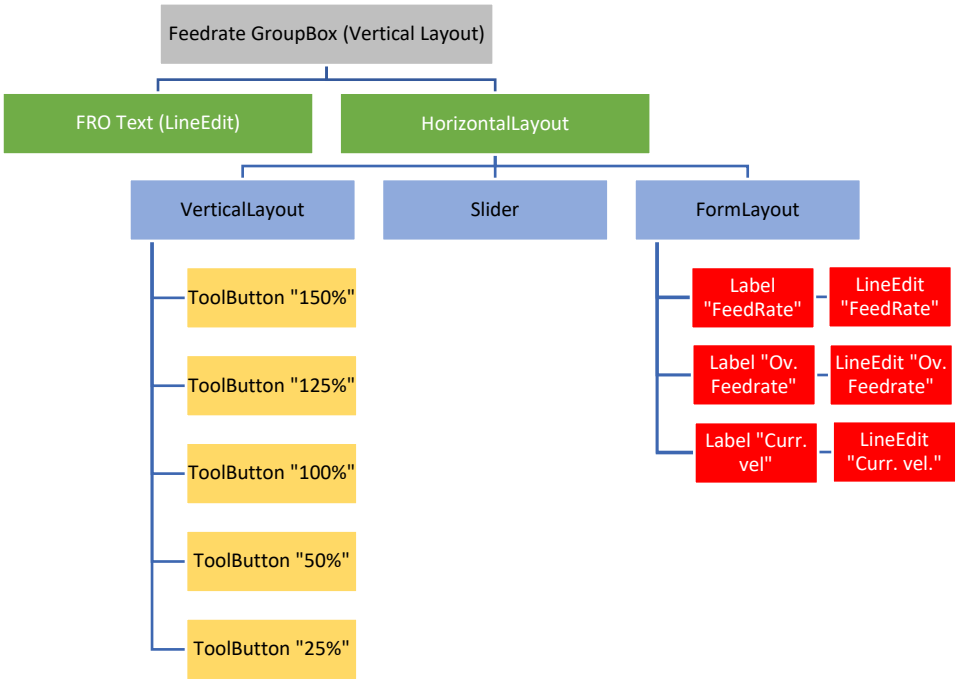
Weźmy na przykład część interfejsu taką jak poniżej:



Projektując tego typu blok ciężko w pierwszej chwili zdecydować jakiego typu rozmieszczenia użyć. Odpowiedzią jest skorzystanie z kilku kontenerów, tak jak pokazano na poniższym rysunku:



W formie drzewka wygląda to następująco:



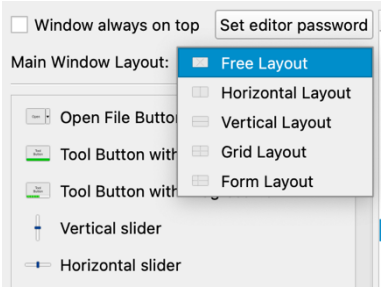
Głównym kontenerem jest widżet **GroupBox** z rozmieszczeniem pionowym (**Vertical Layout**). Znajdują się w nim dwa elementy zaznaczone kolorem zielonym: widżet **LineEdit** (tekst „FRO: 100%”) oraz kolejny kontener z rozmieszczeniem poziomym (**Horizontal Layout**). W nim z kolei znajdują się trzy elementy zaznaczone kolorem niebieskim: kontener z przyciskami 25%-100% (**Vertical Layout**), suwak ustawiania prędkości (**Slider**) oraz kontener z formularzem do wyświetlania i edycji parametrów (**Form Layout**).

W pierwszej chwili może się to wydawać nieco skomplikowane, ale taka hierarchiczna budowa ma wiele zalet. Między innymi wymusza porządek w projekcie oraz pozwala na łatwe wykorzystanie całych bloków w różnych miejscach lub nawet innych projektach. Jeśli na przykład chcemy wykorzystać grupę przycisków z powyższego przykładu w innym projekcie, zaznaczamy tylko odpowiedni kontener i klikamy CTRL-C. Zamykamy edytor, zmieniamy ekran, ponownie wchodzimy w tryb edycji i klikamy CTRL-V. Przy odrobinie wprawy myślenie o interfejsie w kategorii bloków i grup staje się naturalne i projektowanie zaczyna przebiegać wygodnie i szybko.

4.3. Rozmieszczenie elementów w głównym oknie

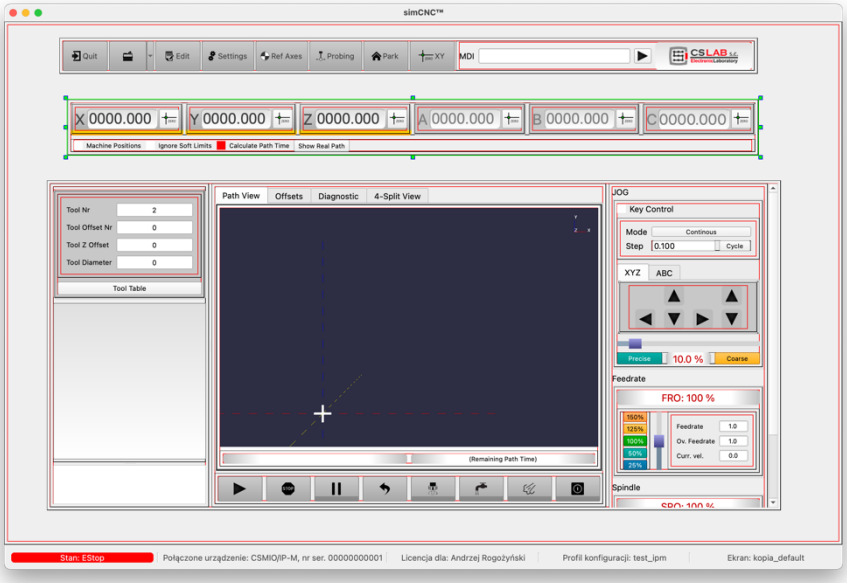
Główne okno programu simCNC jest również kontenerem, dla którego można wybrać typ rozmieszczenia elementów. Nawiązując do budowy hierarchicznej, omawianej w poprzednim podrozdziale – jest to kontener główny, na samym szczycie hierarchii.

Typ rozmieszczenia dla głównego okna wybieramy w oknie edytora:



O ile **Horizontal**, **Vertical**, **Grid** i **Form Layout** są już znane, to nie wspomniano jeszcze o “**Free Layout**”. „**Free Layout**” oznacza wyłączenie auto-rozmieszczenia. Wyłączenie auto-rozmieszczenia jest wygodne na wczesnym etapie projektowania elementów

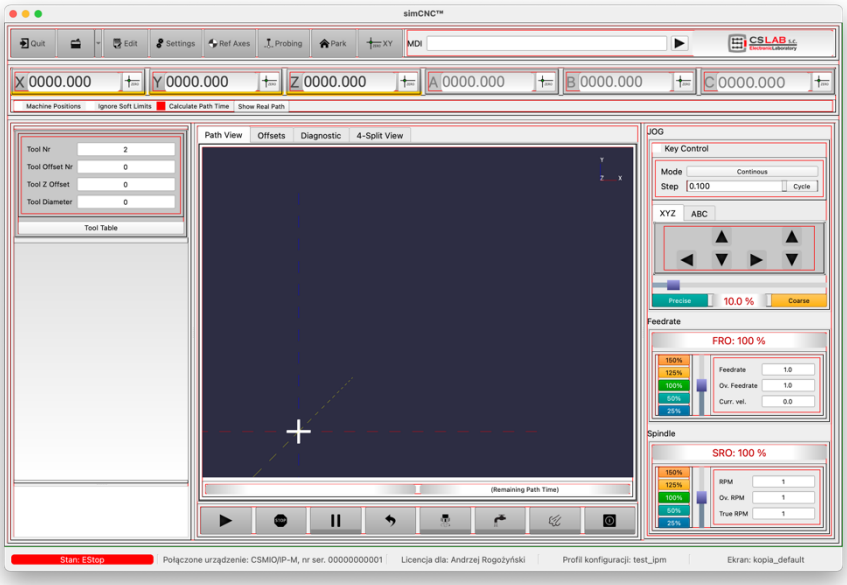
ekranu, lub gdy chcemy wprowadzić znaczące modyfikacje w projekcie. Wyboru typu rozmieszczenia dla okna dokonujemy w momencie, gdy mamy zaprojektowane główne grupy elementów.



Na powyższym zrzucie widać projekt domyślnego interfejsu simCNC z wyłączonym auto-rozmieszczeniem głównego kontenera. W tym projekcie mamy trzy główne grupy interfejsu:

- Grupę z podgrupami paska przycisków, MDI i logo
- Grupę z podgrupami widżetów pozycji osi i opcji
- Splitter z podgrupami z resztą elementów

Koncepcja projektu zakłada rozmieszczenie pionowe głównych grup. Poniżej widać zrzut po ustawieniu „**Vertical Layout**” dla głównego kontenera. Od tego momentu zmiana wielkości okna będzie powodowała automatyczne dopasowanie całej zawartości do nowego rozmiaru.



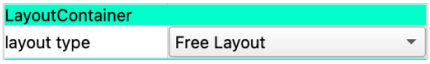
4.4. Widżety zawierające kontenery

Następujące widżety w edytorze interfejsu służą do grupowania elementów i zawierają w sobie kontenery z możliwością ustawienia typu rozmieszczenia:

- **GroupBox**
- **Frame**
- **TabBox**
- **ScrollArea**

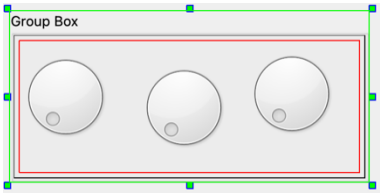
Właściwości tych widżetów i ich wygląd pokazano w rozdziale poświęconym widżetom.

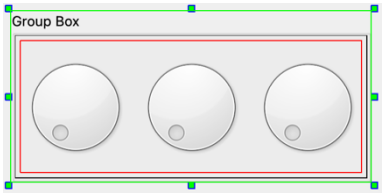
Podobnie jak w przypadku głównego kontenera, dla wyżej wymienionych widżetów można wyłączyć auto-rozmieszczanie poprzez wybór „**Free Layout**”.



Czasem bywa to przydatne podczas wstępnego projektowania zawartości widżetu.

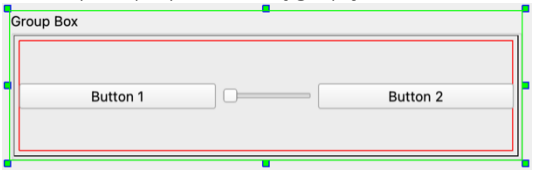
Poniżej pokazano przykład widżetu GroupBox z trzema widżetami Dial, przed i po załączeniu auto-rozmieszczenia poziomego (**Horizontal Layout**):



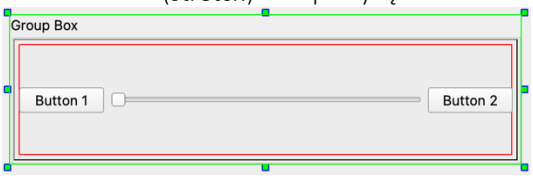


4.5. Podział przestrzeni w kontenerach

Bardzo często zachodzi potrzeba zdefiniowania w jaki sposób system auto-rozmieszczenia ma przydzielać dostępną przestrzeń w kontenerze poszczególnym elementom. Weźmy dla przykładu taką grupę:



Widzimy tutaj dwa przyciski i suwak w rozmieszczeniu poziomym. Pozornie wszystko wygląda poprawnie, ale operowanie suwakiem byłoby bardziej precyzyjne, gdyby był szerszy. Edytor ekranów simCNC pozwala dokładnie kontrolować ten aspekt poprzez ustawienia podziału przestrzeni w kontenerze (**stretch**) oraz politykę skalowania elementów (**size policy**).



Powyżej widzimy tą samą grupę, ale dla suwaka została ustawiona polityka skalowania w poziomie (**horizontal size policy**) na „**Expanding**” – czyli maksymalne wykorzystanie dostępnego miejsca.

4.5.1. Ustawienia zasad skalowania elementu (size policy)

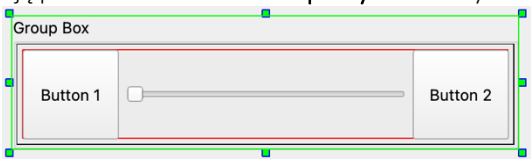
Dla każdego elementu można ustawić następujące zasady skalowania w pionie (**vertical size policy**) i poziomie (**horizontal size policy**):

Rodzaj	Opis
Fixed	Wielkość elementu ustawiona na sztywno za pomocą właściwości **minimum width** i **minimum height**. Ustawienie tych właściwości na zero powoduje, że używana jest domyślna, minimalna wielkość.
Minimum	Domyślna wielkość widżetu jest jego wielkością minimalną. Widżet może się powiększać, ale nie jest to wymuszane.

Maximum	Domyślna wielkość widżetu jest jego wielkością maksymalną. Widżet może być pomniejszony, ale nie poniżej wymiarów, które uniemożliwią jego używanie.
Preferred	Widżet może być powiększany i pomniejszany, ale nie poniżej wymiarów, które uniemożliwią jego używanie. Powiększanie elementu nie jest wymuszane.
Expanding	Wymuszenie zajmowania jak największej dostępnej przestrzeni dla elementu. Widżet może być też pomniejszany, ale nie poniżej wymiarów, które uniemożliwią jego używanie.
Minimum Expanding	Wymuszenie zajmowania jak największej dostępnej przestrzeni dla elementu.
Ignored	Dla elementu zostanie przydzielona potrzebna przestrzeń o ile jest to możliwe. Jeśli kontener jest zbyt mały, element może być pominięty i nie będzie w ogóle wyświetlony.

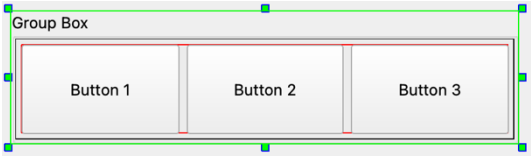
W praktyce najczęściej używane ustawienia to **Fixed**, **Preferred** oraz **Expanding**.

Poniżej dla przykładu – ta sama grupa co wcześniej, ale vertical size policy dla przycisków zmieniono z **Fixed** na **Preferred**. Zgodnie z oczekiwaniami wysokość przycisków dostosowała się by wykorzystać dostępną przestrzeń. W poziomie z kolei suwak zajmuje najwięcej miejsca, ponieważ przyciski mają parametr **horizontal size policy** ustawiony na **Preferred**, a suwak na **Expanding**.

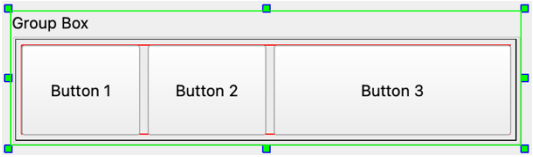


4.5.2. Stosunek wielkości elementów w kontenerze (stretch)

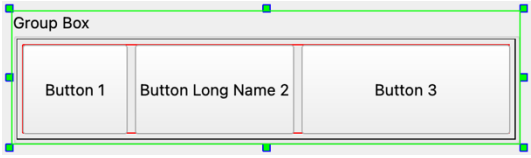
O tej właściwości kontenerów wspomniano już podczas omawiania rodzajów rozmieszczeń, ale dla przypomnienia – parametrem **stretch** można szybko i wygodnie kontrolować, jak przestrzeń ma być przydzielana dla elementów znajdujących się w kontenerze. By parametr ten działał poprawnie, najlepiej ustawić **size policy** dla elementów na **preffered** (**Fixed** np. zawsze wymusza rozmiar domyślny elementu).
Poniżej widać grupę trzech przycisków w rozmieszczeniu poziomym (**Horizontal Layout**). Parametry **size policy** przycisków ustawiono na **preffered**, a w polu **stretch** dla kontenera ustawiono „1,1,1”.



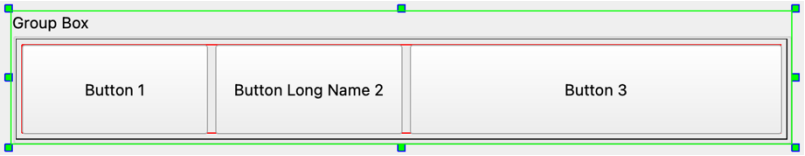
Wartość „1,1,1” oznacza, że zalecana wielkość każdego z trzech przycisków powinna być taka sama. Zauważmy co się stanie, gdy zmienimy parametr **stretch** kontenera na „1,1,2”:



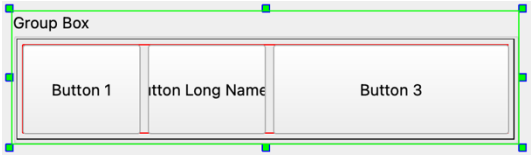
Jak widać, wielkość trzeciego przycisku jest teraz dwukrotnie większa. Warto wspomnieć, że w parametrze **stretch** znaczenie ma stosunek wartości, a nie wartości bezwzględne – czyli podając „10,10,20” efekt będzie identyczny. Użycie większych wartości bywa przydatne, gdy chcemy bardziej precyzyjnie zdefiniować podział. Wpisanie „10,10,15” spowoduje, że trzeci przycisk będzie 1,5x większy od pozostałych.
Kolejną sprawą jest to, że podane wielkości to wielkości zalecane. Proszę spojrzeć co stanie się w przypadku, gdy parametr **stretch** mamy ustawiony na „1,1,2”, ale opis drugiego przycisku zmienimy na znacznie dłuższy.



Jak widać, mimo zalecanego podziału, system auto-rozmieszczenia przydzielił więcej miejsca dla drugiego przycisku, aby zmieścić wyświetlany w nim tekst. Ustawiony podział będzie zastosowany dopiero wtedy, gdy wielkość kontenera będzie wystarczająca.



Innym wyjściem, jeśli zależy nam koniecznie na zachowaniu pożądanego podziału jest ustawienie **horizontal size policy** na **Ignored** dla drugiego przycisku.



Jak widać, zmiana zasad skalowania dla drugiego przycisku spowodowała, że ustawiony parametrem **stretch** podział został zachowany, ale kosztem tego, że tekst w przycisku został obcięty.

5. Połączenie interfejsu graficznego z programem simCNC

Widżety w swoich właściwościach mogą posiadać wejścia (**input**) oraz wyjścia (**output**). Za pomocą tych właściwości łączymy interfejs graficzny z funkcjami i wartościami z programu simCNC.

Wejścia widżetu pozwalają aktualizować jego stan lub zawartość, gdy dana wartość w simCNC ulega zmianie. Za przykład może posłużyć wyświetlanie aktualnej pozycji osi: tworzymy widżet **Label** i jako „**Input: text**" wybieramy z listy „**Axis X display position**". Gdy pozycja osi (w tym przypadku osi X) ulegnie zmianie, tekst widżetu zostanie zaktualizowany.

Wyjścia widżetu pozwalają by interfejs graficzny wywoływał akcje lub modyfikował parametry w programie simCNC. Przykładem może być tutaj przycisk startu obróbki: tworzymy widżet **ToolButton** i jako „**Output: Clicked**" w jego właściwościach wybieramy „**Start trajectory**".

5.1. Sygnały wejściowe widżetów

Nazwa	Opis
Anti-dive delay	Wartość opóźnienia funkcji Anti-dive przy automatycznej kontroli wysokości palnika plazmowego (THC)
Anti-dive velocity	Wartość prędkości funkcji Anti-dive przy automatycznej kontroli wysokości palnika plazmowego (THC)
Axis (…) abs position	Aktualna pozycja absolutna osi (koordynaty maszynowe)
Axis (…) current work offset	Aktualny offset roboczy osi
Axis (…) display position	Aktualna pozycja osi z możliwością przełączania pomiędzy koordynatami maszynowymi i programowymi
Axis (…) prog position	Aktualna pozycja programowa osi
Axis (…) tool offset	Offset narzędzia w danej osi
Axis (…) tool wear offset	Offset zużycia narzędzia w danej osi
Axis (…) velocity	Prędkość ruchu w danej osi
Calculated path time	Wyliczony prognozowany czas wykonania pliku gcode
Current spindle speed	Aktualne obroty wrzeciona
Current torch voltage	Aktualne napięcie łuku palnika plazmowego
Current velocity	Aktualna prędkość ruchu
Feedrate	Zadana prędkość posuwu
Feedrate override	Zadana prędkość posuwu z uwzględnieniem regulatora (FRO)
FRO	Wartość regulatora prędkości posuwu
GCode file path	Ścieżka aktualnie załadowanego pliku gcode
GCode line number	Numer wykonywanej linii pliku gcode
IO pin value	Wartość pinu sprzętowego we/wy
JOG mode	Aktualny tryb JOG
JOG speed	Zadana prędkość JOG
JOG step	Skok dla krokowego trybu JOG
Machine param	Wartość parametru maszynowego o zadanym numerze
Override spindle speed	Zadane obroty wrzeciona z uwzględnieniem regulatora (SRO)
Remain path time	Prognozowany czas pozostały do zakończenia wykonywania pliku gcode
Screen name	Nazwa aktualnie załadowanego ekranu
Selected tool nr	Numer wybranego narzędzia
Signal value	Wartość sygnału we/wy
Spindle CCW percent	Wartość procentowa aktualnych obrotów wrzeciona (obroty lewe)
Spindle CW percent	Wartość procentowa aktualnych obrotów wrzeciona (obroty prawe)
Spindle speed	Zadane obroty wrzeciona
Spindle tool nr	Numer narzędzia załadowanego we wrzecionie
SRO	Wartość regulatora prędkości obrotowej wrzeciona
THC init position	Zapamiętana współrzędna „Z", przy której nastąpiło załączenie funkcji automatycznej kontroli wysokości palnika
THC max deviation positive	Maksymalny zakres ruchu w kierunku **dodatnim** dla automatycznej kontroli wysokości palnika plazmowego

THC max deviation negative	Maksymalny zakres ruchu w kierunku **ujemnym** dla automatycznej kontroli wysokości palnika plazmowego
THC mode	Wybrany tryb pracy automatycznej kontroli wysokości palnika
THC position deviation	Aktualna wartość wyjściowa funkcji automatycznej kontroli wysokości palnika – odległość od THC init position
THC smart analog amplification	Wartość wzmocnienia dla trybu smart-analog funkcji automatycznej kontroli wysokości palnika
THC velocity	Prędkość ruchu dla funkcji automatycznej kontroli wysokości palnika
THC voltage deadband	Zakres napięcia braku regulacji dla automatycznej kontroli wysokości palnika
Tool diameter	Średnica aktualnie wybranego narzędzia
Tool diameter wear	Offset zużycia narzędzia (średnica)
Tool offset number	Numer wybranego offsetu narzędzia
Torch on mode	Tryb wykrywania łuku palnika plazmowego
Torch on voltage max	Górna granica napięcia dla funkcji wykrywania łuku palnika plazmowego
Torch on voltage min	Dolna granica napięcia dla funkcji wykrywania łuku palnika plazmowego
Torch voltage division factor	Dzielnik dla pomiaru napięcia łuku palnika plazmowego
Torch voltage potentiometer max	Wartość napięcia łuku dla pozycji maksymalnej potencjometru regulacji wysokości cięcia
Torch voltage potentiometer min	Wartość napięcia łuku dla pozycji minimalnej potencjometru regulacji wysokości cięcia
Torch voltage threshold	Wartość progowa napięcia łuku dla ruchu góra/dół dla trybów „analog" i „smart-analog" funkcji automatycznej kontroli wysokości palnika
Work offset number	Numer aktualnie wybranego offsetu roboczego

5.2. Sygnały wyjściowe widżetów

Nazwa	Opis
Close simCNC	Zamknięcie programu simCNC
Edit G-Code	Otworzenie okna edycji pliku gcode
Execute probing script	Uruchomienie makra python do pomiaru narzędzia
JOG (...)+ pressed	Start ruchu JOG w kierunku dodatnim dla osi
JOG (...)+ released	Zatrzymanie ruchu JOG w kierunku dodatnim dla osi
JOG (...)- pressed	Start ruchu JOG w kierunku ujemnym dla osi
JOG (...)- released	Zatrzymanie ruchu JOG w kierunku ujemnym dla osi
Open settings window	Otworzenie okna ustawień programu simCNC
Path simulation	Wykonanie symulacji pliku gcode, analiza prędkości, przyspieszeń itp.
Ref (...) axis	Uruchomienie procedury bazowania osi
Ref all axes	Uruchomienie sekwencji bazowania osi wybranych do automatycznego bazowania w ustawieniach simCNC
Rewind trajectory	Przewinięcie pliku gcode do początku
Run script	Uruchamia wskazane makro python
Run spindle clockwise	Załączenie prawych obrotów wrzeciona
Run spindle counter-clockwise	Załączenie lewych obrotów wrzeciona
Set anti-dive delay	Ustawienie wartości opóźnienia anti-dive funkcji automatycznej kontroli wysokości palnika plazmowego
Set anti-dive velocity	Ustawienie prędkości anti-dive funkcji automatycznej wysokości palnika
Set axis (...) current work offset	Ustawienie wartości offsetu roboczego w danej osi
Set axis (...) prog position	Modyfikacja wartości offsetu roboczego w danej osi poprzez podanie aktualnej wartości pozycji programowej
Set current tool diameter	Ustawienie wartości średnicy aktualnie wybranego narzędzia
Set current tool diameter wear	Ustawienie wartości zużycia dla średnicy aktualnie wybranego narzędzia
Set feedrate	Ustawienie wartości zadanej posuwu
Set flood on/off	Załącza/wyłącza chłodziwo
Set FRO	Ustawia aktualną wartość regulatora prędkości posuwu
Set IO pin value	Ustawienie stanu pinu (wyjściowego)
Set JOG mode	Ustawienie trybu pracy funkcji JOG

Set JOG speed	Ustawienie prędkości ruchu JOG (0 – 100%)
Set JOG step	Ustawienie skoku dla trybu krokowego ruchu JOG
Set machine param	Ustawienie wartości parametru maszynowego
Set mist on/off	Załącza/wyłącza chłodzenie mgłą
Set pause on/off	Zatrzymuje/wznawia wykonywanie pliku gcode
Set selected tool number	Ustawia numer aktualnie wybranego narzędzia
Set spindle speed	Ustawia zadane obroty wrzeciona
Set spindle tool number	Ustawia numer narzędzia załadowanego we wrzecionie
Set SRO	Ustawia aktualną wartość regulatora obrotów wrzeciona
Set THC max deviation negative	Ustawia maksymalny zakres ruchu w kierunku ujemnym dla funkcji automatycznej kontroli wysokości palnika
Set THC max deviation positive	Ustawia maksymalny zakres ruchu w kierunku dodatnim dla funkcji automatycznej kontroli wysokości palnika
Set THC off	Wyłącza funkcję automatycznej kontroli wysokości palnika
Set THC on	Włącza funkcję automatycznej kontroli wysokości palnika
Set THC smart-analog amplification	Ustawia wzmocnienie dla trybu “smart -analog”, funkcji automatycznej kontroli wysokości palnika
Set THC velocity	Ustawia prędkość ruchu dla funkcji automatycznej kontroli wysokości palnika
Set THC voltage deadband	Ustawia dozwolony zakres wahań napięcia łuku palnika, dla którego nie jest wykonywana korekta wysokości
Set tool number offset	Ustawia numer offsetu roboczego
Set torch on voltage max	Ustawia górną granicę napięcia łuku dla funkcji wykrywania łuku palnika plazmowego
Set torch on voltage min	Ustawia dolną granice napięcia łuku dla funkcji wykrywania łuku palnika plazmowego
Set torch voltage division factor	Ustawia wartość dzielnika pomiaru napięcia łuku palnika plazmowego
Set torch voltage potentiometer max	Ustawia zadaną wartość napięcia łuku dla maksymalnej pozycji potencjometru
Set torch voltage potentiometer min	Ustawia zadaną wartość napięcia łuku dla minimalnej pozycji potencjometru
Set torch voltage threshold	Ustawienie wartości progowej dla ruchu góra/dół palnika (tryb „analog” i „smart-analog” funkcji automatycznej kontroli wysokości palnika)
Set work offset number	Ustawia numer offsetu roboczego
Show real trajectory	Włącza widok rzeczywistej trajektorii ruchu, z uwzględnieniem optymalizacji i stałej prędkości skrawania.
Start trajectory	Uruchomienie obróbki – start aktualnie załadowanego pliku gcode
Stop trajectory	Zatrzymanie obróbki
Switch to EStop/Idle state	Przełączanie programu simCNC pomiędzy stanami zatrzymania i gotowości
Tool table	Wyświetla okno listy narzędzi

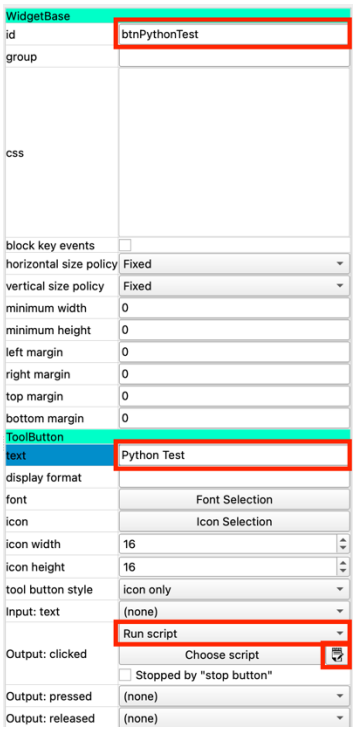
6. Połączenie interfejsu graficznego ze skryptami Python

System interfejsu graficznego simCNC umożliwia:

- wykonywanie skryptów użytkownika po kliknięciu przycisku
- odwołanie się do widżetów z poziomu skryptu
    - uruchamianie akcji widżetu
    - odczyt i modyfikację właściwości widżetu (np. tekstu)

6.1. Wywołanie skryptu po kliknięciu przycisku na ekranie

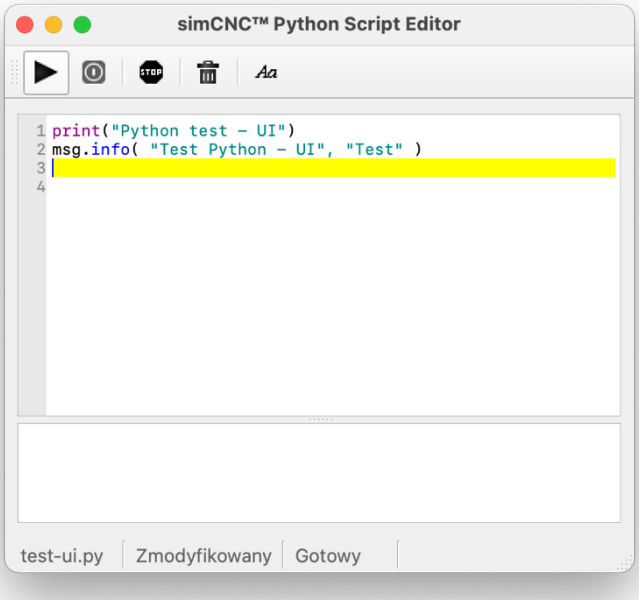
W poprzednim rozdziale przedstawiono listę możliwych sygnałów wyjściowych widżetów. Często zachodzi jednak potrzeba zaimplementowania funkcji niestandardowej, wywoływanej przez operatora z interfejsu graficznego. Poniżej pokazano przykład w jaki sposób można to zrealizować:



Tworzymy w edytorze interfejsu przycisk (**ToolButton**). Nadajemy mu identyfikator – np. **btnPythonTest**, można również ustawić etykietę przycisku, np. „Python Test”.

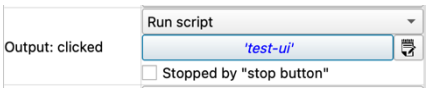
Następnie dla właściwości **Output: clicked** przycisku ustawiamy **Run script**. Otworzy się okno wyboru skryptu. Zamykamy je, bo dopiero za moment stworzymy nowy plik.

Otwieramy edytor skryptu (ikona obok przycisku **Choose script**).



Wpisujemy w edytorze kod skryptu (jak powyżej). Nasz przykładowy skrypt wyświetla komunikat w konsoli oraz okno informacyjne.

Nagrywamy plik, najlepiej w katalogu z danymi naszego projektowanego ekranu. W naszym przykładzie ekran ma nazwę „my_test_screen”. Przechodzimy więc do podkatalogu „screens/my_test_screen/scripts” względem miejsca instalacji simCNC i nadajemy nazwę pliku – np. „test-ui”.



Następnie możemy zamknąć okno edytora Python i klikamy przycisk **Choose script** we właściwościach widżetu oraz wybieramy plik „test-ui”. Nagrywamy zmiany w projekcie klikając **Save** oraz zamykamy edytor interfejsu i gotowe – stworzyliśmy przycisk, po kliknięciu którego zostanie uruchomione nasze makro Python. Można sprawdzić jego działanie klikając na przycisk. W konsoli python (o ile mamy ten widżet w projekcie) powinien pojawić się tekst „Python test – UI” oraz dodatkowo powinno pojawić się okno z komunikatem „Test Python – UI”.

6.2. Odwołanie się do elementów interfejsu z poziomu skryptu Python

System interfejsu graficznego simCNC umożliwia odwoływanie się do widżetów z poziomu skryptów Python. Kiedy zachodzi taka potrzeba? Wyobraźmy sobie sytuację, że mamy obrabiarkę z automatyczną wymianą narzędzi i chcemy mieć wyświetlany na

ekranie status, lub kod błędu ze zmieniarki narzędzi. Wymiana narzędzia najczęściej realizowana jest poprzez makro maszynowe **M6**. Kod makra **M6** musi więc mieć możliwość zaktualizowania tekstu widżetu na ekranie, by wyświetlić pożądane informacje.

Tworząc kod makra we wbudowanym w simCNC edytorze importowana jest automatycznie klasa o nazwie **gui**, która posiada w sobie wszystkie elementy graficzne załadowanego ekranu. Do konkretnego elementu odwołujemy się poprzez identyfikator widżetu (jego właściwość **id**):

gui.<id>.<nazwa metody>(<argumenty>)

Korzystając z przykładu widżetu z poprzedniego podrozdziału, chcąc zmienić tekst na przycisku robimy to następujący sposób:

gui.btnPythonTest.setText("Some New Text")

6.2.1. Metody klasy widżetu

Klasa widżetu udostępnia programiście następujące metody:

Nazwa metody	Opis
executeClickedOutput	Uruchamia akcję zdefiniowaną na kliknięcie lewym przyciskiem myszy Argumenty: (brak) Wartość zwracana: (brak)
executePressedOutput	Uruchamia akcję zdefiniowaną na wciśnięcie lewego klawisza myszy na widżecie Argumenty: (brak) Wartość zwracana: (brak)
executeReleasedOutput	Uruchamia akcję zdefiniowaną na zwolnienie lewego klawisza myszy na widżecie Argumenty: (brak) Wartość zwracana: (brak)
executeOutput	Uruchamia akcję zdefiniowaną na określone wyjście widżetu. Argumenty: • Nazwa wyjścia widżetu • Wartość zwracana: (brak)
getAttribute	Pobranie wartości właściwości widżetu. Argumenty: • Nazwa właściwości Wartość zwracana: • Wartość właściwości
getAttributes	Pobranie listy właściwości widżetu Argumenty: (brak) Wartość zwracana: • Lista nazw właściwości widżetu
getOutputs	Pobranie listy sygnałów wyjściowych widżetu Argumenty: (brak) Wartość zwracana: • Lista nazw sygnałów wyjściowych widżetu
getText	Pobranie właściwości **text** widżetu. Argumenty: (brak) Wartość zwracana: • Zawartość właściwości **text** widżetu
setAttribute	Ustawienie właściwości widżetu Argumenty: • Nazwa właściwości

	• Wartość właściwości Wartość zwracana: (brak)
setText	Ustawienie właściwości **text** widżetu Argumenty: • Wartość dla właściwości **text** widżetu Wartość zwracana: (brak)

6.2.2. Zmiana stylizacji widżetu

Korzystając z metody **setAttribute** możemy dynamicznie ustawić arkusz stylu (**css**) dla widżetu. Nazwa właściwości to **styleSheet**. W poniższych przykładach jako identyfikator widżetu użyto **btnPythonTest** z podrozdziału 6.1.

Przykład 1 (zmiana koloru tła na czerwony):

gui.btnPythonTest.setAttribute(„styleSheet”, „background-color: red;”)

Przykład 2 (zmiana koloru tła na czerwony i koloru czcionki na żółty):

gui.btnPythonTest.setAttribute(„styleSheet”, „background-color: red; color: yellow;”)

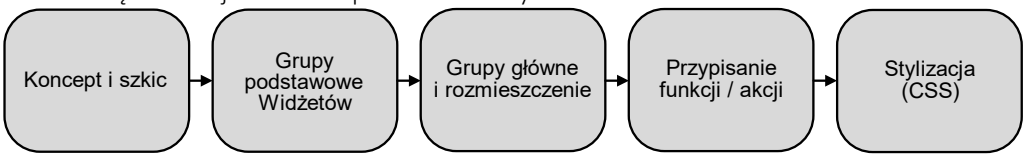
Przykład 3 (zmiana wielkości czcionki):

gui.btnPythonTest.setAttribute(„styleSheet”, „font-size: 24px;”)

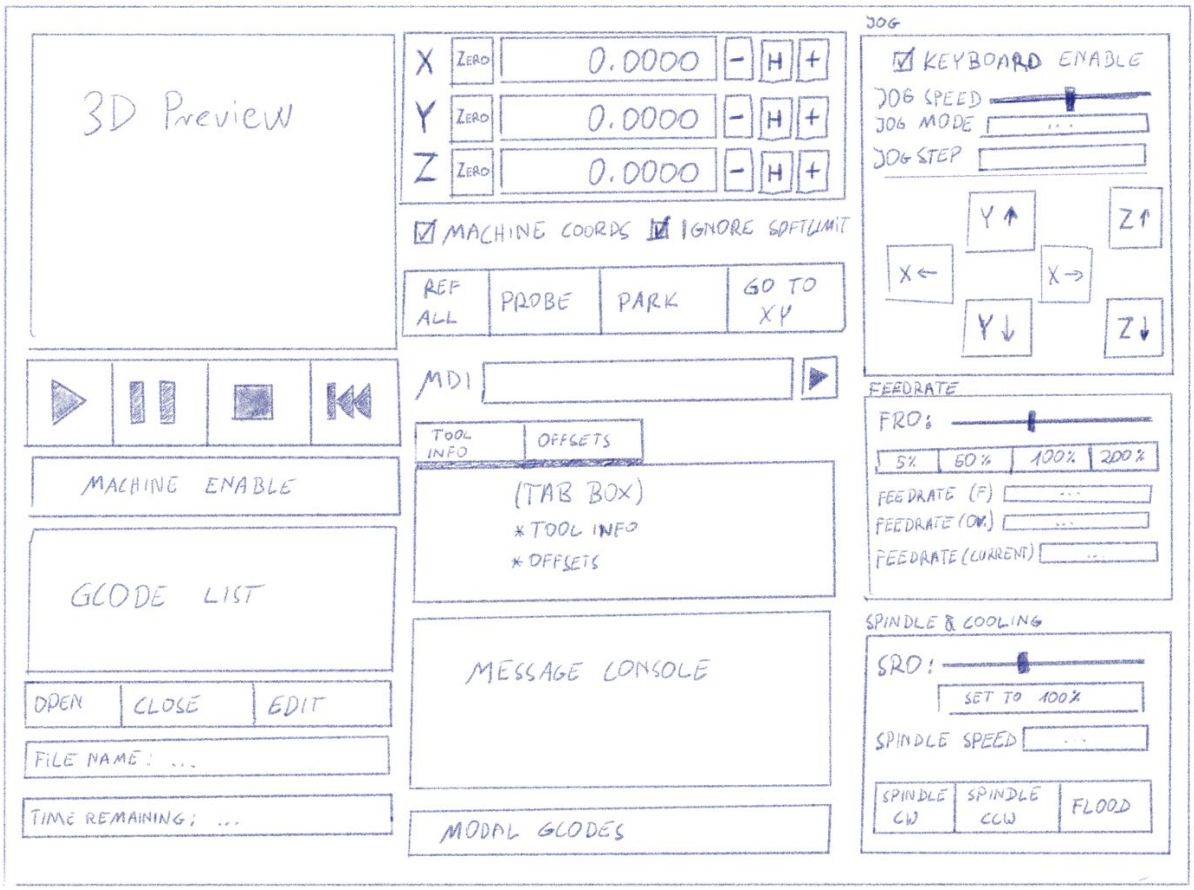
Dodatek – projekt interfejsu krok po kroku

W niniejszym dodatku przedstawiono budowę kompletnego interfejsu, krok po kroku – korzystając z informacji zawartych w poprzednich rozdziałach.

Dla przypomnienia – taka będzie kolejność zadań podczas budowy:



Koncept i szkic



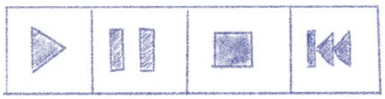
Zakładamy budowę kompaktowego interfejsu dla trzy osiowej frezarki, według powyższego szkicu. Jak widać, wystarczy nawet odręczny rysunek. Jeśli przeznaczenie niektórych elementów nie jest jasne – spokojnie, wszystko będzie po kolei wytłumaczone w dalszej części.

Stworzenie nowego projektu interfejsu i rozpoczęcie edycji

- Wybieramy pozycję z menu: **Konfiguracja→ustaw ekran**
- W oknie wyboru ekranu klikamy przycisk **Utwórz nowy**
- Nadajemy nazwę, na przykład „ui_example”
- Wybieramy nazwę nowo utworzonego interfejsu na liście i klikamy przycisk **Ładuj**
- Wybieramy pozycję z menu: **Konfiguracja→Otwórz edytor interfejsu**

Grupy podstawowe widżetów

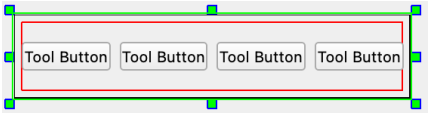
Grupa przycisków „Start”, „Pauza”, „Stop” i „Przewiń”



Mamy tutaj cztery przyciski w rozmieszczeniu poziomym, skorzystamy więc z kontenera **Horizontal Layout**.

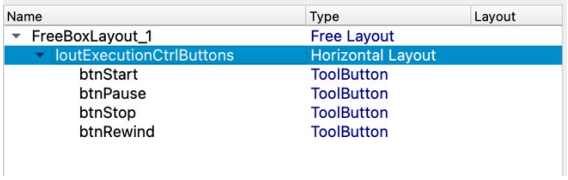
- Z listy widżetów w oknie edytora przeciągamy **Horizontal Layout** na okno simCNC
- Z tej samej listy przeciągamy do kontenera cztery widżety **Tool Button**

W efekcie powinniśmy otrzymać coś takiego:

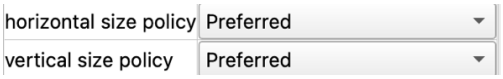


- Zmieniamy domyślne nazwy elementów na takie, które pozwolą później łatwo zorientować się jakie jest ich przeznaczenie. Tutaj zostało przyjęte: **loutExecutionCtrlButtons** dla kontenera oraz **btnStart**, **btnPause**, **btnStop** i **btnRewind** dla przycisków. Nazwę określamy w polu **id** właściwości widżetu.

Drzewo obiektów w oknie edytora powinno wyglądać następująco:



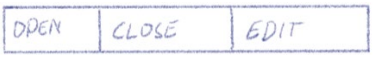
- Następnie zmieniamy politykę skalowania przycisków, by ich wielkość dostosowywała się do wielkości kontenera. Należy kliknąć na **btnStart** oraz trzymając klawisz shift – kliknąć na **btnRewind**. Mając zaznaczone wszystkie cztery przyciski zmieniamy właściwości **horizontal** i **vertical size policy** na **preffered**.



- Ustawiamy ikony dla przycisków, wybierając obiekt i klikając przycisk **Icon Selection** przy właściwości **icon** widżetu (ikony użyte w tym przykładzie można pobrać ze strony https://soft.cs-lab.eu/ui_example_icons.zip)
  - **btnStart** → „icon_play.png”
  - **btnPause** → „icon_pause.png”
  - **btnStop** → „icon_stop.png”
  - **btnRewind** → „icon_rewind.png”

Wygląd grupy jest teraz zgodny z tym, co chcieliśmy uzyskać:

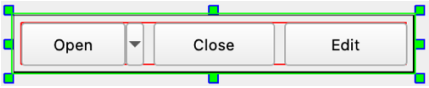
Grupa przycisków „Open”, „Close” i „Edit”



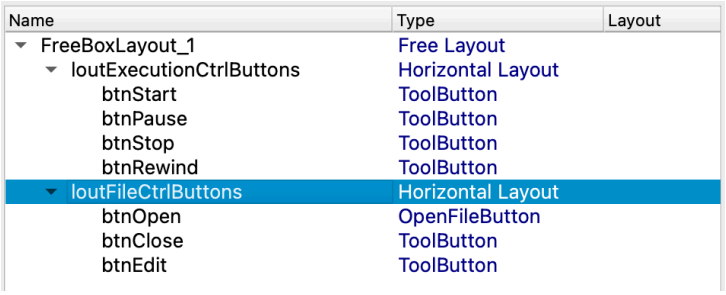
Sytuacja bardzo podobna do poprzedniej – trzy przyciski w rozmieszczeniu poziomym, z tym, że dla przycisku **Open** użyjemy dedykowanego widżetu, który będzie pamiętał listę ostatnio otwieranych plików.

- Przeciągamy kolejny obiekt **Horizontal Layout** z okna edytora do okna simCNC
- Do kontenera przeciągamy widżet **Open File Button** oraz dwa widżety **Tool Button**
- Nadajemy nazwy (właściwość **id**) dla kontenera i widżetów
  - **loutFileCtrlButtons** dla kontenera
  - **btnOpen**, **btnClose** oraz **btnEdit** dla przycisków
- Zaznaczamy wszystkie trzy przyciski i zmieniamy **horizontal** i **vertical size policy** na **Preffered**, by wielkości przycisków dostosowały się do wielkości kontenera
- Ustawiamy etykiety przycisków, modyfikując właściwość **text** widżetów
  - „Open”, „Close” oraz „Edit” odpowiednio dla **btnOpen**, **btnClose** i **btnEdit**

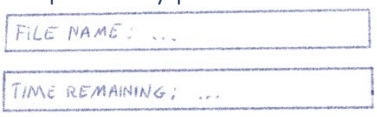
Grupa jest gotowa i wygląda następująco:



Z kolei drzewo obiektów naszego interfejsu powinno obecnie wyglądać tak:



Grupa nazwy pliku i czasu obróbki



Szkic sugerowałby, że są tutaj osobne grupy, ale pamiętajmy, że szkic jest tylko ogólną koncepcją. Często zdarza się, że podczas projektowania wprowadzane są pewne drobne zmiany, czy to w celu uzyskania lepszego wyglądu, czy też uproszczenia konstrukcji. Nie stanowi to problemu, o ile nie zmieniamy całkowicie koncepcji. W takim wypadku dobrze byłoby wykonać nowy szkic.

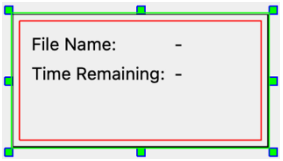
- Przeciągamy obiekt **Form Layout** z okna edytora do okna simCNC
- Nadajemy nazwę dodanemu kontenerowi (właściwość **id**) na **loutFileInfo**
- Ustawiamy kontenerowi właściwość **vertical size policy** na **Maximum** – dzięki temu unikniemy później niepotrzebnego powiększania się widżetu przy skalowaniu okna
- Przeciągamy do dodanego kontenera cztery elementy **Label** i nadajemu im nazwy (**id**) według poniższej tabeli:

lbFileNameDesc	**lbFileName**
lbTimeRemainingDesc	**lbTimeRemaining**

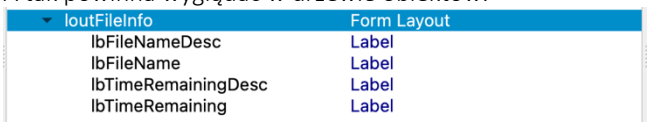
- Zmieniamy właściwość **text** dodanym etykietom według poniższej tabeli:

File Name	-
Time Remaining	-

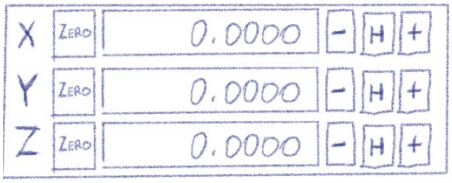
Grupa jest gotowa i powinna wyglądać tak jak poniżej:



A tak powinna wyglądać w drzewie obiektów:



Grupa wskaźników pozycji osi



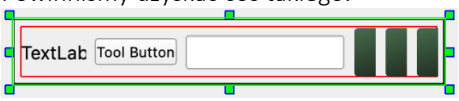
Dla każdej z osi mamy tutaj grupę sześciu widżetów w rozmieszczeniu poziomym:

- Nazwa osi (**Label**)
- Przycisk zerowania pozycji programowej (**Tool Button**)
- Pole wyświetlania i edycji pozycji (**Line Edit**)
- Trzy kontrolki włączników krańcowych i bazowania (**Digital IO indicator**)

Zaczynamy od stworzenia grupy dla jednej osi:

- Przeciągamy kontener **Horizontal Layout** z okna edytora do okna simCNC
- Do kontenera przeciągamy kolejno widżety, umieszczając je od lewej do prawej strony kontenera:
  - **Label**
  - **Tool Button**
  - **Line Edit**
  - 3x **Digital IO Indicator**

Powinniśmy uzyskać coś takiego:

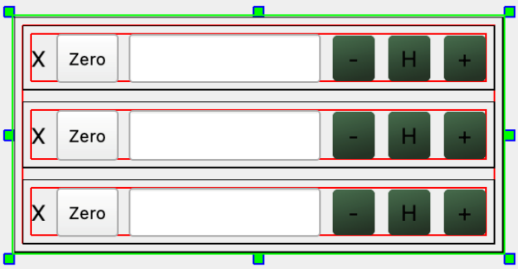


- Ustawiamy **horizontal** i **vertical size policy** dla przycisku (**Tool Button**) na **Preferred**
- Ustawiamy właściwość **text** dla przycisku na „Zero”
- Ustawiamy **vertical size policy** dla pola edycji (**Line Edit**) na **Preffered**
- Ustawiamy właściwość **text** obiektu Label na **X**
- Ustawiamy właściwość **text** kontrolek **Digital IO indicator** kolejno (od lewej) na „-”, „H” oraz „+”

Grupę jednej osi mamy praktycznie gotową:

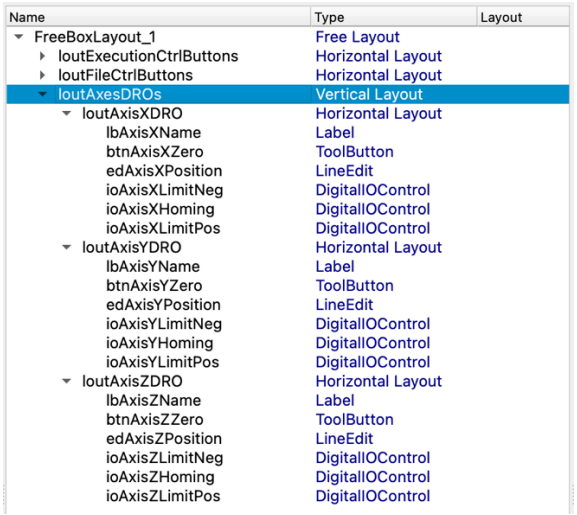
- Przeciągamy kontener **Vertical Layout** z okna edytora do okna simCNC
- Zaznaczamy kontener, w którym przed chwilą dodawaliśmy widżety i wciskamy CTRL-X (wytnij)
- Zaznaczamy pusty kontener **Vertical Layout**, który dodaliśmy i wciskamy CTRL-V (wklej) trzy razy

Efekt powinien wyglądać jak poniżej:



- Ustawiamy nazwy (właściwość **id**) widżetów
  - Grupa osi X – nazwa kontenera: **loutAxisXDRO** oraz widżety kolejno, od lewej:
    - **lbAxisXName**
    - **btnAxisXZero**
    - **edAxisXPosition**
    - **ioAxisXLimitNeg**
    - **ioAxisXHoming**
    - **ioAxisXLimitPos**
  - Grupa osi Y i Z – identycznie jak oś X, tylko w nazwach wszędzie zmieniamy „AxisX" odpowiednio na „AxisY" i „AxisZ"
  - Nazwa kontenera nadrzędnego (**Vertical Layout**): **loutAxesDROs**

Tak powinno wyglądać drzewo obiektów:



Pamiętajmy o tym, by co jakiś czas zapisać projekt wciskając CTRL-S, lub klikając klawisz **Save** w oknie edytora.

Grupa pól wyboru „Machine coords" i „Ignore Soft Limit"

Grupa dwóch widżetów typu **Checkbox** w rozmieszczeniu poziomym:

- przełączanie pomiędzy wyświetlaniem współrzędnych maszynowych i programowych (**Machine Coords)**
- wyłączenie limitów programowych (**Ignore SoftLimit**)

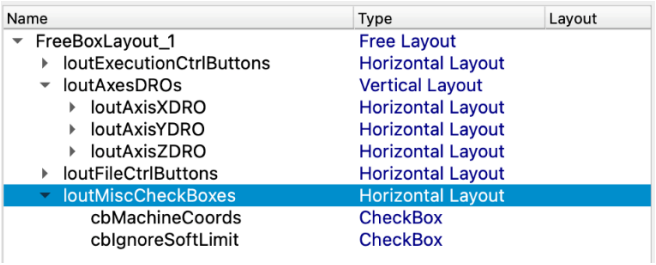
Zaczynamy standardowo od dodania kontenera (**Horizontal Layout**):

- Przeciągamy kontener **Horizontal Layout** z okna edytora do okna simCNC
- Do kontenera przeciągamy dwa widżety **Checkbox**
- Ustawiamy nazwy (właściwość **id**)
  - Kontener: **loutMiscCheckboxes**
  - **cbMachineCoords** oraz **cbIgnoreSoftLimit** dla widżetów
- Ustawiamy wyświetlany tekst (właściwość **text**) dla widżetów **Checkbox**

- „Machine Coords” i „Ignore SoftLimit”

W efekcie grupa powinna wyglądać jak poniżej:

A tak grupa powinna wyglądać w drzewie obiektów projektu:



Grupa przycisków „Ref All”, „Probe”, „Park” oraz „Go To XY”

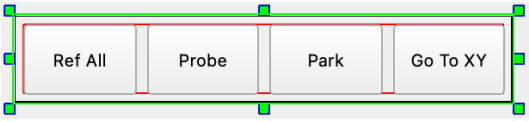
Mamy tutaj grupę czterech przycisków (**Tool Button**) w rozmieszczeniu poziomym (**Horizontal Layout**).

- „Ref All” → wywołanie bazowania osi
- „Probe” → wykonanie pomiaru narzędzia
- „Park” → przemieszczenie osi maszyny na pozycję parkowania
- „Go To XY” → ruch osi X, Y do zera programowego

Ponownie zaczynamy od dodania kontenera:

- Przeciągamy kontener **Horizontal Layout** z okna edytora do okna simCNC
- Do kontenera przeciągamy cztery przyciski **Tool Button**
- Zaznaczamy (z wciśniętym klawiszem shift) widżety przycisków i zmieniamy im właściwości **horizontal** i **vertical size policy** na **Preffered**, by ich wielkość automatycznie dopasowywała się do kontenera
- Ustawiamy nazwy obiektów (właściwość **id**):
  - Nazwa kontenera: **loutMiscButtons**
  - Nazwy widżetów: **btnRefAll**, **btnProbe**, **btnPark**, **btnGoToXY**
- Ustawiamy właściwość **text** przyciskom:
  - Odpowiednio (od lewej): „Ref All”, „Probe”, „Park”, „Go To XY”

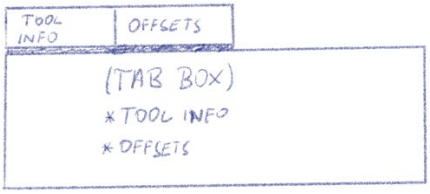
Efekt powinien być taki jak poniżej:



A tak powinna wyglądać grupa w drzewie obiektów:

Name	Type	Layout
FreeBoxLayout_1	Free Layout	
loutExecutionCtrlButtons	Horizontal Layout	
loutAxesDROs	Vertical Layout	
loutAxisXDRO	Horizontal Layout	
loutAxisYDRO	Horizontal Layout	
loutAxisZDRO	Horizontal Layout	
loutFileCtrlButtons	Horizontal Layout	
loutMiscCheckBoxes	Horizontal Layout	
cbMachineCoords	CheckBox	
cbIgnoreSoftLimit	CheckBox	
loutMiscButtons	Horizontal Layout	
btnRefAll	ToolButton	
btnProbe	ToolButton	
btnPark	ToolButton	
btnGoToXY	ToolButton	

Widżet z kartami „Tool Info” i „Offsets”



- Przeciągamy widżet **Tab Box** z okna edytora do okna simCNC
- Zaznaczamy widżet i ustawiamy ilość zakładek (właściwość **tabs quantity**) na „2”.

W drzewie obiektów widzimy trzy nowe obiekty: **TabWidget**, który kontroluje wyświetlanie i przełączanie kart oraz dwa obiekty **TabWidgetFrame**, które są kontenerami poszczególnych kart – to w nich będziemy umieszczać widżety.

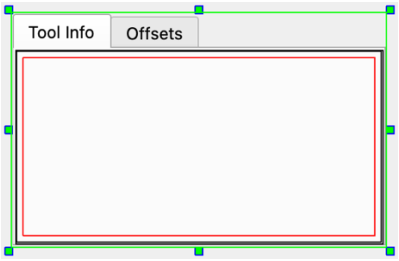
Name	Type	Layout
FreeBoxLayout_1	Free Layout	
loutExecutionCtrlButtons	Horizontal Layout	
loutAxesDROs	Vertical Layout	
loutFileCtrlButtons	Horizontal Layout	
loutMiscCheckBoxes	Horizontal Layout	
loutMiscButtons	Horizontal Layout	
TabWidget_1	TabWidget	
TabWidgetFrame_1	TabWidgetFrame	FreeBox
TabWidgetFrame_2	TabWidgetFrame	FreeBox

- Nadajemy nazwy (właściwość **id**):
  - Obiekt **TabWidget**: **twToolInfoAndOffsets**
  - Karty: **tabToolInfo** oraz **tabOffsets**

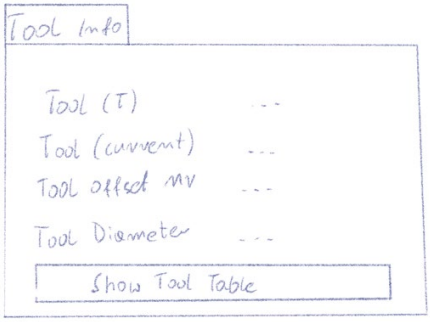
Name	Type	Layout
FreeBoxLayout_1	Free Layout	
loutExecutionCtrlButtons	Horizontal Layout	
loutAxesDROs	Vertical Layout	
loutFileCtrlButtons	Horizontal Layout	
loutMiscCheckBoxes	Horizontal Layout	
loutMiscButtons	Horizontal Layout	
twToolInfoAndOffsets	TabWidget	
tabToolInfo	TabWidgetFrame	FreeBox
tabOffsets	TabWidgetFrame	FreeBox

- Ustawiamy etykiety kartom (właściwość **title**) - Trzeba najpierw zaznaczyć odpowiednią kartę w drzewie obiektów, klikając np. na **tabToolInfo**:
  - „Tool Info” dla **tabToolInfo** i „Offsets” dla **tabOffsets**

Nasz widżet powinien w tym momencie wyglądać tak jak poniżej:



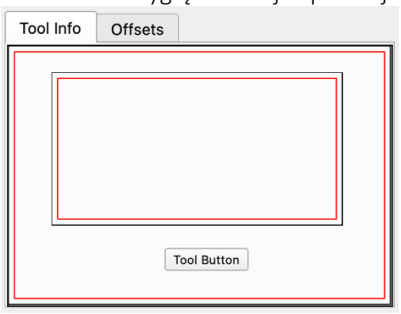
Teraz możemy przejść do tworzenia zawartości zakładek. Poniżej pokazano koncept karty **Tool Info**:



Widzimy tutaj kilka obiektów typu **Label** – opisy i wyświetlane wartości (trzykropki na szkicu) oraz jeden przycisk **Show Tool Table** do otwierania okna tabeli narzędzi. Obiekty **Label** rozmieścimy w formularzu (**Form Layout**), natomiast karta będzie miała ustawione rozmieszczenie pionowe (**Vertical Layout**) zawierające formularz i przycisk (**Tool Button**).

- Do karty **Tool Info** przeciągamy **Form Layout** z okna edytora
- Do karty **Tool Info** przeciągamy **Tool Button** z okna edytora, umieszczając go poniżej kontenera **Form Layout**

Powinno to wyglądać tak jak poniżej:



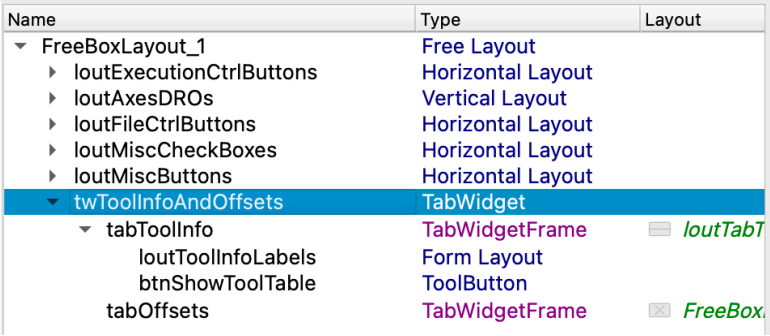
- Klikamy **tabToolInfo** w drzewie obiektów i ustawiamy właściwość **layout type** na **Pionowe**. Definuje to rozmieszczenie obiektów w karcie.
- Ustawiamy nazwę (**id**) kontenera karty na **loutTabToolInfo**

WidgetBase - Layout	
id	loutTabToolInfo
group	

- Nadajemy nazwy (**id**) dla kontenera **Form Layout** i dla przycisku
  - Kontener: **loutToolInfoLabels**
  - Przycisk: **btnShowToolTable**
- Ustawiamy właściwość **horizontal size policy** dla przycisku na **Preffered**
- Ustawiamy właściwość **text** dla przycisku na „Show Tool Table”

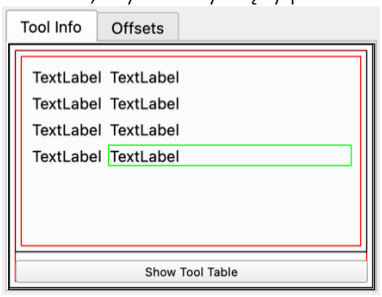
W tej chwili widżet powinien wyglądać tak:

A tak powinno to wyglądać w drzewie obiektów:



Możemy przystąpić do umieszczania obiektów **Label** w kontenerze **loutToolInfoLabels**.

- Przeciągamy osiem obiektów **Label** do kontenera **loutToolInfoLabels**. Umieszczamy je w sposób odpowiadający szkicowi, czyli cztery rzędy po dwa widżety.



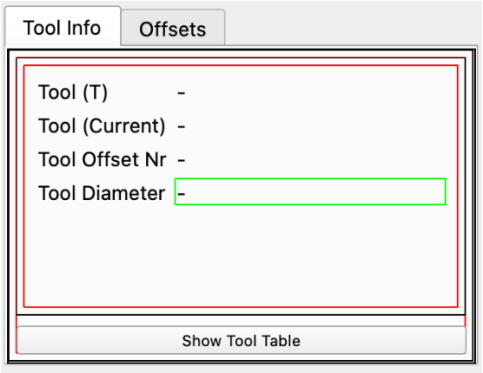
- Nadajemy widżetom **Label** nazwy (**id**), według poniższej tabeli

lbSelectedToolDesc	lbSelectedTool
lbCurrentToolDesc	lbCurrentTool
lbToolOffsetNrDesc	lbToolOffsetNr
lbToolDiameterDesc	lbToolDiameter

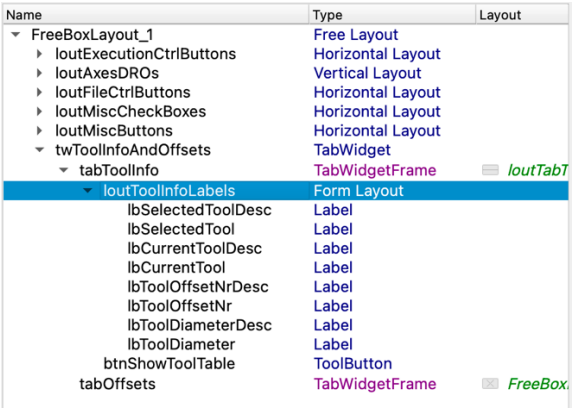
- Nadajemy widżetom **Label** właściwość **text** według poniższej tabeli

Tool (T)	-
Tool (Current)	-
Tool Offset Nr	-
Tool Diameter	-

Widżet powinien teraz wyglądać następująco:

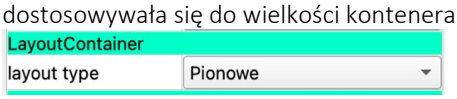


A tak drzewo obiektów:

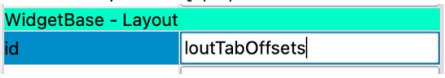


Pozostaje zakładka „Offsets". Tutaj sprawa jest prostsza, ponieważ zakładka zawiera tylko jeden widżet typu **Offsets Table**.

- Klikamy na **tabOffsets** w drzewie obiektów
- Ustawiamy rozmieszczenie zakładki (**layout type**) na **Pionowe** (typ rozmieszczenia nie ma tu większego znaczenia, bo w kontenerze będzie umieszczony tylko jeden widżet, ale jakiś trzeba wybrać, by wielkość widżetu automatycznie dostosowywała się do wielkości kontenera).

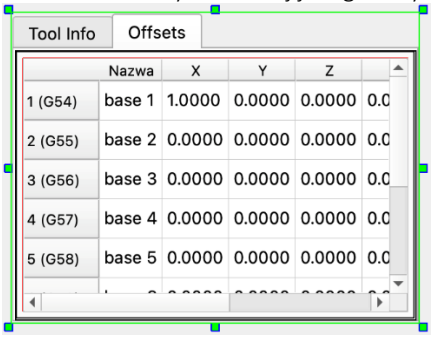


- Ustawiamy nazwę (**id**) dla kontenera zakładki na **loutTabOffsets**

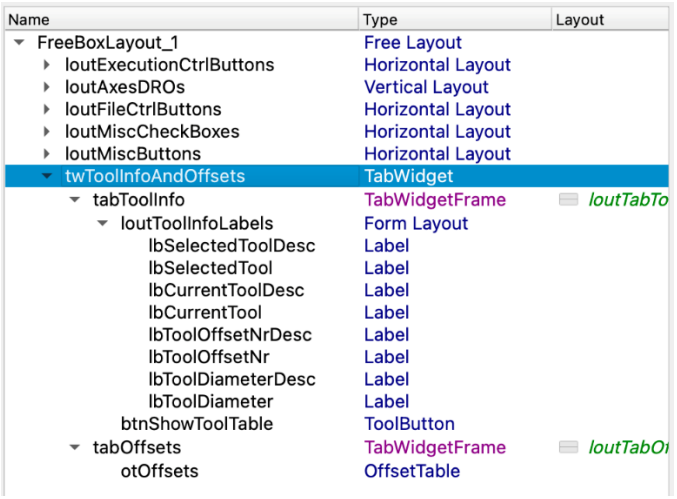


- W oknie simCNC klikamy kartę „Offsets" by ją uaktywnić
- Przeciągamy do karty „Offsets" widżet **Offset Table** z okna edytora
- Nadajemy nazwę (**id**) widżetowi **Offsets Table** na **otOffsets**

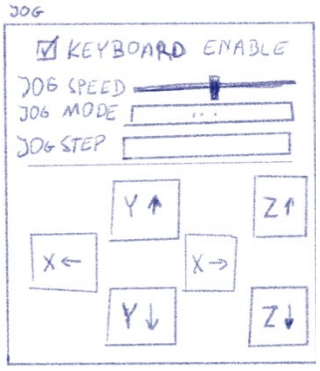
Widżet od strony wizualnej jest gotowy i powinien teraz wyglądać tak jak poniżej:



W drzewie obiektów kompletny widżet powinien wyglądać tak:



Grupa „JOG”

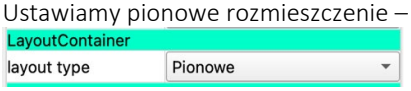


Od góry widzimy na szkicu:

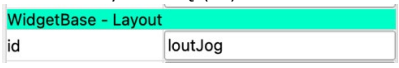
- Pole wyboru (**Check Box**) zezwolenia na kontrolę osi z klawiatury
- Etykietę (**Label**) i suwak (**Horizontal Slider**) do ustawiania prędkości ruchu JOG
- Etykietę (**Label**) i przycisk (**Tool Button**) do zmiany trybu JOG
- Etykietę (**Label**) i przycisk (**Tool Button**) do wyboru kroku JOG
- Linię oddzielającą (**Frame**)
- Grupę przycisków (**Tool Button**) rozmieszczonych w siatce (**Grid Layout**) do kontroli poszczególnych osi

Skorzystamy tutaj z widżetu-kontenera **Group Box**. Poniżej przedstawiono po kolei czynności przy projektowaniu:

- Przeciągamy widżet **Group Box** z okna edytora do okna simCNC
- Ustawiamy nazwę (**id**) widżetu na **gbJog**
- Ustawiamy pionowe rozmieszczenie – właściwość **layout type** na **Pionowe**



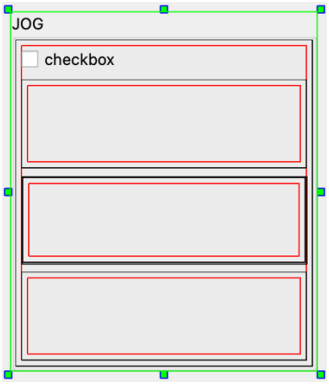
- Ustawiamy nazwę (**id**) kontenera na **loutJog**



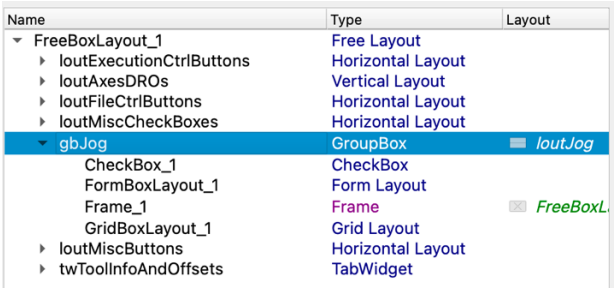
- Ustawiamy etykietę grupy (właściwość **title**) na „JOG”
- Do stworzonej grupy przeciągamy z edytora następujące widżety, umieszczając je kolejno od góry:
  - **Check Box**
  - **Form Layout**
  - **Frame**
  - **Grid Layout**

  Przy umieszczaniu widżetów w kontenerze z włączonym auto-rozmieszczeniem należy zwrócić uwagę na wyświetlające się znaczniki, które pokazują, gdzie będzie umiejscowiony nowy obiekt.

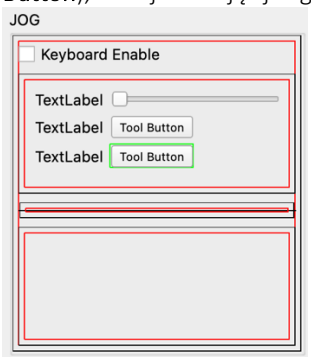
Tak w tej chwili powinien wyglądać widżet:



A tak drzewo obiektów:



- Ustawiamy nazwy (**id**) dodanym obiektom, po kolei, od góry: **cbKeyboardJogEnable**, **loutJogConfig**, **frJogLine** oraz **loutJogButtons**
- Ustawiamy właściwość **text** widżetu **cbKeyboardJogEnable** na „Keyboard Enable”
- Ustawiamy właściwość **shape** widżetu **frJogLine** na **Horizontal Line**
- Ustawiamy właściwość **vertical size policy** widżetu **frJogLine** na **Maximum**
- Do kontenera **loutJogConfig** przeciągamy z edytora trzy widżety **Label** oraz **Horizontal Slider** i dwa przyciski (**Tool Button**), umiejscawiając je zgodnie ze szkicem



- Nadajemy nowym widżetom nazwy (**id**) według poniższej tabeli:

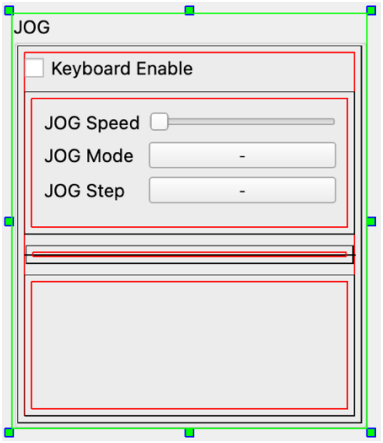
lbJogSpeedDesc	**slJogSpeed**
lbJogModeDesc	**btnJogMode**
lbJogStepDesc	**btnJogStep**

- Ustawiamy widżetom właściwość **text** według poniższej tabeli:

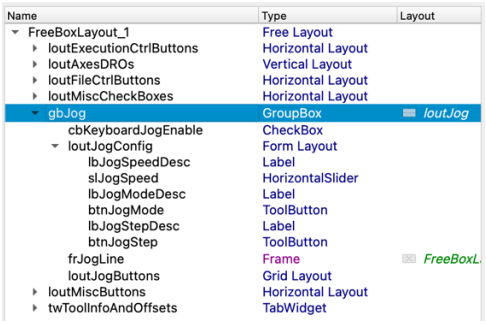
JOG Speed	(nie dotyczy)
JOG Mode	-
JOG Step	-

- Widżetom **btnJogMode** i **btnJogStep** zmieniamy właściwość **horizontal size policy** na **Preffered**, by dopasowały się w poziomie do wielkości kontenera

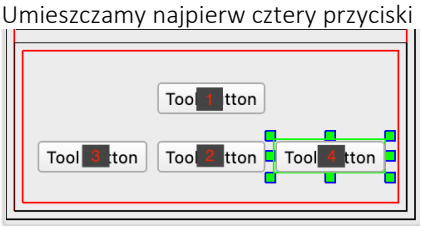
Grupa powinna obecnie wyglądać tak jak poniżej:



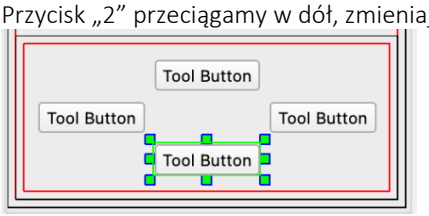
A tak drzewo obiektów:



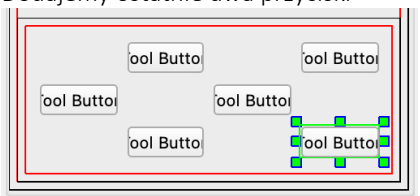
- Do grupy **loutJogButtons** dodajemy sześć przycisków **Tool Button**, umiejscawiając je zgodnie ze szkicem. Ponownie należy zwrócić uwagę na wyświetlające się znaczniki pokazujące, w którym miejscu zostanie „upuszczony” dodawany widżet. W przypadku tej grupy najłatwiej postępować w następującej kolejności:
  - Umieszczamy najpierw cztery przyciski



  - Przycisk „2” przeciągamy w dół, zmieniając jego pozycję w siatce



  - Dodajemy ostatnie dwa przyciski



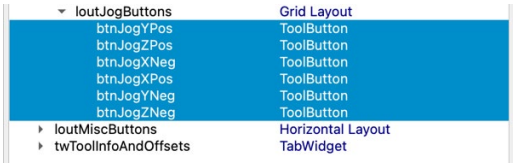
- Nadajemy nazwy (**id**) przyciskom według poniższej tabeli

	btnJogYPos		**btnJogZPos**
btnJogXNeg		**btnJogXPos**	
	btnJogYNeg		**btnJogZNeg**

- Ustawiamy przyciskom właściwość **text** według poniższej tabeli:
(strzałki to znaki *unicode*, dla ułatwienia, można je skopiować i wkleić)

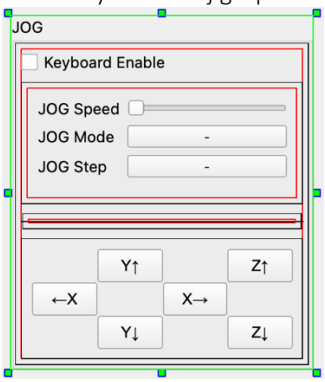
	Y↑		Z↑
←X		X→	
	Y↓		Z↓

- W drzewie obiektów zaznaczamy wszystkie przyciski **btnJog…** i ustawiamy właściwości **horizontal** i **vertical size policy** na **Preffered**

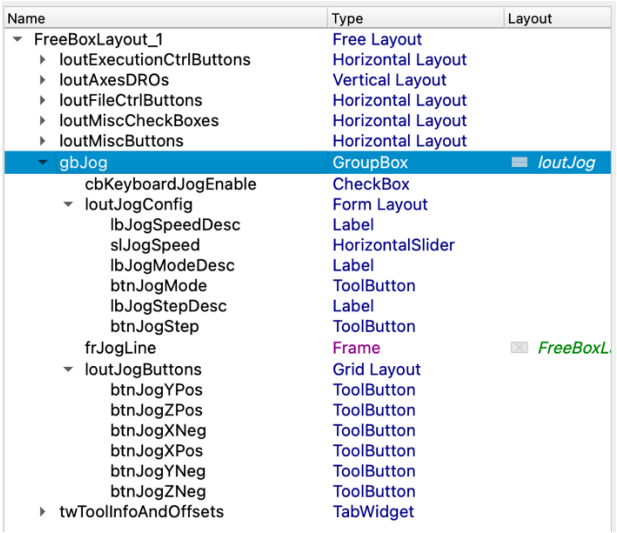


- Ustawiamy przyciskom **btnJog…** nieco większą czcionkę (właściwość **font**) – np. „Arial 14”
- Zaznaczamy obiekt **loutJogButtons** i ustawiamy na „0” następujące właściwości:
  - **left, right, top** i **bottom margin**
  - **horizontal** i **vertical spacing**

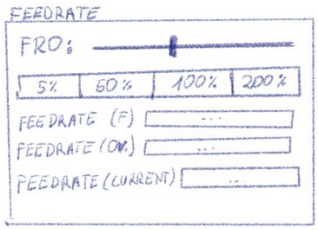
Od strony wizualnej grupa JOG jest gotowa i powinna wyglądać tak jak poniżej:



A tak grupa JOG powinna wyglądać w drzewie obiektów:



Grupa „Feedrate”

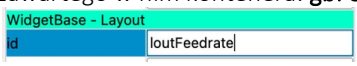


Na powyższym szkicu mamy następujące elementy (od góry):

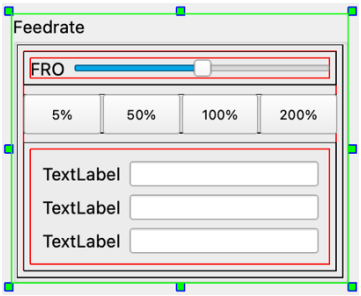
- Etykietę (**Label**) i suwak (**Horizontal Slider**) do korekty prędkości obróbki (FRO)
- Grupę (**Horizontal Layout**) przycisków (**Tool Button**) pozwalających na szybkie ustawienie często wykorzystywanych wartości FRO
- Etykietę (**Label**) i pole edycji (**Line Edit**) do wyświetlania i zmiany zadanej prędkości obróbki
- Etykietę (**Label**) i pole edycji (**Line Edit**) do wyświetlania prędkości obróbki z zastosowaną korektą FRO
- Etykietę (**Label**) i pole edycji (**Line Edit**) do wyświetlania aktualnej prędkości wypadkowej osi maszyny

Tak jak poprzednio, skorzystamy z widżetu-kontenera **Group Box**.

- Przeciągamy widżet **Group Box** z okna edytora do okna simCNC
- Ustawiamy typ pionowy rozmieszczenia zmieniając właściwość **layout type** na **Pionowe**
- Nadajemy nazwy (**id**) dla widżetu i zawartego w nim kontenera: **gbFeedrate** i **loutFeedrate**



- Zmieniamy etykietę grupy (właściwość **title)** na „Feedrate”
- Dodajemy do kontenera następujące kontenery, umieszczając je od góry do dołu:
  - 2x **Horizontal Layout**
  - **Form Layout**
- Nadajemy im nazwy (**id**): **loutFroSlider**, **loutFroButtons**, **loutFeedrateInfo**
- Do kontenera **loutFroSlider** dodajemy widżety **Label** i **Horizontal Slider** oraz nadajemy im nazwy **lbFroDesc** i **slFro**
- Zmieniamy właściwość **text** widżetu **lbFroDesc** na „FRO”
- Do kontenera **loutFroButtons** dodajemy cztery przyciski (**Tool Button**) i nadajemy im nazwy (**id**): **btnFro5**, **btnFro50**, **btnFro100**, **btnFro200**
- Zmieniamy właściwość **text** przyciskom odpowiednio na „5%”, „50%”, „100%” i „200%”
- Zmieniamy właściwość **horizontal size policy** przyciskom na **Preffered**
- Zmieniamy właściwość **vertical size policy** dla kontenera **loutFroSlider** na **Maximum** (spowoduje to, że kontener nie będzie rozciągany w pionie)
- Zmieniamy właściwość **vertical size policy** dla kontenera **loutFroButtons** na **Maximum**
- Do kontenera **loutFeedrateInfo** dodajemy trzy widżety **Label** i trzy **Line Edit** umiejscawiając je tak jak na szkicu: **Label** po lewej, **Line Edit** po prawej



- Nadajemy widżetom nazwy (**id**) zgodnie z poniższą tabelą:

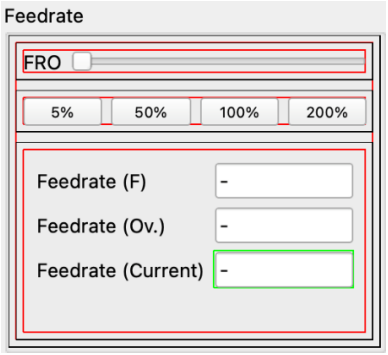
lbFeedrateDesc	**edFeedrate**
lbFeedrateOvDesc	**edFeedrateOv**
lbFeedrateCurrentDesc	**edFeedrateCurrent**

- Zmieniamy widżetom właściwość **text** zgodnie z poniższą tabelą

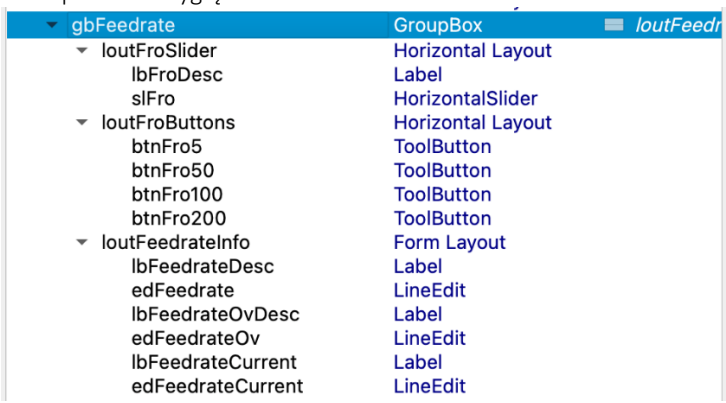
Feedrate (F)	-
Feedrate (Ov.)	-
Feedrate (Current)	-

- Ustawiamy właściwość tylko do odczytu (**read only**) widżetom **edFeedrateOv** i **edFeedrateCurrent**

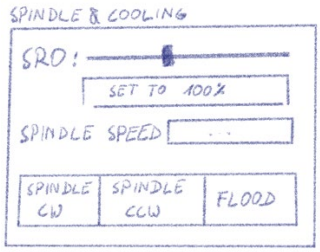
Grupa „Feedrate" jest pod względem wizualnym gotowa i powinna wyglądać tak jak poniżej:



A tak powinna wyglądać w drzewie obiektów:



Grupa „Spindle & Cooling"

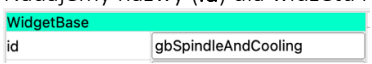


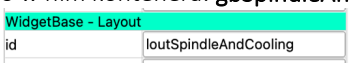
Na powyższym szkicu mamy następujące elementy (od góry):

- Etykietę (**Label**) i suwak (**Horizontal Slider**) do korekty obrotów wrzeciona (SRO)
- Przycisk resetowania korekty obrotów wrzeciona (**Tool Button**)
- Etykietę (**Label**) i pole edycji (**Line Edit**) do odczytu i modyfikacji zadanych obrotów wrzeciona
- Trzy przyciski (**Tool Button**) do załączania wrzeciona i chłodziwa

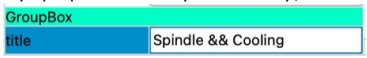
Tak jak poprzednio – skorzystamy z widżetu-kontenera **Group Box**.

- Przeciągamy widżet **Group Box** z okna edytora do okna simCNC
- Ustawiamy typ pionowy rozmieszczenia zmieniając właściwość **layout type** na **Pionowe**
- Nadajemy nazwy (**id**) dla widżetu i zawartego w nim kontenera: **gbSpindleAndCooling** i **loutSpindleAndCooling**



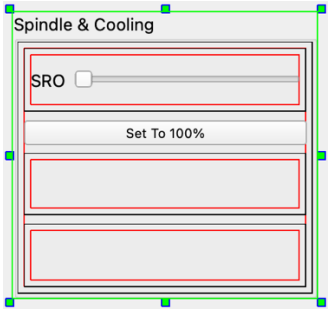


- Zmieniamy etykietę grupy (właściwość **title**) na „Spindle && Cooling" (Znak „&" jest znakiem specjalnym, dlatego żeby był poprawnie wyświetlony, trzeba go wpisać podwójnie)



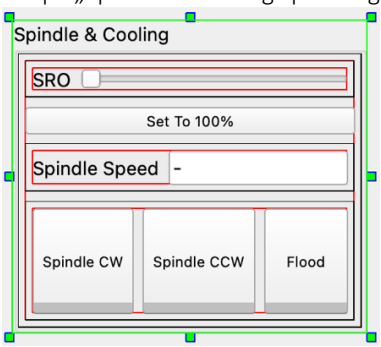
- Do kontenera dodajemy następujące elementy, umieszczając je od góry do dołu
  - **Horizontal Layout**
  - **Tool Button**

    - 2x **Horizontal Layout**
- Nadajemy nazwy (**id**): **loutSroSlider**, **btnSroReset**, **loutSpindleInfo**, **loutSpindleAndCoolingCtrl**
- Do kontenera **loutSroSlider** dodajemy **Label** oraz **Horizontal Slider**
- Nadajemy nazwy (**id**): **lbSroDesc** oraz **slSro**
- Ustawiamy właściwość **text** widżetu **lbSroDesc** na „SRO”
- Ustawiamy właściwość **text** widżetu **btnSroReset** na „Set to 100%”
- Ustawiamy właściwość **horizontal size policy** widżetu **btnSroReset** na **Preffered**

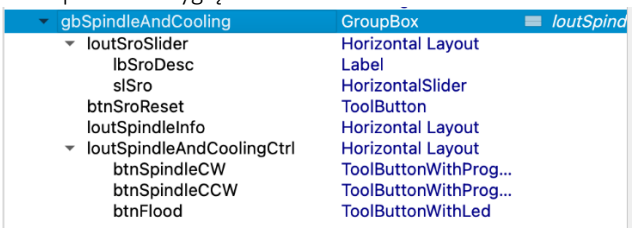


- Do kontenera **loutSpindleInfo** dodajemy **Label** oraz **Line Edit**
- Nadajemy im nazwy (**id**): **lbSpindleSpeedDesc** oraz **edSpindleSpeed**
- Ustawiamy właściwość **text** dodanym widżetom na „Spindle Speed” oraz „-”.
- Do dolnego kontenera **loutSpindleAndCoolingCtrl** dodajemy kolejno od lewej elementy:
    - 2x **Tool Button with Progress Bar**
    - **Tool Button with LED**
- Nadajemy dodanym przyciskom nazwy (**id**): **btnSpindleCW**, **btnSpindleCCW** oraz **btnFlood**
- Ustawiamy przyciskom właściwość **text**: „Spindle CW”, „Spindle CCW” oraz „Flood”
- Zmieniamy przyciskom właściwości **horizontal** i **vertical size policy** na **Preffered**
- Zaznaczamy właściwość **LED visible** dla przycisku **btnFlood**
- Zmieniamy właściwość **vertical size policy** kontenerom **loutSroSlider** oraz **loutSpindleInfo** na **Maximum**

Grupa „Spindle & Cooling” pod względem wizualnym jest gotowa i powinna wyglądać jak poniżej:



A tak powinna wyglądać w drzewie obiektów:



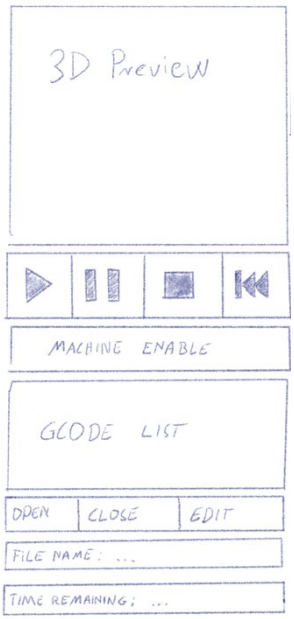
Grupy główne i rozmieszczenie

Mając stworzone wszystkie potrzebne podstawowe grupy widżetów, możemy przystąpić do projektowania grup głównych. Na szkicu z początku rozdziału można zauważyć, że projekt zbudowany jest z trzech kolumn widżetów, a w każdej kolumnie elementy są rozmieszczone pionowo.

Lewa kolumna

Na szkicu widzimy lewą główną grupę widżetów. Zawiera ona kolejno (od góry):

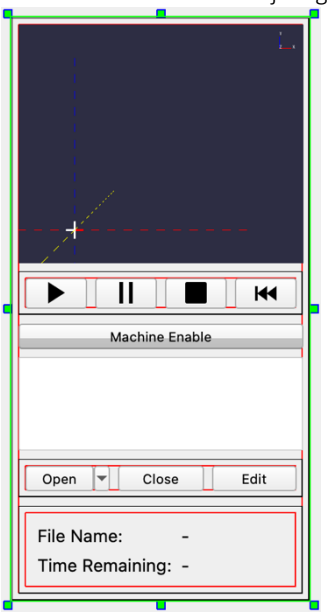
- Podgląd 3D ścieżki (**Path View**)
- Grupę przycisków **loutExecutionCtrlButtons**
- Przycisk załączania maszyny w stan gotowości (**Tool Button with LED**)
- Podgląd pliku w formie tekstowej – listę G-Kodów (**GCode List**)
- Grupę przycisków **loutFileCtrlButtons**
- Grupę nazwy pliku i prognozowanego czasu obróbki - **loutFileInfo**

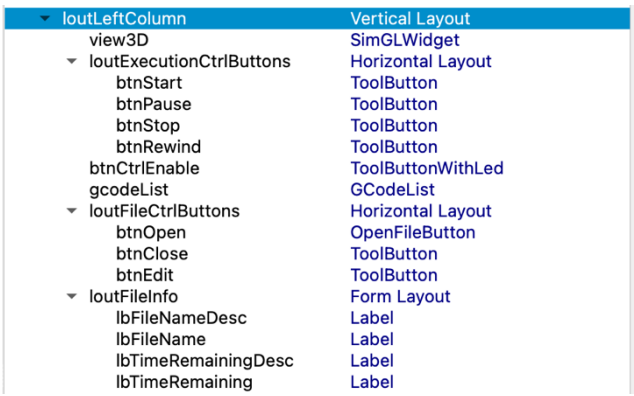


Lista operacji przy projektowaniu:

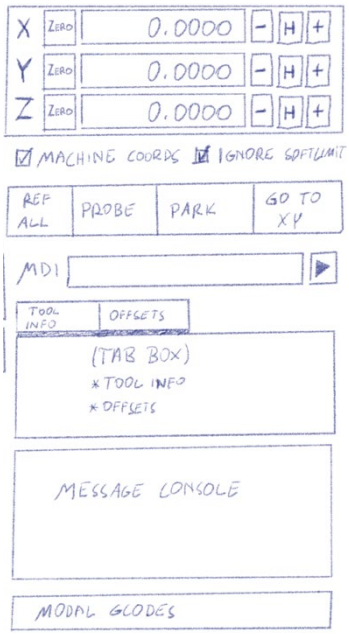
- Przeciągamy kontener rozmieszczenia pionowego **Vertical Layout** z okna edytora do okna simCNC
- Nadajemy nazwę (**id**) kontenera: **loutLeftColumn**
- Do kontenera przeciągamy następujące elementy, rozmieszczając je od góry do dołu:
  - **PathView**
  - Zaprojektowaną wcześniej grupę **loutExecutionCtrlButtons**
  - **Tool Button with LED**
  - **GCode List**
  - Zaprojektowaną wcześniej grupę **loutFileCtrlButtons**
  - Zaprojektowaną wcześniej grupę **loutFileInfo**
- Widżetom **PathView**, **ToolButton with LED** i **GCode List** nadajemy nazwy (**id**): **view3D**, **btnCtrlEnable** oraz **gcodeList**
- Ustawiamy właściwość **LED visible** dla przycisku **btnCtrlEnable**
- Ustawiamy właściwości **horizontal** i **vertical size policy** dla przycisku **btnCtrlEnable** na **Preffered**
- Ustawiamy właściwość **text** tego samego przycisku na „Machine Enable”

Lewa kolumna widżetów jest gotowa i powinna wyglądać tak jak poniżej:





Centralna kolumna



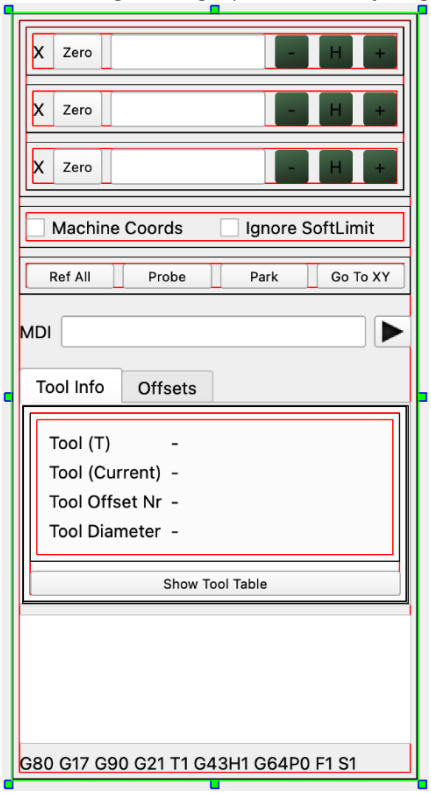
Na szkicu widzimy centralną grupę główną widżetów. Zawiera ona kolejno (od góry):

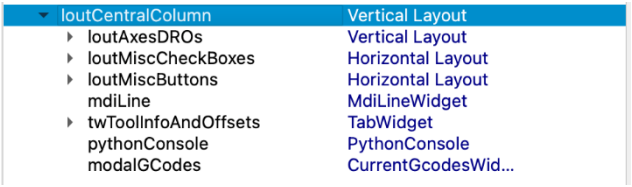
- Grupę wskaźników osi **loutAxesDROs**
- Grupę pól wyboru **loutMiscCheckBoxes**
- Grupę przycisków **loutMiscButtons**
- Widżet ręcznego wprowadzania komend maszynowych **MDI line**
- Widżet z kartami **twToolInfoAndOffsets**
- Konsolę komunikatów **Python Console**
- Widżet listy G-Kodów modalnych **CurrentGCodesWidget**

Lista operacji przy projektowaniu:

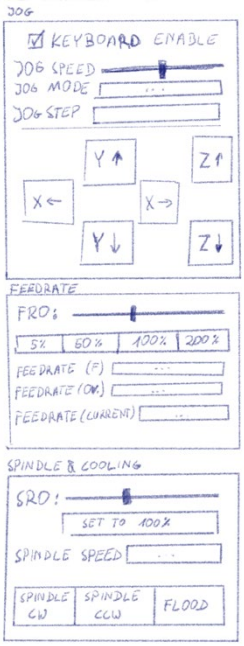
- Przeciągamy kontener rozmieszczenia pionowego **Vertical Layout** z okna edytora do okna simCNC
- Nadajemy nazwę (**id**) kontenera: **loutCentralColumn**
- Do kontenera przeciągamy następujące elementy, rozmieszczając je od góry do dołu:
    - Zaprojektowaną wcześniej grupę **loutAxesDROs**
    - Zaprojektowaną wcześniej grupę **loutMiscCheckBoxes**
    - Zaprojektowaną wcześniej grupę **loutMiscButtons**
    - **MDI line**
    - Zaprojektowany wcześniej widżet **twToolInfoAndOffsets**
    - **Python Console**
    - **Current g-codes**
- Nadajemy nazwy (**id**) widżetom **MDI line**, **Python Console** oraz **Current g-codes** odpowiednio na: **mdiLine**, **pythonConsole**, **modalGCodes**

Centralna główna grupa widżetów jest gotowa i powinna wyglądać tak jak poniżej:





Prawa kolumna



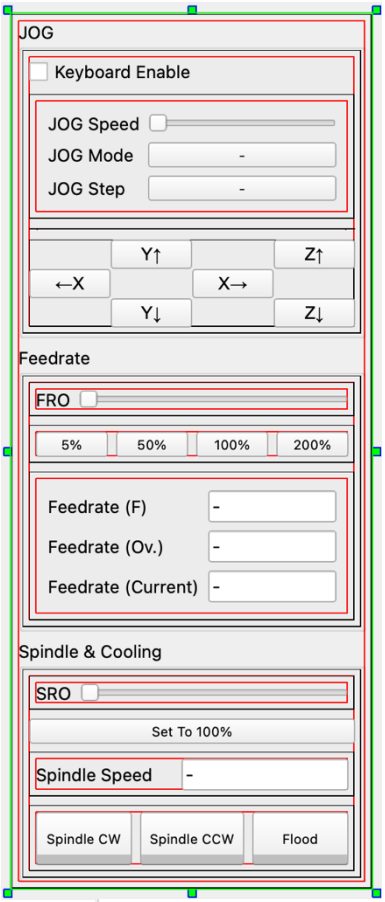
Na szkicu widzimy prawą grupę główną widżetów. Zawiera ona kolejno (od góry):

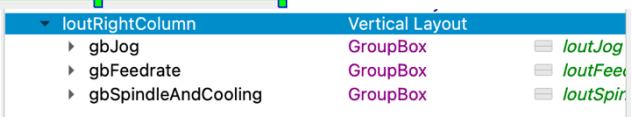
- Grupę sterowania ręcznego **gbJog**
- Grupę kontroli posuwu **gbFeedrate**
- Grupę kontroli wrzeciona i chłodziwa **gbSpindleAndCooling**

Lista operacji przy projektowaniu:

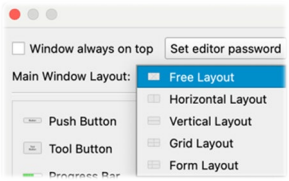
- Przeciągamy kontener rozmieszczenia pionowego **Vertical Layout** z okna edytora do okna simCNC
- Nadajemy właściwość **id** kontenera: **loutRightColumn**
- Do kontenera dodajemy następujące elementy, rozmieszczając je od góry do dołu:
    - Zaprojektowaną wcześniej grupę **gbJog**
    - Zaprojektowaną wcześniej grupę **gbFeedrate**
    - Zaprojektowaną wcześniej grupę **gbSpindleAndCooling**

Prawa grupa główna jest gotowa i powinna wyglądać tak jak poniżej:





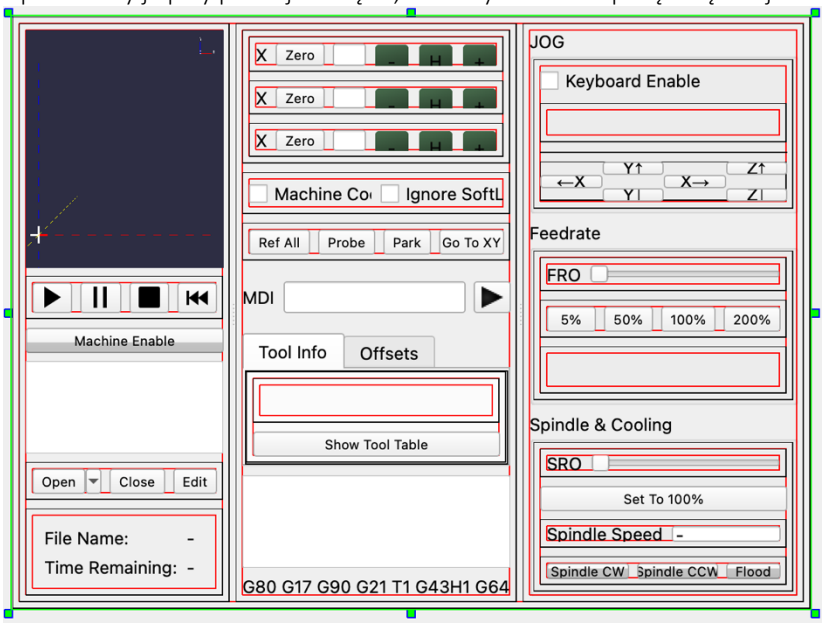
Rozmieszczenie w głównym kontenerze



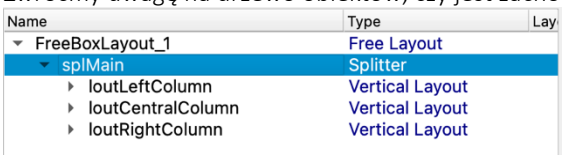
Mając przygotowane trzy główne grupy naszego interfejsu, pora na wybór rozmieszczenia w głównym oknie. Jak widać na obrazku po prawej, dla głównego kontenera mamy do wyboru cztery typy rozmieszczenia: poziome, pionowe, w siatce i w formularzu. W tym wypadku skorzystamy jednak ze **splittera**, by operator mógł w łatwy sposób zmieniać podział miejsca pomiędzy trzema kolumnami interfejsu. **Splittera** nie ma w dostępnych opcjach, ale w bardzo prosty sposób można to obejść.

Wykonajmy następujące czynności:

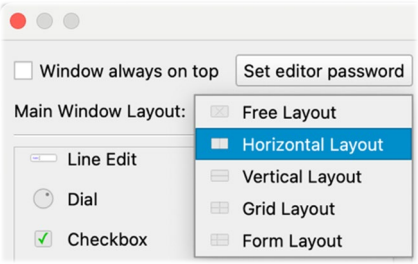
- Z okna edytora do okna simCNC przeciągamy obiekt **Splitter**
- Nadajemy mu nazwę (**id**): **splMain**
- Do kontenera **splMain** przeciągamy kolejno grupy **loutLeftColumn**, **loutCentralColumn** oraz **loutRightColumn**. Upuszczamy je przy prawej krawędzi, tak żeby zachować pożądaną kolejność kolumn.



- Zwróćmy uwagę na drzewo obiektów, czy jest zachowana poprawna hierarchia.



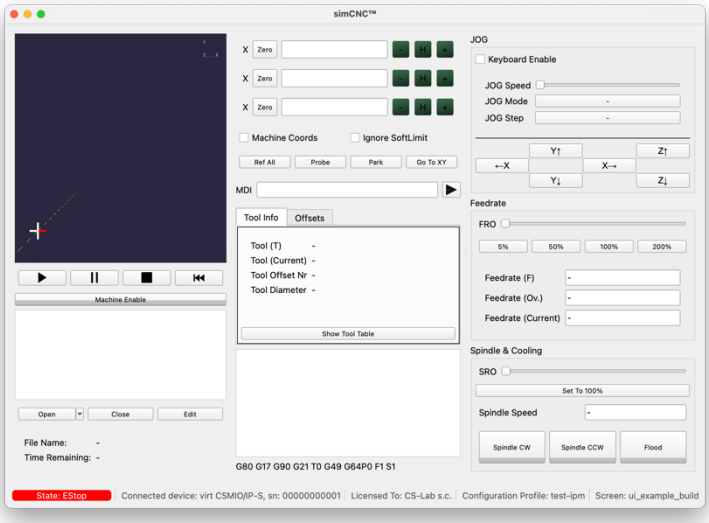
- Ustawiamy rozmieszczenie **Horizontal Layout** dla głównego okna:



Z uwagi, że po dodaniu splittera, w głównym oknie mamy tak naprawdę tylko jedną grupę (**splMain**), nie ma znaczenia czy wybierzemy rozmieszczenie poziome czy pionowe. Grupa **splMain** i tak zostanie po prostu dopasowana do rozmiarów okna.

- W drzewie obiektów zaznaczamy obiekt typu **Horizontal Layout** na samej górze i nadajemy mu nazwę (**id**): **loutMain**

Możemy teraz zamknąć na chwilę okno edytora (pamiętajmy o zachowaniu zmian) i przyjrzeć się efektom dotychczasowej pracy. Okno programu powinno wyglądać jak poniżej:



Jak widać, wygląd projektu jest zgodny z naszą koncepcją. Można go nieco uatrakcyjnić wizualnie, ale tym zajmiemy się później. Teraz jest czas na stronę funkcjonalną, czyli spowodowanie, by interfejs spełniał swoje podstawowe zadanie – prezentacji informacji i przyjmowania komend operatora.

Przypisanie funkcji / akcji

Większość obiektów interfejsu jest na obecnym etapie nieaktywna. By widżety spełniały założone zadania, musimy zdefiniować im ich funkcje wejściowe i wyjściowe. Ponownie otwieramy edytor interfejsu (menu **Konfiguracja→Otwórz edytor interfejsu**). Poniżej znajduje się tabela widżetów projektu wraz z właściwościami wejść i wyjść, które należy ustawić.
Przykładowo dla przycisku startu wykonywania G-Kodu (przycisk z ikoną „play" w lewej kolumnie pod widokiem 3D) – **btnStart** ustawiamy właściwość **Output: clicked** na **Start trajectory**).

„Id" Widżetu	Typ właściwości	wartość	opis
btnStart	Output: clicked	Start trajectory	Start G-Kodu po kliknięciu przycisku
btnPause	Output: clicked	Set Pause On/Off	Załączenie/wyłączenie pauzy wykonywania G-Kodu po kliknięciu przycisku
btnStop	Output: clicked	Stop trajectory	Zatrzymanie wykonywania G-Kodu po kliknięciu przycisku
btnRewind	Output: clicked	Rewind trajectory	Przewinięcie G-Kodu do początku po kliknięciu przycisku
btnCtrlEnable	Output: clicked	Switch to EStop/idle state	Przełączanie pomiędzy stanem EStop i gotowości po kliknięciu przycisku
btnCtrlEnable	Input: LED state	CSMIO enable	Stan diody (LED) na przycisku będzie zmieniał się wraz ze stanem gotowości sterownika CSMIO
btnOpen	-	-	Zastosowano widżet otwierania plików G-Kodu, który nie wymaga konfiguracji
btnClose	Output: clicked	Run script	W liście akcji nie ma zamykania pliku, ale można to zrobić z poziomu makra Python. Lista makr dla widżetów znajduje się w podrozdziale poniżej.
btnEdit	Output: clicked	Edit G-Code	Otworzenie edytora pliku G-Kod
lbFileName	Input: text	GCode file path	Wyświetlanie ścieżki załadowanego pliku G-Kod
lbTimeRemaining	Input: text	Remain path time	Wyświetlanie pozostałego czasu obróbki
btnAxisXZero	Output: clicked	Run Script	Zerowanie koordynaty programowej dla osi X z poziomu makra Python. Lista makr dla widżetów znajduje się w podrozdziale poniżej.
btnAxisYZero	Output: clicked	Run Script	Jak wyżej, ale dla osi Y
btnAxisZZero	Output: clicked	Run Script	Jak wyżej, ale dla osi Z
edAxisXPosition	Input: text	Axis X display position	Wyświetlanie koordynaty (programowej/maszynowej) dla osi X
edAxisXPosition	Output: returnPressed	Set axis X prog position	Modyfikacja koordynaty programowej dla osi X po wciśnięciu klawisza „return"
edAxisYPosition	Input: text	Axis Y display position	Wyświetlanie koordynaty (programowej/maszynowej) dla osi Y
edAxisYPosition	Output: returnPressed	Set axis Y prog position	Modyfikacja koordynaty programowej dla osi Y po wciśnięciu klawisza „return"
edAxisZPosition	Input: text	Axis Z display position	Wyświetlanie koordynaty (programowej/maszynowej) dla osi Z
edAxisZPosition	Output: returnPressed	Set axis Z prog position	Modyfikacja koordynaty programowej dla osi Z po wciśnięciu klawisza „return"
ioAxisXLimitNeg	Input: state	Signal value (IP;0;mkit;0→limit--;0)	Stan kontrolki powiązany ze stanem sygnału „limit--" MotionKit'a nr 0 CSMIO/IP
ioAxisXHoming	Input: state	Signal value (IP;0;mkit;0→Home;0)	Stan kontrolki powiązany ze stanem sygnału „home" MotionKit'a nr 0 CSMIO/IP
ioAxisXLimitPos	Input: state	Signal value (IP;0;mkit;0→limit++;0)	Stan kontrolki powiązany ze stanem sygnału „limit++" MotionKit'a nr 0 CSMIO/IP
ioAxisYLimitNeg	Input: state	Signal value (IP;0;mkit;1→limit--;0)	Stan kontrolki powiązany ze stanem sygnału „limit--" MotionKit'a nr 1 CSMIO/IP
ioAxisYHoming	Input: state	Signal value (IP;0;mkit;1→Home;0)	Stan kontrolki powiązany ze stanem sygnału „home" MotionKit'a nr 1 CSMIO/IP
ioAxisYLimitPos	Input: state	Signal value (IP;0;mkit;1→limit++;0)	Stan kontrolki powiązany ze stanem sygnału „limit++" MotionKit'a nr 1 CSMIO/IP
ioAxisZLimitNeg	Input: state	Signal value (IP;0;mkit;2→limit--;0)	Stan kontrolki powiązany ze stanem sygnału „limit--" MotionKit'a nr 2 CSMIO/IP
ioAxisZHoming	Input: state	Signal value (IP;0;mkit;2→Home;0)	Stan kontrolki powiązany ze stanem sygnału „home" MotionKit'a nr 2 CSMIO/IP

„Id" Widżetu	Typ właściwości	wartość	opis
ioAxisZLimitPos	Input: state	Signal value (IP;0;mkit;2→limit++;0)	Stan kontrolki powiązany ze stanem sygnału „limit++" MotionKit'a nr 2 CSMIO/IP
cbMachineCoords	Input: checkbox state	Ref position displayed	Stan pola wyboru powiązany z tym czy aktualnie wyświetlane są koordynaty maszynowe
cbMachineCoords	Output: stateChanged	Set position display to ref	Przełączenie na wyświetlanie koordynat maszynowych po zaznaczeniu pola wyboru
cbIgnoreSoftLimit	Input: checkbox state	Global soft limit disabled	Stan pola wyboru powiązany ze stanem wyłączenia limitów programowych
cbIgnoreSoftLimit	Output: stateChanged	Disable global soft limit	Wyłączenie limitów programowych po zaznaczeniu pola wyboru
btnRefAll	Output: clicked	Ref all axes	Uruchomienie bazowania wszystkich osi po kliknięciu przycisku
btnProbe	Output: clicked	Execute probing script	Uruchomienie domyślnego makra Python pomiaru narzędzia po kliknięciu przycisku
btnPark	Output: clicked	Run script	Uruchomienie makra Python jazdy do pozycji parkowej. Lista makr dla widżetów znajduje się w podrozdziale poniżej.
btnGoToXY	Output: clicked	Run script	Uruchomienie makra Python jazdy do pozycji offsetu roboczego. Lista makr dla widżetów znajduje się w podrozdziale poniżej.
lbSelectedTool	Input: text	Selected tool number	Wyświetlanie nr aktualnie wybranego narzędzia
lbCurrentTool	Input: text	Spindle tool number	Wyświetlanie nr narzędzia, które aktualnie znajduje się we wrzecionie
lbToolOffsetNr	Input: text	Tool offset number	Wyświetlanie wybranego nr offsetu narzędzia
lbToolDiameter	Input: text	Tool diameter	Wyświetlanie średnicy aktualnego narzędzia
btnShowToolTable	Output: clicked	Tool Table	Wyświetlenie listy narzędzi po kliknięciu przycisku
cbKeyboardJogEnable	Input: checkbox state	Key Control	Stan pola wyboru powiązany ze stanem załączenia kontroli maszyny z klawiatury
cbKeyboardJogEnable	Output: state changed	Key Control	Zezwolenie na kontrolę maszyny z klawiatury, jeśli pole wyboru jest zaznaczone
slJogSpeed	Input: value	Jog speed	Pozycja suwaka powiązana z prędkością JOG'a
slJogSpeed	Output: valueChanged	Set Jog Speed	Zmiana pozycji suwaka będzie zmieniać aktualną prędkość JOG'a
btnJogMode	Input: text	Jog mode	Aktualny tryb JOG'a będzie ustawiał tekst na przycisku
btnJogMode	Output: clicked	Set Jog mode	Kliknięcie przycisku będzie zmieniać aktualny tryb JOG'a
btnJogStep	Input: text	Jog step	Aktualny krok JOG'a będzie ustawiać tekst na przycisku
btnJogStep	Output: clicked	Run Script	Uruchomienie makra Python, które będzie cyklicznie przełączać pomiędzy wartościami. Lista makr dla widżetów znajduje się w podrozdziale poniżej.
btnJogXNeg	Output: pressed	Jog X- pressed	Wciśnięty przycisk wywoła jazdę JOG osią X w kierunku ujemnym
btnJogXNeg	Output: released	JOG X- released	Puszczony przycisk zatrzyma jazdę JOG osią X w kierunku ujemnym
btnJogXPos	Output: pressed	Jog X+ pressed	Wciśnięty przycisk wywoła jazdę JOG osią X w kierunku dodatnim
btnJogXPos	Output: released	JOG X+ released	Puszczony przycisk zatrzyma jazdę JOG osią X w kierunku dodatnim
btnJogYNeg	Output: pressed	Jog Y- pressed	Wciśnięty przycisk wywoła jazdę JOG osią Y w kierunku ujemnym
btnJogYNeg	Output: released	JOG Y- released	Puszczony przycisk zatrzyma jazdę JOG osią Y w kierunku ujemnym
btnJogYPos	Output: pressed	Jog Y+ pressed	Wciśnięty przycisk wywoła jazdę JOG osią Y w kierunku dodatnim
btnJogYPos	Output: released	JOG Y+ released	Puszczony przycisk zatrzyma jazdę JOG osią Y w kierunku dodatnim
btnJogZNeg	Output: pressed	Jog Z- pressed	Wciśnięty przycisk wywoła jazdę JOG osią Z w kierunku ujemnym
btnJogZNeg	Output: released	JOG Z- released	Puszczony przycisk zatrzyma jazdę JOG osią Z w kierunku ujemnym

„Id" Widżetu	Typ właściwości	wartość	opis
btnJogZPos	Output: pressed	Jog Z+ pressed	Wciśnięty przycisk wywoła jazdę JOG osią Z w kierunku dodatnim
btnJogZPos	Output: released	JOG Z+ released	Puszczony przycisk zatrzyma jazdę JOG osią Z w kierunku dodatnim
slFro	Input: value	Fro	Pozycja suwaka będzie ustawiana wraz ze zmianami wartości FRO
slFro	Output: valueChanged	Set Fro	Zmiana pozycji suwaka będzie zmieniała aktualną wartość FRO
btnFro5	Output: clicked	Run Script	Uruchomienie makra Python, które będzie ustawiało wartość FRO na 5%. Lista makr dla widżetów znajduje się w podrozdziale poniżej.
btnFro50	Output: clicked	Run Script	Uruchomienie makra Python, które będzie ustawiało wartość FRO na 50%. Lista makr dla widżetów znajduje się w podrozdziale poniżej.
btnFro100	Output: clicked	Run Script	Uruchomienie makra Python, które będzie ustawiało wartość FRO na 100%. Lista makr dla widżetów znajduje się w podrozdziale poniżej.
btnFro200	Output: clicked	Run Script	Uruchomienie makra Python, które będzie ustawiało wartość FRO na 200%. Lista makr dla widżetów znajduje się w podrozdziale poniżej.
edFeedrate	Input: text	Feedrate	Wyświetlanie aktualnie ustawionej prędkości obróbki
edFeedrate	Output: returnPressed	Set Feedrate	Ustawianie prędkości obróbki przy naciśnięciu klawisza return.
edFeedrateOv	Input: text	Feedrate override	Wyświetlanie aktualnie ustawionej prędkości obróbki, z uwzględnieniem FRO
edFeedrateCurrent	Input: text	Current velocity	Wyświetlanie aktualnej prędkości wypadkowej – prędkości narzędzia względem materiału.
slSro	Input: text	Sro	Pozycja suwaka będzie ustawiana wraz ze zmianami SRO
slSro	Output: valueChanged	Set Sro	Zmiana pozycji suwaka będzie zmieniała aktualną wartość SRO
btnSroReset	Output: clicked	Run Script	Uruchomienie makra Python, które będzie ustawiało wartość SRO na 100%. Lista makr dla widżetów znajduje się w podrozdziale poniżej.
edSpindleSpeed	Input: text	Spindle Speed	Wyświetlanie aktualnie ustawionych obrotów wrzeciona
edSpindleSpeed	Output: returnPressed	Set Spindle Speed	Ustawianie prędkości obrotowej wrzeciona przy naciśnięciu klawisza return
btnSpindleCW	Output: clicked	Run Spindle Clockwise	Włączenie prawych obrotów wrzeciona po kliknięciu przycisku
btnSpindleCW	Input: value	Spindle CW percent	Pasek na przycisku będzie obrazował osiągnięcie zadanych obrotów (obroty prawe)
btnSpindleCCW	Output: clicked	Run Spindle Counter-Clockwise	Włączenie lewych obrotów wrzeciona po kliknięciu przycisku
btnSpindleCCW	Input: value	Spindle CCW percent	Pasek na przycisku będzie obrazował osiągnięcie zadanych obrotów (obroty lewe)
btnFlood	Output: clicked	Set Flood On/Off	Załączanie/wyłączanie chłodziwa
btnFlood	Input: LED state	Flood on	Dioda na przycisku będzie zapalona, gdy chłodziwo jest załączone

Makra Python dla widżetów z akcją „Run script”

Jak można zauważyć w powyższej tabeli, niektóre widżety nie mają przypisanej konkretnej funkcji, tylko **Run script**, czyli wywołanie makra Python. Poniżej znajdują się nazwy plików wraz z ich kodami źródłowymi. Do stworzenia plików można użyć wbudowanego edytora simCNC (menu **Makra→Pokaż edytor makr**), lub dowolnego innego edytora np. **VS Code**. Pliki najlepiej zachować w katalogu projektowanego ekranu, w podkatalogu scripts. Pliki muszą mieć rozszerzenie „.py”.

Dla Windows będzie to ścieżka: *„C:\Program Files\simCNC\screens\ui_example\scripts”*
Dla Linux: *“/opt/simCNC/screens/ui_example/scripts”*
Dla macOS: *“/Applications/CS-Lab/simCNC.app/Contents/MacOS/screens/ui_example/scripts/”*

*Makro dla przycisku **btnClose** - plik „btnClose.py”*

```
d.closeGCodeFile( )
print(„GCode file closed.”)
```

*Makro dla przycisku **btnAxisXZero** - plik „btnAxisXZero.py”*

```
d.setAxisProgPosition( Axis.X, 0 )
print("Axis X prog position set to 0.000")
```

*Makro dla przycisku **btnAxisYZero** - plik „btnAxisYZero.py”*

```
d.setAxisProgPosition( Axis.Y, 0 )
print("Axis Y prog position set to 0.000")
```

*Makro dla przycisku **btnAxisZZero** - plik „btnAxisZZero.py”*

```
d.setAxisProgPosition( Axis.Z, 0 )
print("Axis Z prog position set to 0.000")
```

*Makro dla przycisku **btnPark** - plik „btnPark.py”*

```
d.executeGCode( "G0G53 Z0" );
d.executeGCode( "G0G53 X0 Y0" );
print("Go to park position finished")
```

*Makro dla przycisku **btnGoToXY** - plik „btnGoToXY.py”*

```
d.executeGCode( "G0G53 Z0" );
d.executeGCode( "G0 X0 Y0" );
print("Go to material zero XY position finished")
```

*Makro dla przycisku **btnJogStep** - plik „btnJogStep.py”*

```
currentStep = d.getJogStep( )
newStep = currentStep * 10.0
if newStep > 1.0:
        newStep = 0.001
d.setJogStep( newStep )
print("JOG step set to: {: 3f}".format(newStep))
```

*Makro dla przycisku **btnFro5** - plik „btnFro5.py”*

```
d.setFRO( 5 )
print("FRO set to: {:.1f}%".format(d.getFRO( )))
```

*Makro dla przycisku **btnFro50** - plik „btnFro50.py”*

```
d.setFRO( 50 )
print("FRO set to: {:.1f}%".format(d.getFRO( )))
```

*Makro dla przycisku **btnFro100** - plik „btnFro100.py”*

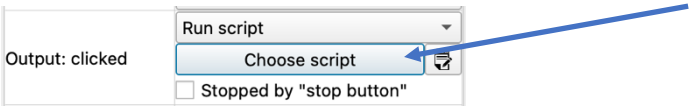
```
d.setFRO( 100 )
print("FRO set to: {:.1f}%".format(d.getFRO( )))
```

*Makro dla przycisku **btnFro200** - plik „btnFro200.py”*

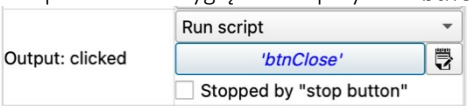
```
d.setFRO( 200 )
print("FRO set to: {:.1f}%".format(d.getFRO( )))
```

Przypisanie makr widżetom

W celu przypisania makra do widżetu należy zaznaczyć widżet i w liście właściwości, pod akcją **Run script** kliknąć przycisk wyboru makra.



Tak powinno to wyglądać dla przycisku **btnClose**, po wybraniu makra „btnClose.py”



W ten sposób przypisujemy widżetom wszystkie powyższe makra.

Uzupełnienia i drobne poprawki

Na tym etapie mamy już interfejs, który działa i można z niego korzystać. Zanim zajmiemy się wizualnymi udoskonaleniami i kosmetyką, wykonamy kilka drobnych poprawek.

1. Poprawka tekstu w etykietach osi. Powinno być **X**, **Y**, **Z**, a jest trzy razy **X**.

   W trybie edycji, zmieniamy właściwość **text** widżetu **lbAxisYName** na „Y”, a widżetu **lbAxisZName** na „Z”.
2. Ustawienie zakresu wartości dla suwaków **slJogSpeed**, **slFro** oraz **slSro**
   a. Właściwość **maximum** widżetu **slJogSpeed** ustawiamy na 100
   b. Właściwość **maximum** widżetu **slFro** ustawiamy na 200
   c. Właściwość **maximum** widżetu **slSro** ustawiamy na 200
3. Ustawienie zakresu wartości dla pasków wizualizacji obrotów wrzeciona na przyciskach **btnSpindleCW** i **btnSpindleCCW**
   a. Właściwość **maximum** widżetu **btnSpindleCW** na 1
   b. Właściwość **maximum** widżetu **btnSpindleCCW** na 1
4. Ustawienie formatu wyświetlania koordynat na X.XXX
   a. W polu właściwości **display format** widżetu **edAxisXPosition**, **edAxisYPosition** oraz **edAxisZPosition** wpisujemy „%.3f” (bez cudzysłowu)

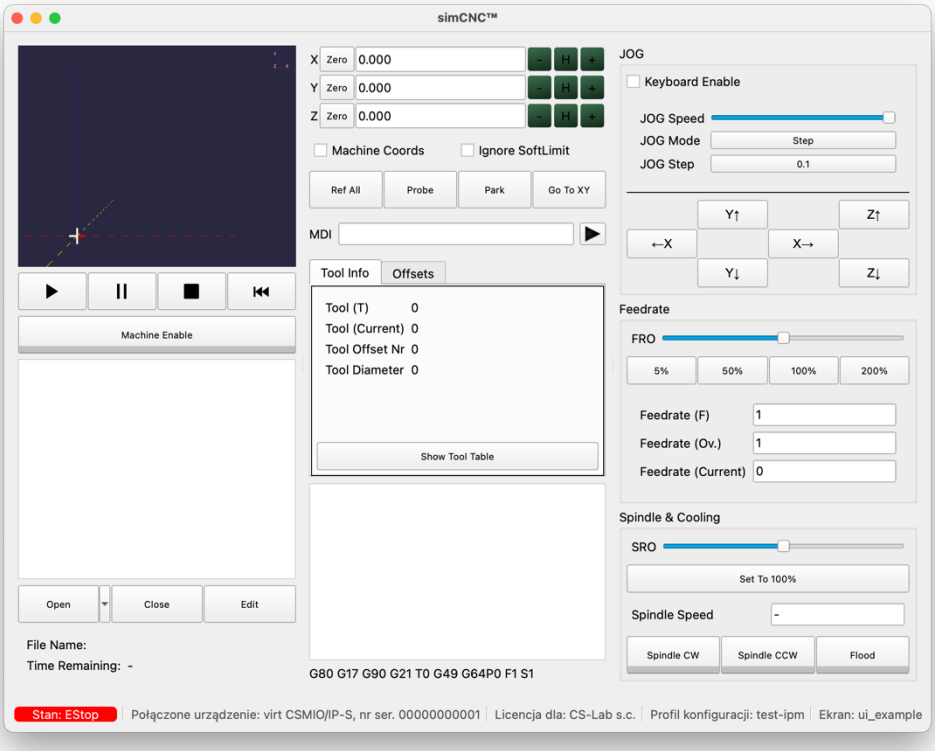
Stylizacja

Jak wspomniano wcześniej, ten krok jest opcjonalny. Nasz interfejs na tym etapie powinien funkcjonować prawidłowo. Warto poświęcić jednak jeszcze odrobinę pracy, by udoskonalić jego wygląd.

Na początek „dostroimy" nieco skalowanie i zasady podziału miejsca niektórych widżetów i kontenerów. W tym celu ustawiamy następujące właściwości obiektom:

Nazwa („id") obiektu	Nazwa właściwości	wartość
loutLeftColumn	stretch	1,0,0,1,0,0
loutExecutionCtrlButtons	minimum height	40
loutExecutionCtrlButtons	left, right, top, bottom margin	0
loutExecutionCtrlButtons	spacing	1
btnCtrlEnable	minimum height	40
loutFileCtrlButtons	minimum height	40
loutFileCtrlButtons	left, right, top, bottom margin	0
loutFileCtrlButtons	spacing	1
loutFileInfo	left, right, top, bottom margin	0
loutAxesDROs	left, right, top, bottom margin	0
loutAxesDROs	spacing	1
loutAxisXDRO	left, right, top, bottom margin	1
loutAxisXDRO	spacing	1
loutAxisYDRO	left, right, top, bottom margin	1
loutAxisYDRO	spacing	1
loutAxisZDRO	left, right, top, bottom margin	1
loutAxisZDRO	spacing	1
loutMiscButtons	minimum height	40
loutMiscButtons	left, right, top, bottom margin	0
loutMiscButtons	spacing	1
loutToolInfoLabels	left, right, top, bottom margin	0
btnShowToolTable	minimum height	30
loutRightColumn	stretch	1,0,0
loutJogConfig	vertical size policy	Maximum
frJogLine	vertical size policy	Maximum
loutFroButtons	minimum height	30
loutFroButtons	left, right, top, bottom margin	0
loutFroButtons	spacing	1
btnFro5	vertical size policy	Preferred
btnFro50	vertical size policy	Preferred
btnFro100	vertical size policy	Preferred
btnFro200	vertical size policy	Preferred
btnSroReset	minimum height	30
loutSpindleAndCoolingCtrl	minimum height	40
loutSpindleAndCoolingCtrl	left, right, top, bottom margin	0
loutSpindleAndCoolingCtrl	spacing	1

Interfejs powinien wyglądać teraz nieco zgrabniej i kompaktowo:



Ustawmy jeszcze kolor czerwony dla kontrolek sygnalizujących stany włączników krańcowych:

Nazwa („id”) obiektu	Nazwa właściwości	wartość
ioAxisXLimitNeg	color	(czerwony)
ioAxisXLimitPos	color	(czerwony)
ioAxisYLimitNeg	color	(czerwony)
ioAxisYLimitPos	color	(czerwony)
ioAxisZLimitNeg	color	(czerwony)
ioAxisZLimitPos	color	(czerwony)

(…) oraz wyrównanie do prawej i nieco większą czcionkę dla widżetów nazw osi i wyświetlających koordynaty:

Nazwa („id”) obiektu	Nazwa właściwości	wartość
edAxisXPosition	horizontal alignment	Right
edAxisXPosition	font	Arial 20
lbAxisXName	font	Arial 20
edAxisYPosition	horizontal alignment	Right
edAxisYPosition	font	Arial 20
lbAxisYName	font	Arial 20
edAxisZPosition	horizontal alignment	Right
edAxisZPosition	font	Arial 20
lbAxisZName	font	Arial 20

(…) wyłączmy dodatkową ramkę w widżecie z zakładkami „Tool Info” i „Offsets”:

Nazwa („id”) obiektu	Nazwa właściwości	wartość
tabToolInfo	shape	Brak ramki
tabOffsets	shape	Brak ramki

(…) wycentrujmy wyświetlanie niektórych parametrów

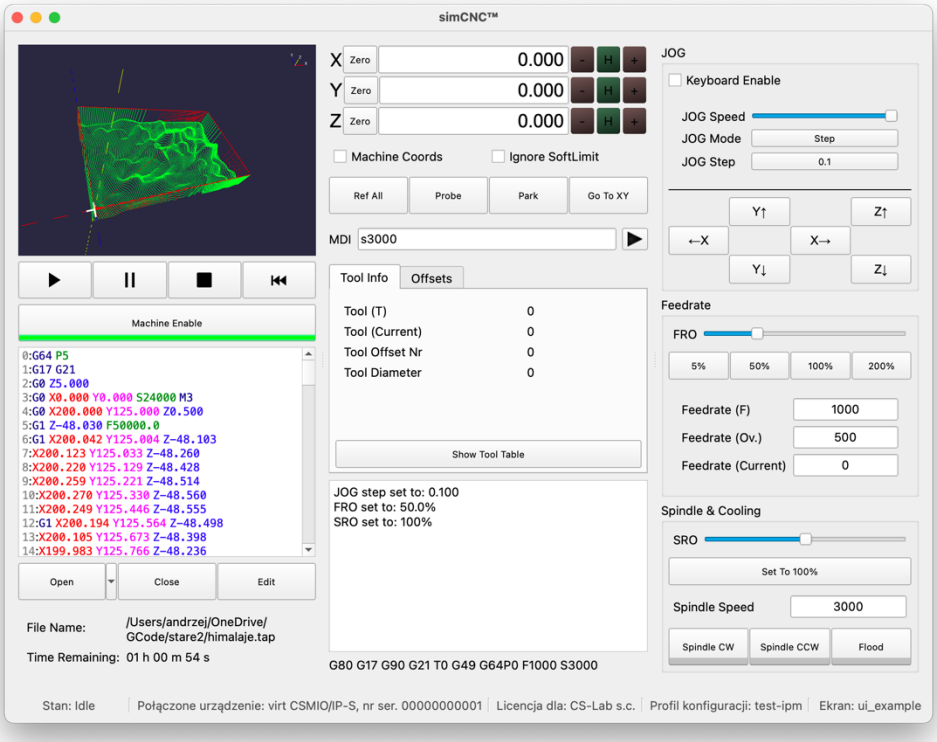
Nazwa („id”) obiektu	Nazwa właściwości	wartość
lbSelectedTool	Horizontal alignment	Wycentrowany

Nazwa („id”) obiektu	Nazwa właściwości	wartość
lbCurrentTool	Horizontal alignment	Wycentrowany
lbToolOffsetNr	Horizontal alignment	Wycentrowany
lbToolDiameter	Horizontal alignment	Wycentrowany
edFeedrate	Horizontal alignment	Wycentrowany
edFeedrateOv	Horizontal alignment	Wycentrowany
edFeedrateCurrent	Horizontal alignment	Wycentrowany
edSpindleSpeed	Horizontal alignment	Wycentrowany

i ustawmy właściwości widżetu lbFileName, by mieściły się dłuższe nazwy:

Nazwa („id”) obiektu	Nazwa właściwości	wartość
lbFileName	Word wrap	(zaznaczone)
lbFileName	Vertical size policy	Maximum
lbFileName	Minimum height	32

Interfejs prezentuje się teraz następująco:



Ostateczna kosmetyka z użyciem arkuszy stylu css

Stylizacja z użyciem arkuszy stylu to narzędzie o dużych możliwościach, ale przeznaczone raczej dla bardziej zaawansowanych użytkowników. Dokładne omówienie wszystkich właściwości wykracza poza ramy niniejszej instrukcji. W dalszej części tego rozdziału przedstawiono prosty przykład, a dla zainteresowanych poniższe linki, gdzie można znaleźć bardziej szczegółowe informacje na ten temat.

https://www.w3schools.com/css/css_intro.asp
https://doc.qt.io/qt-5/stylesheet-syntax.html

Zachęcam też do zapoznania się i eksperymentowania z plikami css w projektach ekranów domyślnych („default"). Proszę pamiętać tylko o wykonaniu kopii i pracowaniu na niej, ponieważ wszelkie zmiany w ekranach domyślnych mogą zostać nadpisane po aktualizacji wersji simCNC.

Właściwości przekazywane przez simCNC

Do arkuszy css simCNC przekazuje kilka właściwości, które umożliwiają zmianę wyglądu elementów w zależności od np. stanu programu, czy aktywnego trybu ciemnego.

Nazwa właściwości	Wartości	Opis
darkTheme	true false	Zwraca **true** jeśli aktywny jest ciemny motyw
state	estop idle homing trajectory jog mdi mpg	Zwraca aktualny stan maszyny
csmioState	init disabled prepare run fault	Zwraca aktualny stan urządzenia CSMIO/IP
pause	true false	Zwraca **true** jeśli obróbka jest wstrzymana (pauza)
Axis<X,Y,Z, …>Referenced	true false	Zwraca **true** jeśli dana oś jest poprawnie zbazowana
Axis<X,Y,Z, …>Enabled	true false	Zwraca **true** jeśli dana oś jest załączona w konfiguracji

Założenia przykładu

W poniższym przykładzie wykonamy następujące stylizacje:

- Zmienimy kolor tła i linii osi Y w podglądzie 3D
- Zmienimy kolory nazw osi X, Y, Z oraz uzależnimy je od stanu zbazowania danej osi
- Ściemnimy tło pod niektórymi etykietami
- Sprawimy by przyciski JOG podświetlały się po najechaniu na nie wskaźnikiem myszy
- Sprawimy by przyciski resetujące FRO i SRO podświetlały się po najechaniu na nie wskaźnikiem myszy

Grupowanie

Często zdarza się, że chcemy ustawić podobne właściwości stylu wielu widżetom. Można uprościć sobie pracę, ustawiając w edytorze interfejsu właściwość **group**. Dzięki temu w arkuszu **css** możemy odwoływać się do grupy, a nie do pojedynczych elementów. Ustawiamy teraz widżetom właściwość **group** według poniższej tabeli:

Nazwa („id") widżetu	Nazwa grupy (właściwość group)
btnJogXPos	JogButtons
btnJogXNeg	JogButtons
btnJogYPos	JogButtons
btnJogYNeg	JogButtons
btnJogZPos	JogButtons

btnJogZNeg	JogButtons
btnFro100	Set100PercentButtons
btnSroReset	Set100PercentButtons
lbAxisXName	AxesNameLabels
lbAxisYName	AxesNameLabels
lbAxisZName	AxesNameLabels
lbSelectedTool	DarkerLabels
lbCurrentTool	DarkerLabels
lbToolOffsetNr	DarkerLabels
lbToolDiameter	DarkerLabels
lbFileName	DarkerLabels
lbTimeRemaining	DarkerLabels

Zmiana kolorów w podglądzie 3D

Z katalogu naszego interfejsu otwieramy plik „colors.css". Można użyć dowolnego edytora tekstu, np. **Notatnika** lub **Visual Studio Code**. Zaletą tego drugiego jest to, że koloruje i częściowo analizuje poprawność składni.

W pliku edytujemy kolor tła (**path_view_background**) oraz kolor osi Y (**axisY**), by był lepiej widoczny na nowym tle:

```
[type="path_view_background"] {
  color: #202020;
}
[type="axis_Y"] {
  color: #4646ff;
}
```

Utworzenie nowego pliku arkusza stylu

W katalogu naszego interfejsu zakładamy nowy plik tekstowy o nazwie „widgets.css" - czyli dla naszego przykładu, w systemie Windows, ścieżka będzie następująca:

C:\Program Files\simCNC\screens\ui_example\widgets.css

Program simCNC automatycznie przeszukuje katalog interfejsu w poszukiwaniu plików **css** podczas ładowania ekranu.

Zmiana kolorów etykiet osi i wizualizacja stanu zbazowania

W pliku „widgets.css" dodajemy poniższe instrukcje:

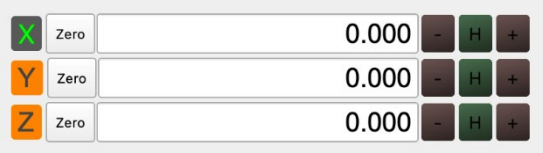
```
[group="AxesNameLabels"] {
  background-color: #e80;
  border-radius: 3px;
  margin: 2px;
  color: #444;
}

[id="lbAxisXName"][axisXReferenced="true"] {
  background-color: #a0000000;
  color: #0f0;
}
[id="lbAxisYName"][axisYReferenced="true"] {
  background-color: #a0000000;
  color: #0c0;
}
[id="lbAxisZName"][axisZReferenced="true"] {
  background-color: #a0000000;
```

Jak widać, odwołujemy się tu do widżetów na dwa sposoby. Najpierw ustalamy wygląd domyślny dla całej grupy (**AxesNameLabels**), natomiast poniżej dla każdej osi osobno wykonywana jest stylizacja warunkowa, jeśli flaga zbazowania osi jest równa **true**.

Modyfikowane właściwości to:

- **background-color** – kolor tła
- **border-radius** – zaokrąglenie rogów
- **margin** – margines
- **color** – kolor tekstu etykiety

Poniżej widzimy efekt, gdy oś X jest zbazowana, a osie Y i Z nie:
*(by zmiany wprowadzane w css były widoczne, należy przeładować ekran: menu **Konfiguracja→Przeładuj ekran**)*



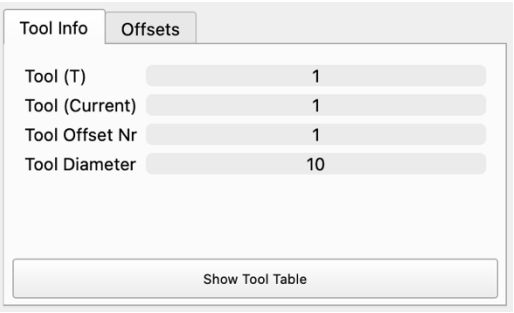
*Ciemniejsze tło pod etykietami z grupy **DarkerLabels***

```
[group="DarkerLabels"] {
  background-color: #10000000;
  border-radius: 5px;
}
```

W tym przypadku ustawiamy dla grupy o nazwie **DarkerLabels** tylko dwa parametry:

- **background-color** – kolor tła
- **border-radius** – zaokrąglenie rogów

Poniżej widać efekt – ciemniejsze tło i zaokrąglone rogi pod etykietami:
*(by zmiany wprowadzane w css były widoczne, należy przeładować ekran: menu **Konfiguracja→Przeładuj ekran**)*

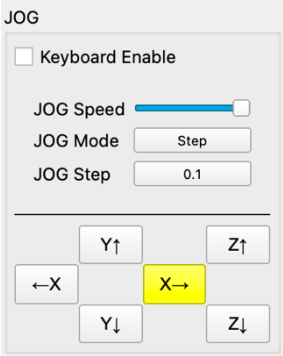


Zmiana koloru (podświetlanie) przycisków JOG

```
[group="JogButtons"]:hover {
  background-color: yellow;
}
```

Jak widać powyżej, modyfikujemy tylko kolor tła (**background-color**), z tym, że robimy to warunkowo. Słowo kluczowe **hover** odpowiada za to, że styl będzie stosowany tylko wtedy, gdy wskaźnik myszy znajduje się nad widżetem. Tworzy to efekt „podświetlania" się przycisku.

Poniżej widać efekt:
*(by zmiany wprowadzane w css były widoczne, należy przeładować ekran: menu **Konfiguracja→Przeładuj ekran**)*

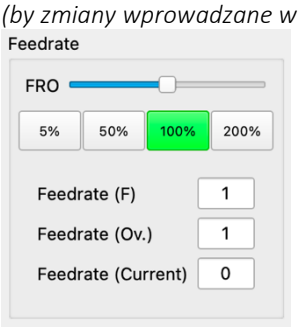


Zmiana koloru (podświetlanie) przycisków resetowania FRO i SRO

```
[group="Set100PercentButtons"]:hover {
  background-color: #0f0;
}
```

Tak samo jak dla przycisków JOG – warunkowo modyfikujemy kolor tła (**background-color**). Słowo kluczowe **hover** odpowiada za to, że styl będzie stosowany tylko wtedy, gdy wskaźnik myszy znajduje się nad widżetem. Tworzy to efekt „podświetlania" się przycisku.

Poniżej widać efekt:
*(by zmiany wprowadzane w css były widoczne, należy przeładować ekran: menu **Konfiguracja→Przeładuj ekran**)*

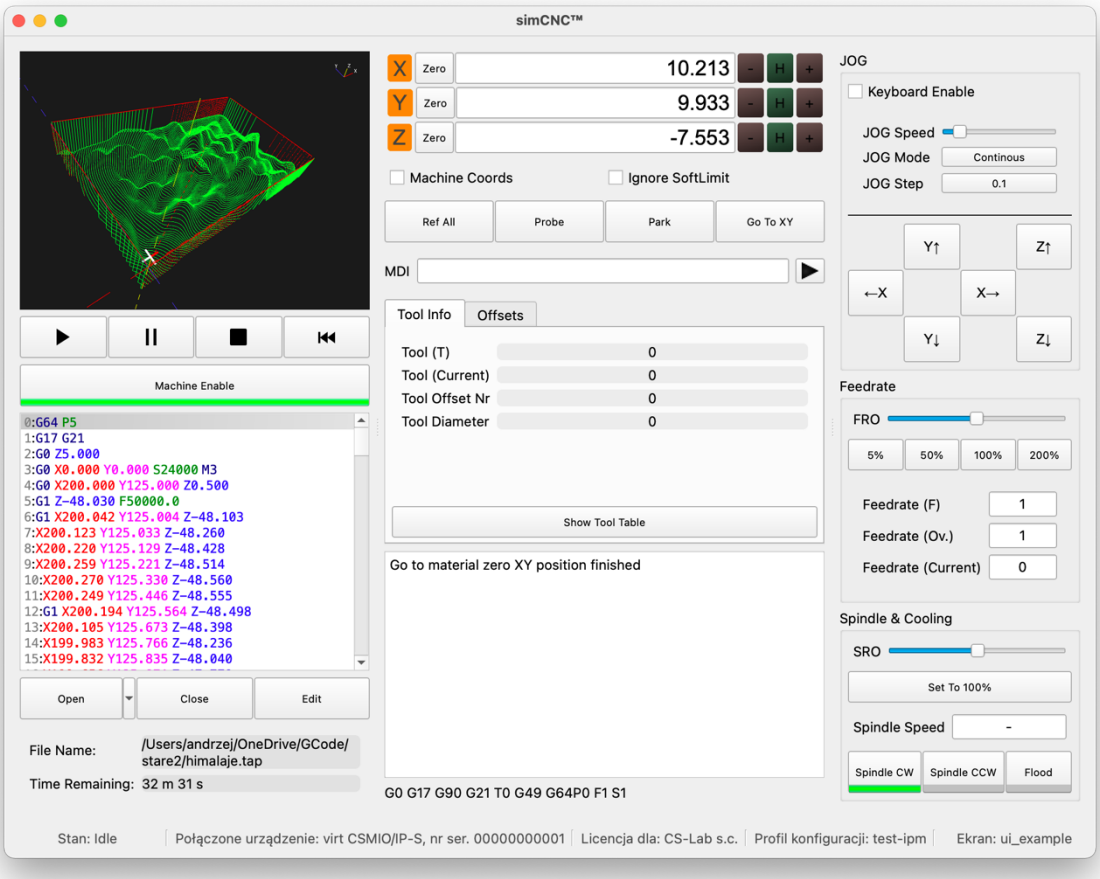


Cała zawartość pliku „widgets.css”

```
[group="AxesNameLabels"] {
  background-color: #e80;
  border-radius: 3px;
  margin: 2px;
  color: #444;
}

[id="lbAxisXName"][axisXReferenced="true"] {
  background-color: #a0000000;
  color: #0f0;
}
[id="lbAxisYName"][axisYReferenced="true"] {
  background-color: #a0000000;
  color: #0c0;
}
[id="lbAxisZName"][axisZReferenced="true"] {
  background-color: #a0000000;
  color: #0c0;
}

[group="DarkerLabels"] {
  background-color: #10000000;
  border-radius: 5px;
}

[group="JogButtons"]:hover {
  background-color: yellow;
}
```

Powyżej cały plik „widgets.css”. Jak widać wystarczy kilkanaście linijek by projekt interfejsu stał się bardziej atrakcyjny wizualnie, a często też wygodniejszy w użyciu i bardziej czytelny dla operatora.

Efekt końcowy i podsumowanie



Tak prezentuje się gotowy projekt. Jak można było się przekonać czytając ten rozdział, wykonanie interfejsu „od zera” wymaga nieco pracy, ale mając taką możliwość, oprogramowanie simCNC da się relatywnie szybko przystosować do wygodnej obsługi wielu rodzajów maszyn i ich osprzętu.

Opisany w tym rozdziale projekt został dołączony do standardowej instalacji simCNC (od wersji 3.410 wzwyż).

Jeśli wykonałeś własny projekt, którym chciałbyś się podzielić z innymi użytkownikami, daj znać na adres office@cs-lab.eu.

Zespół CS-Lab pozdrawia, życzy owocnej pracy i świetnych efektów ☺