MatDeck GUI Functions

MatDeck GUI Widgets can be used in MD Documents, MatDeck Script or Python via the MD Python Library. Widgets can be embedded into MatDeck documents or can be used to create programs and independent applications. Widgets are available on all MatDeck interfaces: in documents, plug-ins and in executable applications compiled from the MatDeck Script and Python.

Widget functions are located in the GUI functions group. Some of these functions are functions for creating widgets and some are helper functions for setting widget size, position, value etc. These functions are the exactly he same for MatDeck Script or Python.

Creating GUI widgets

Widgets are parent-child based which means that you can add new widget on the top of existing widget as its child or if you use a 0 for parent, new widget will be top level widget.

Widget functions

Button function

First argument is parent widget (0 mean there is no parent widget). The second one is text for the button.

```
b:=button(0 , "button")
```

Check Box function

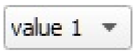
Arguments are parent widget, text and check box state

```
ch:=check_box(0 , "check box" , true)
```

Combo Box function

Arguments are parent widget, and vector of strings

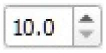
$$cm := \text{combo\_box}\left(0\ ,\ \begin{bmatrix}\text{"value 1"}\\ \text{"value 2"}\end{bmatrix}\right)$$



Double Spin Box function

Arguments are is parent widget, minimum value as real number, maximum value as real number, step as real number and current value as real number

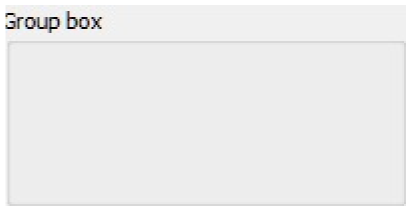
$$dsb := \text{double\_spin\_box}(0\ ,\ 10\ ,\ 10.5\ ,\ 0.1\ ,\ 10)$$



Group Box function

Arguments are parent widget and caption string

$$gb := \text{group\_box}(0\ ,\ \text{"Group box"})$$



Label function

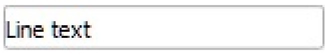
Arguments are parent widget and caption string

$$lb := \text{label}(0\ ,\ \text{"Label"})$$

Line Text Box function

Arguments are parent widget and text

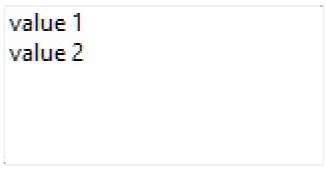
```
ltb := line_text_box(0 , "Line text")
```



List Box function

Arguments are parent widget and caption string

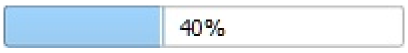
$$li := \mathrm{list\_box}\left(0\ ,\ \begin{bmatrix}\text{"value 1"}\\ \text{"value 2"}\end{bmatrix}\right)$$



Progress bar function

Arguments are parent widget, minimum value, maximum value, string current value and orientation "horizontal" or "vertical"

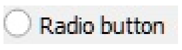
```
pb := progress_bar(0 , 0 , 10 , 4 , "horizontal")
```



Radio button function

Arguments are parent widget, caption string and boolean value

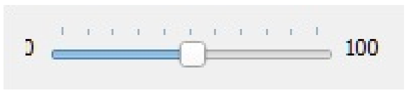
```
rb := radio_button(0 , "Radio button" , false)
```



Slider function

Arguments are parent widget, minimum value, maximum value, step, current value, orientation "h" or "v" and show scale or not

```
sl:=slider(0 , 0 , 100 , 1 , 50 , "h" , true)
```



Spin Box function

Arguments are parent widget, minimum, maximum, step and value

```
sb:=spin_box(0 , 0 , 100 , 50 , 0)
```

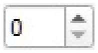
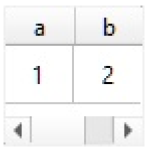


Table Widget function

Arguments are parent widget and table or matrix

tbl:=table_widget(0 , $\begin{array}{|c|c|}\hline a & b \\ \hline 1 & 2 \\ \hline\end{array}$)

```
set_size(tbl , 70 , 70)
```

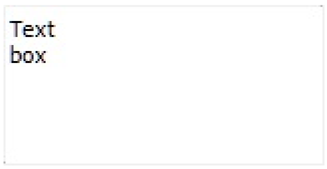


a	b
1	2

Text Box function

Arguments are parent widget and text

```
tb:=text_box(0 , "Text\nbox")
```


Text
box

Displaying widgets

When a widget is created it is not visible and it exists only in the MatDeck memory. You can choose to embed it into a document or you can choose to show it as an independent object on the screen. In the first case MatDeck provides a function embed widget() and in the second case you should use function show(). Note: once you embed a widget you can not show it latter as an independent object with show().
Also see: set visible().

Setting widgets size and position

After the widget is created you can adjust its size and position. Functions for this are set pos() and set size(). If widget is a child widget set pos() refers to its position on the parent.
Also see: pos x(), pos y(), width() and height().

Settings and getting the widgets value

Widgets are created with its default values (usually it is the last argument in the function for creating widgets) but you can change it later. For example if you create a check box with a unchecked state you can change it to a checked state later. The function for setting widget values is set widget value(). You can change values for the following widgets: button, spin box, double spin box, radio button, check box, list box, combo box, line text box, text box, slider, progress bar and label.
Similarly you can get a widget value. For example a selected line in the combo box, or selected numbers from the spin box. Types of values are the same as in the function for creating a widget. For example a combo box is created with a vector of strings for all lines and its value will be one string from that vector. The function for getting widget value is widget value(). You can get values for the following widgets: button, spin box, double spin box, radio button, check box, list box, combo box, line text box, text box, slider, progress bar, label and table widget.

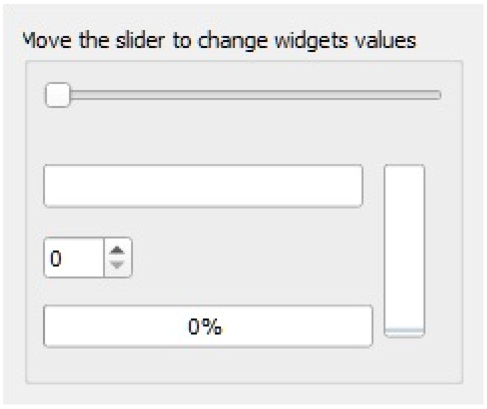
Capturing widgets events

Some of the widgets (not all) have dedicated events. When a button is clicked it generates button event. The widgets events are generated when user interact with GUI objects, for example change the text in the text box or change the selected line in the combo or list box.
In addition you can capture events and use it to execute your custom functions. The function for connecting widget events with your function is on event().The first argument is widget variable and second is a function. Widgets: button, spin box, double spin box, radio button, check box, list box, combo box, line text box, text box and slider.Note: custom callback functions should be designed without function arguments.

Example

```
w:=widget(0 , "GUI example")
set_size(w , 240 , 200)
g:=group_box(w , "Move the slider to change widgets values")
set_pos(g , 10 , 10)
set_size(g , 220 , 180)

s:=slider(g , 0 , 100 , 1 , 0 , "horizontal" , false)
set_pos(s , 10 , 25)
set_size(s , 200 , 22)

ph:=progress_bar(g , 0 , 100 , 0 , "vertical")
set_pos(ph , 180 , 70)

on_event(s , changeValues())

t:=line_text_box(g , "")
set_pos(t , 10 , 70)
sp:=spin_box(g , 0 , 100 , 1 , 0)
set_pos(sp , 10 , 105)
p:=progress_bar(g , 0 , 100 , 0 , "horizontal")
set_pos(p , 10 , 140)
set_size(p , 165 , 22)

changeValues( )
{
  v:=widget_value(s)
  set_widget_value(t , to_string(v))
  set_widget_value(sp , v)
  set_widget_value(p , v)
  set_widget_value(ph , v)
}
```

Embedded GUI (used function is embed widget(w))


Move the slider to change widgets values
0
0%