

Custom Tkinter GUI Designer

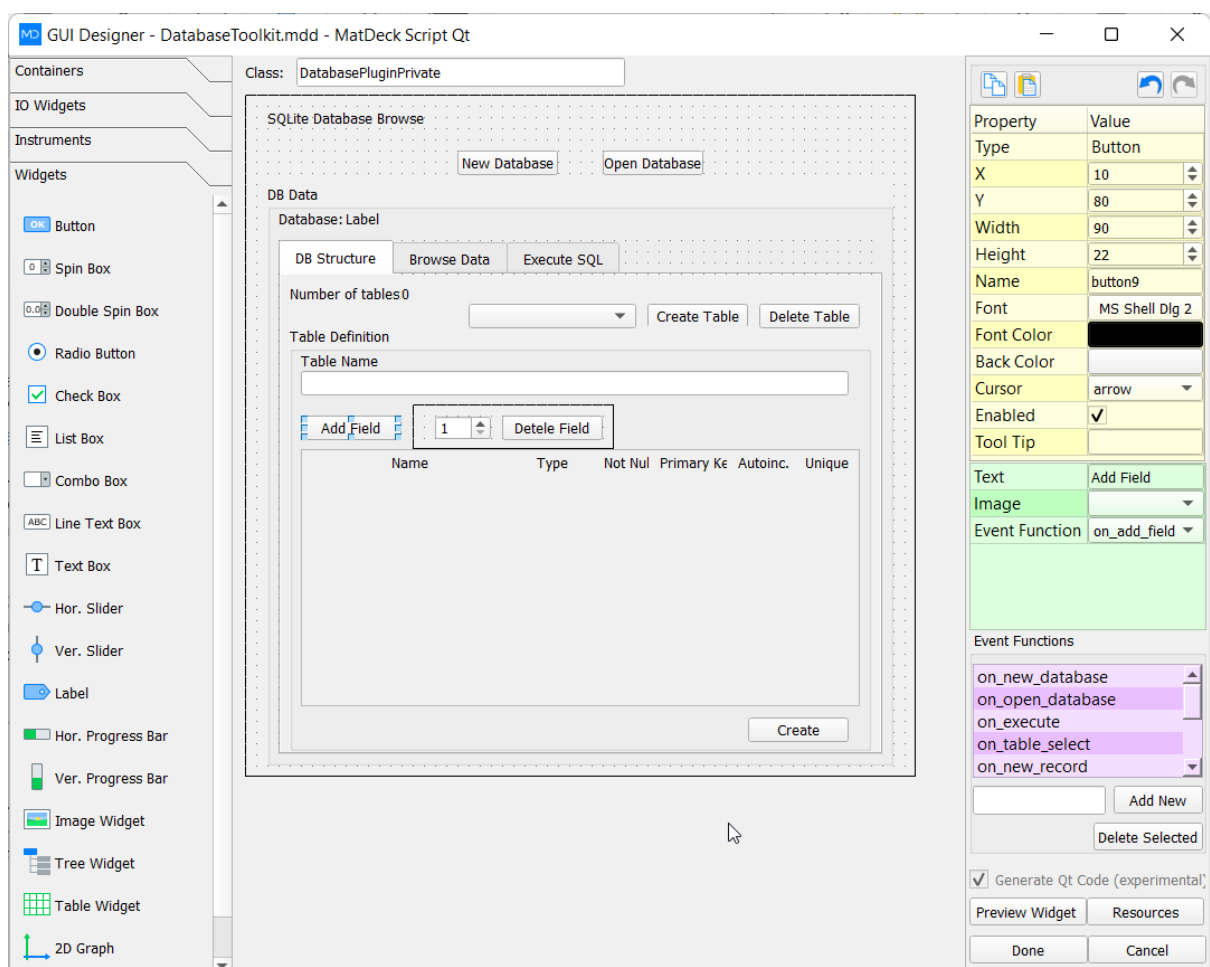
Introduction

The Unlimited Custom Tkinter GUI Designer is available in MD Python Designer, Engineering Designer, Visionary Deck, MatDeck Student, MatDeck Academic, MatDeck Home and MatDeck. Its limited form is available in Lite MD Python Designer.

All MD Products offer six GUI Designers: Custom Tkinter, Kivy, Custom Custom Tkinter, PySide2, MD Python and MatDeck Script except MatDeck Free. MD Python Designer and Higher MD Products all have unlimited access to all GUI Designers whereas Lite MD Python Designer is limited to only a few elements and MatDeck Free does not have any MD GUI Designer.

MD GUI Designers utilize drag and drop visual aids to create a fully functioning GUI with a sleek and modern interface, the GUI widgets and components are all automatically generated leaving no work to the user. MD Python Designer comes with all necessary languages, libraries, modules and extensions already installed, meaning that no extra setup is needed and the user can start creating straight-away.

MD Python Designer also comes with a specialized IDE and GUI Designer for Custom Tkinter. As mentioned before all the necessary libraries and modules come preinstalled.



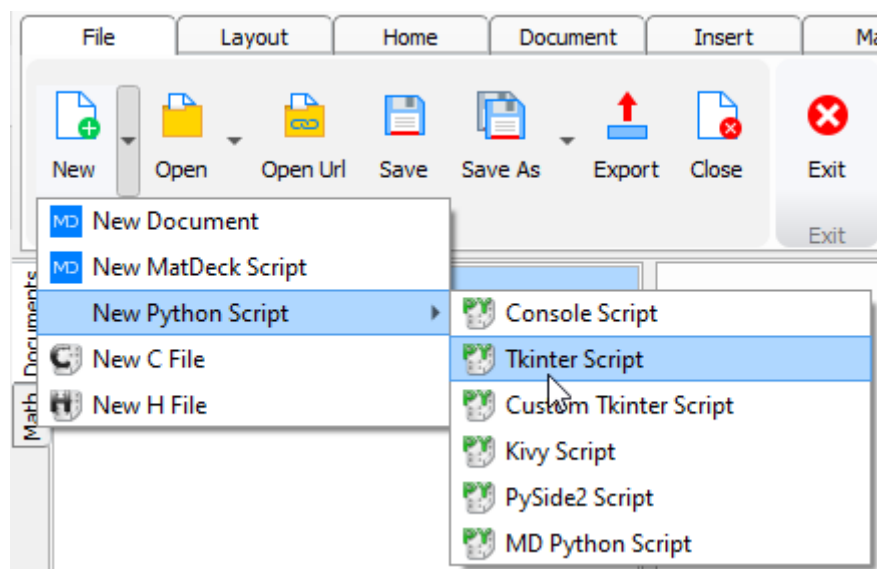
This is the what a MD GUI Designer looks like. As you can see the from itself, contains no code and was created solely through the use of dragging and dropping GUI elements into the work area.

Please Note: Our GUI Designers generate 2 files, one file is anormal Python file and the other file is used by our GUI Designer to save and remember what your GUI looked like in the GUI Designer, it is a .mpy file and it cannot be opened like a Python file.

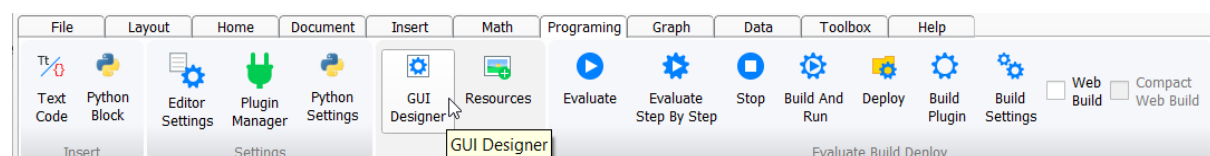
Name	Date modified	Type	Size
Example.mpy	14/11/2022 18:52	MPY File	3 KB
Example.py	14/11/2022 18:52	Python File	3 KB

Opening a MD GUI Designer

The first thing you need to do to open the Custom Tkinter GUI Designer is to select the Custom Tkinter Script. This is done by hovering over the small triangle next to the New icon.

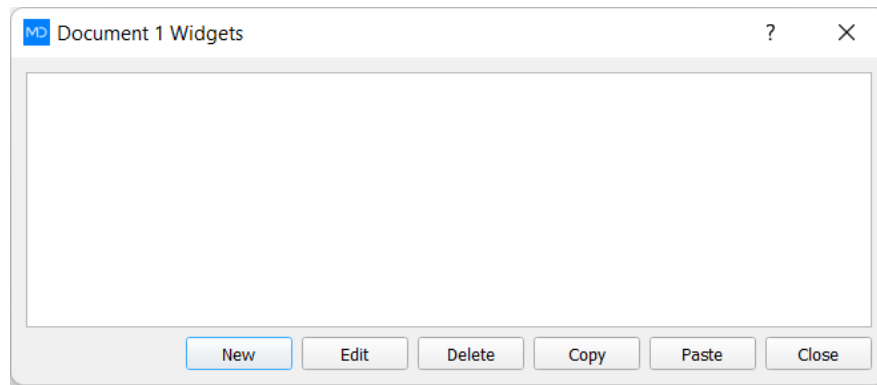


To open the Custom Tkinter GUI Designer, go to Programing tab and press the 'GUI Designer' button (Picture 1).



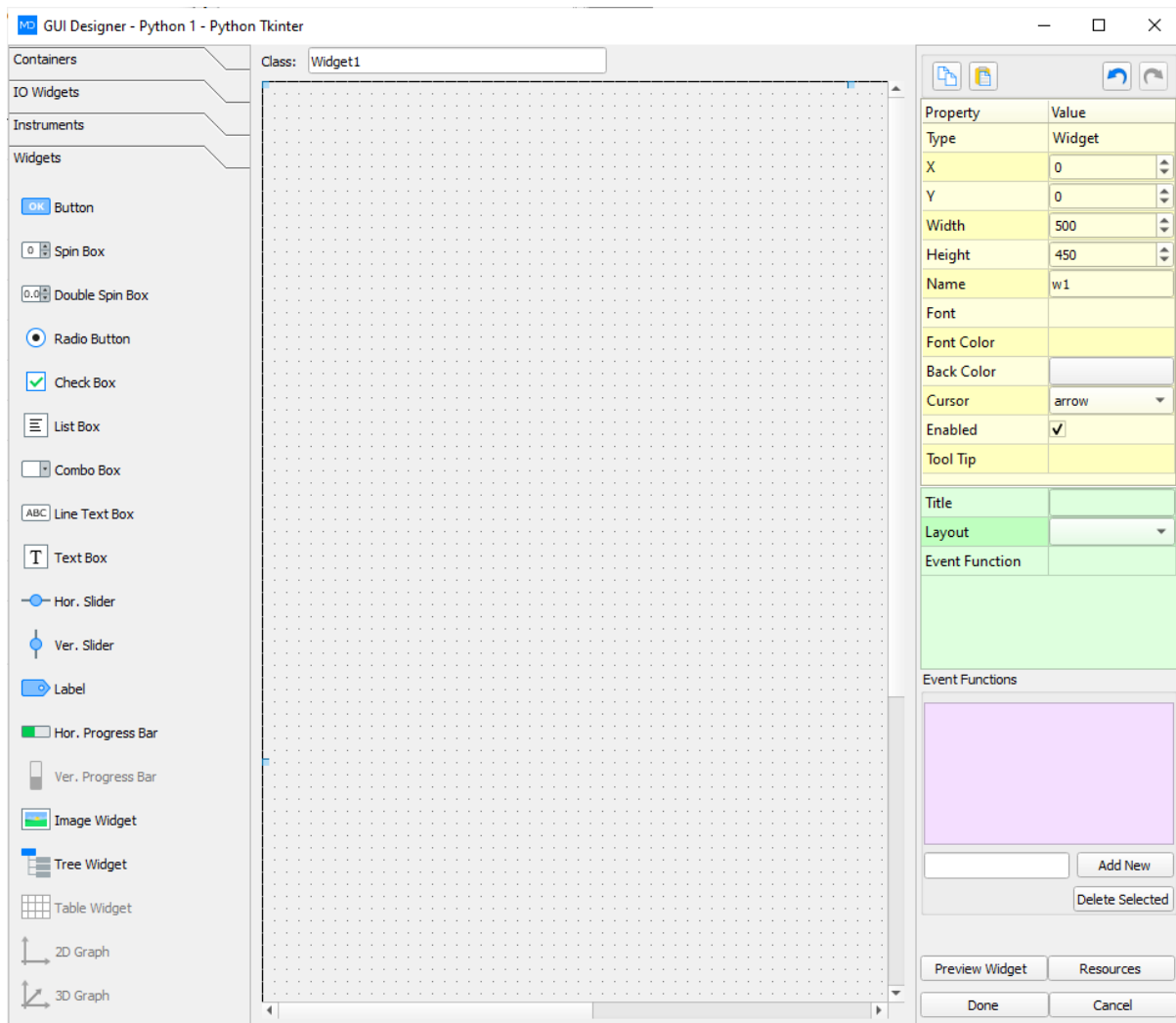
Picture 1: Widgets button

A new window will open, which will contain a list of all the widgets that have been created in the current Custom Tkinter Script (Picture 2). From this window you can create a new widget, edit or delete the existing one. To open the Custom Tkinter GUI Designer, you will need to click the New button, the edit button will only open the Custom Tkinter GUI Designer if there is a widget beforehand.



Picture 2: Document widgets editor

To create a GUI, simply, press 'New' and an empty Custom Tkinter GUI Designer tab will open. When doing so, make sure that you haven't selected any widgets that may be present in the window. To edit or delete a pre-existing widget, select the said widget first and then press on the suitable button at the bottom of the tab. The default look of an empty Custom Tkinter GUI Designer is shown on picture below.

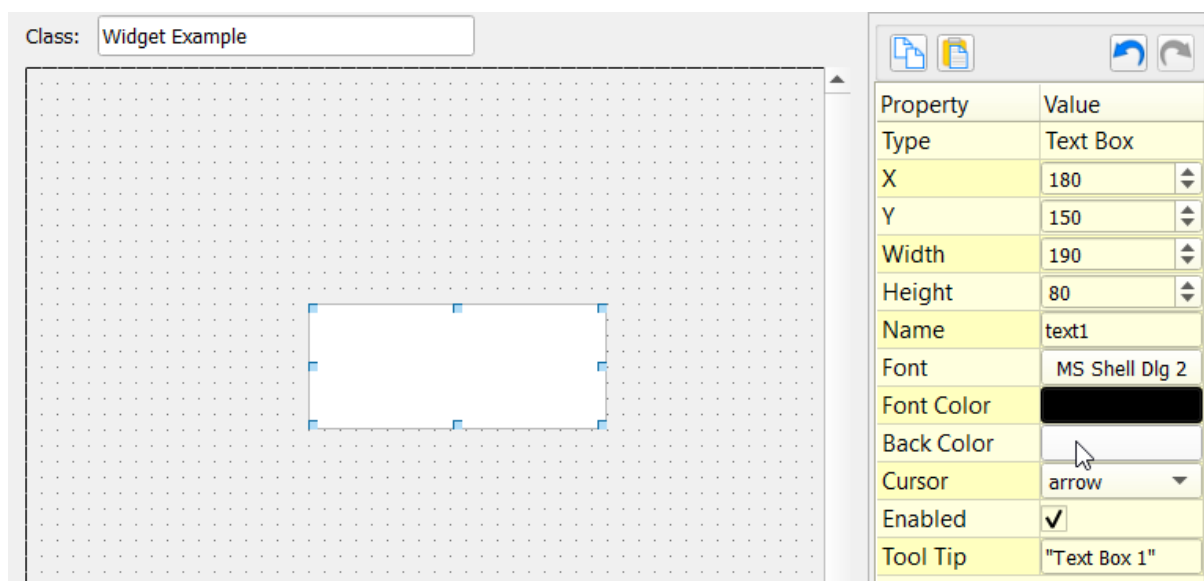


Picture 3: GUI Designer

On the left side of the Custom Tkinter GUI Designer are all the GUI elements; in the middle of the GUI Designer is the working area, where widgets and GUI elements can be placed. To place one of the elements just click on the preferred GUI element and select where on the working area you would like to place it.

For example, to add a Text Box to the GUI. You will need to click the Text Box button on the left hand side and then click anywhere on the working area.

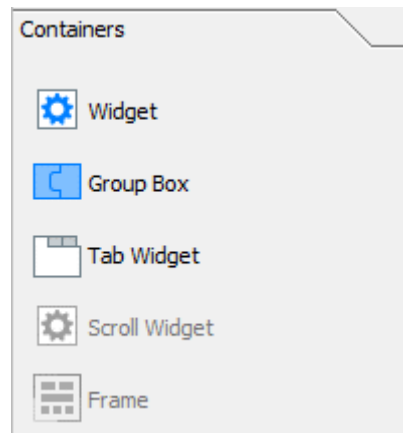
On the right-hand side of the Custom Tkinter GUI Designer, the properties of the selected element are displayed and these properties can be modified here (Picture 4). In the example below, we have also edited the name of the widget to 'Widget Example'. To do this, you just have to write your preferred name in the field above the main working area.



Picture 4: GUI Element Property

Please note that to actually connect or interact between GUI elements, you have to add the necessary code for the interaction itself. To do so, edit the generated code. We focus on saving you time, money and effort on creating the GUI and try to simplify any coding required, however your imagination is the limited to what your code will do and so we do not want to ruin your imagination.

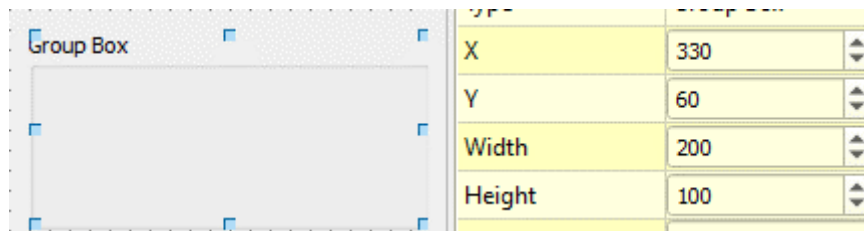
If we open the Containers menu, from the top left part of the Custom Tkinter GUI Designer, a new list of GUI elements will appear. These elements are called containers and can be placed on the main widget area. They act as carriers for other GUI elements.



Picture 5: Containers

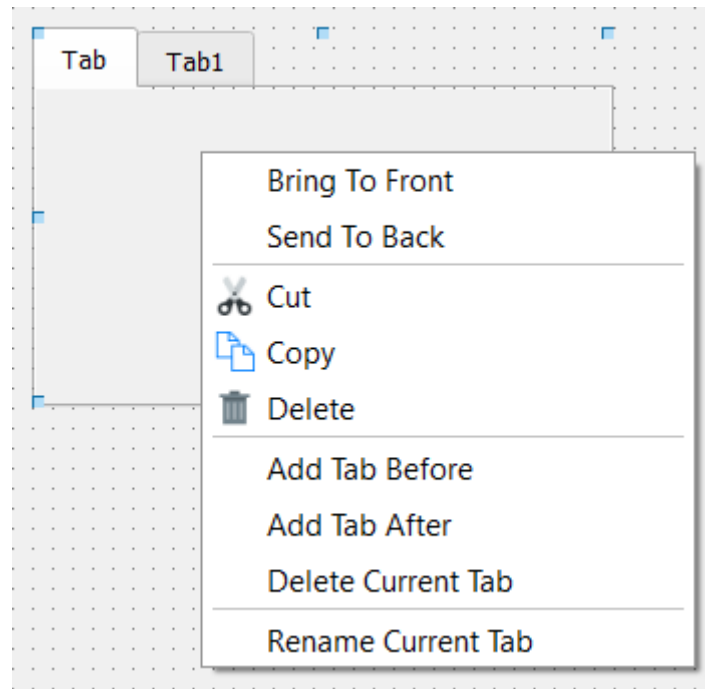
To insert any of these containers on the main form, use the same method as any other GUI element. First, select the preferred container and then, select where on the working area you want to place it.

When a container or element is selected, it can be resized by moving the blue squares around it or it can be resized by changing its Width and Height values from the Property table , the Property Table is found on the right-hand side.



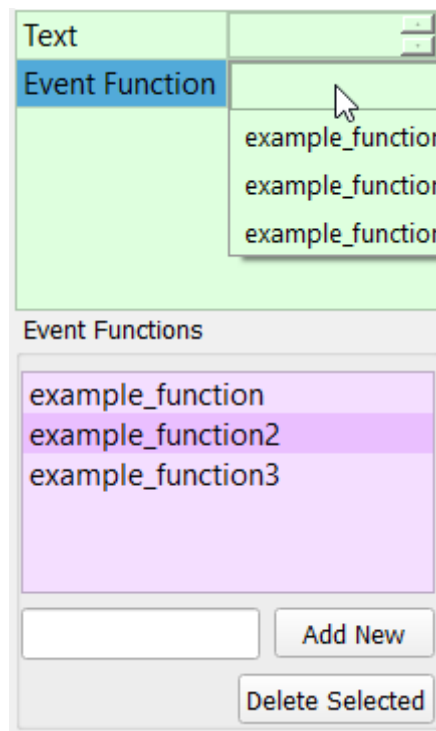
Picture 6: Resizing of object

If you wish to preview your widget and check on your progress at any point during your work you can do so by pressing the 'Preview Widget' button in the bottom right corner. It will output a new interactive window where you can check and use your widget.



When a Tab widget is used, we can use the context menu to rename the current tab or to insert a new tab, to open the context menu, right click on the GUI widget.

You can add event functions to selected GUI elements. The Event Function list is empty on all newly created Widgets; we have to define a new event function by using the Event Functions part of the GUI Designer, it is color-coded in green and purple. To add an event function, input the name of the it in the empty field found in the Event Function area and press 'Add New'. After creating a new Event Function, it will appear in the Event Function list for the selected GUI element (Picture 5).

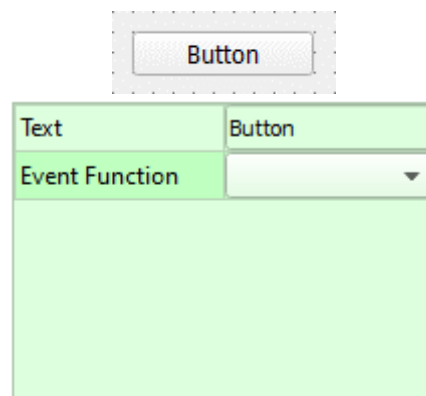


Picture 7: Event Functions

The Event Functions container is a global container which contains all the created event functions for the Widget that we are currently creating/editing. In the field named 'Event Function', you can select one of the defined functions by choosing one of the options that appears in the drop down list when clicked.

The green window showed above is used to edit certain aspects of GUI elements, primarily the event function, and it is not available to all elements. Meaning, not all GUI elements will have the same options present and some elements will have options only specific to that element and not present in other elements. Below, all examples of this are explained.


Button



The image shows a button widget with the text "Button". Below it is a configuration panel with a light green background. The panel has two columns. The first column contains the labels "Text" and "Event Function". The second column contains the value "Button" and a dropdown menu. Below these fields is a large empty green area.

When the Button GUI element is inserted into the working area, the green box above will be visible on the left hand side.. The 'Text' field can be used to change and set the visible label on the button. Additionally, you can choose the Event Function that will be used on the Button's event.

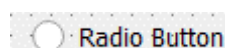
Spin box



The image shows a spin box widget displaying the value "0". Below it is a configuration panel with a light green background. The panel has two columns. The first column contains the labels "Minimum", "Maximum", "Step", "Value", and "Event Function". The second column contains the values "0", "99", "1", "0", and a dropdown menu. Below these fields is a large empty green area.

When a spin box GUI element is inserted, element specific settings that can be changed will appear. For example, the Minimum and Maximum value of the spin box can be set by using the 'Minimum' and 'Maximum' fields respectively. The step at which the spin box's value increments by can also be set by using the 'Step' field. The starting value of the spin box can be set by entering your chosen value in the 'Value' field. Lastly, an Event Function can be added via the Event Function Combo Box.

Radio button



The image shows a radio button widget with the text "Radio Button".

Text	Radio Button
Value	<input type="checkbox"/>
Event Function	<input type="text"/>

When a Radio button GUI element is inserted, the settings above will be visible. The text of the radio button can be altered by writing in the 'Text field. By default, the label used is 'Radio Button'. The 'Value' attribute can be used to define the default value of the selected radio button (whether the radio button is checked or not). On containers/elements that contain several radio buttons; only one of them can be checked at a time. The Event Function will occur when the Radio Button is checked.

List box

Value	<input type="text"/>
Event Function	<input type="text"/>

When a List box GUI element is inserted, the following options will be visible. Using the 'Value' attribute, the default value of the list box can be defined. Event Functions will occur when one of the list boxes containing rows is selected.

Minimum	0
Maximum	99
Step	1
Value	0
Orientation	horizontal
Event Function	<input type="text"/>

- Horizontal slider



When a Slider GUI element is inserted, the following options will appear. The range of values the slider can reach can be set using the 'Minimum' and 'Maximum' fields respectively. The step of the slider which defines how much the slider increments by is set using the 'Step' attribute and the

default (starting) value of slider is set using the 'Value' attribute. The 'Orientation' drop down menu is used to change the slider's orientation. Only horizontal and vertical orientations can be used. If set, The Event Function is used when slider value is changed.

Text

Label

Label

- Label

When a Label is inserted, they only have one customizable setting. Only the Label's visible text can be edited. This is done by writing in your preferred text in the 'Text' field at the top.

Minimum

0

Maximum

100

Value

25

Orientation

horizontal

25%

5%

Progress Bars

When a Progress bar GUI element is inserted, the following attributes can be customized using the available settings. Firstly, the range of allowed values can be set using the 'Minimum' and 'Maximum' attributes respectively. The default (starting) value of the Progress bar can be set by editing the 'Value' field. The orientation of the Progress bar can be changed by selecting an option in the drop down menu of the 'Orientation' option. Only horizontal and vertical orientations can be used on the Progress bar.

Image

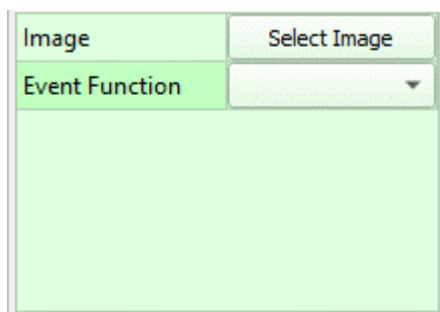
Select Image

Event Function

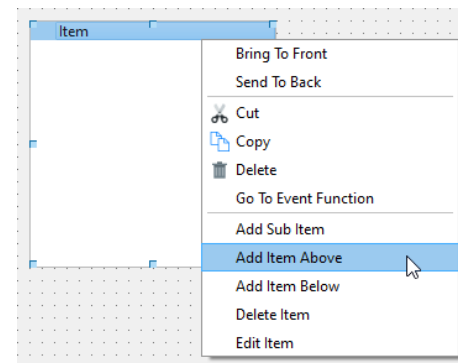
- Image widget

When an Image GUI element is inserted, the picture used can be set by using the 'Image' field at the top. The Event Function will occur when the image is clicked.

When an



- Tree widget



When a Tree Widget GUI element is inserted, the picture used can be set by using the 'Image' field at the top. The Event Function will occur when an item of the Tree view is clicked. To add a Item to the Tree view, right-click on it and select Add Item. To add a sub item, right-click on the item you would like to add it to and select the Add Sub Item. To rename any Item, right-click on it and then choose the Edit Item option.

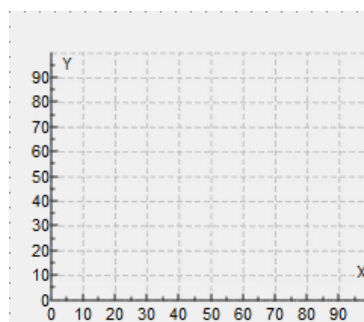
Tables, 2D and 3D graphs don't have any settings at all. All properties of those GUI elements can be customized by using the settings present in their context menus.

Using Table, 2D and 3D Graphs

Graphs and Tables can not be set in the GUI Designer, they can be customized but their value can only be changed directly in the code.

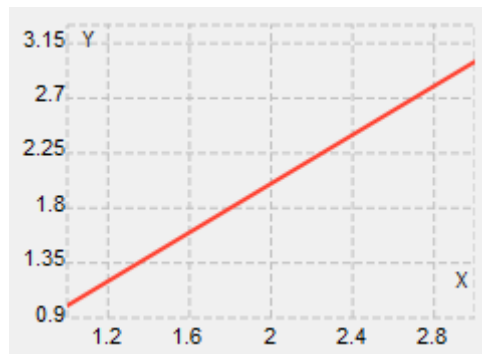
2D Graphs

To set the widget value of a graph you'll need to use an array.



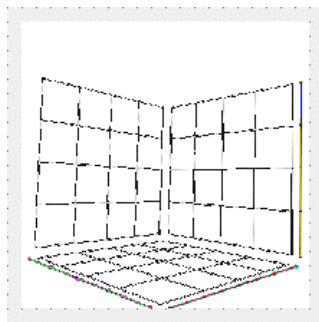
```
30 set_widget_value(graph2d1,[1, 2, 3;1, 2, 3])
31 set_widget_value(self.graph2d1,[[1, 2, 3], [1, 2, 3]])
```

The first line is in MatDeck script with the second line in Python using the MD Python Library. Here is what it looks like with its value assigned.



3D Graphs

To set the widget value of a graph you'll need to a MD Function for 3D Graphs, like the one you can see below. Here is what it looks like with its value assigned.

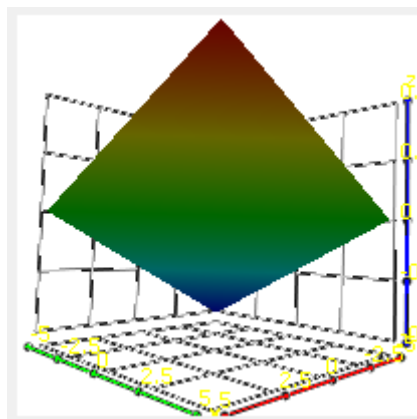


```

39 a := surface3d(sin(x+y), x, -5, 5, 50, y, -5, 5, 50)
40 set_widget_value(graph3d1, a)
41 a = surface3d(sin(x+y), x, -5, 5, 50, y, -5, 5, 50)
42 set_widget_value(self.graph3d1, a)

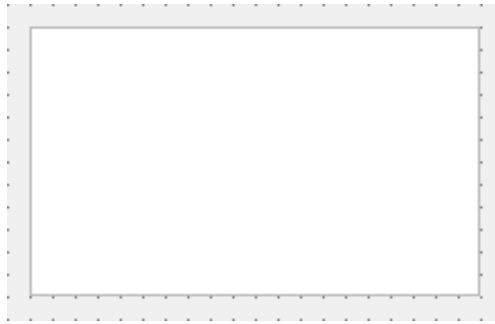
```

The first line is in MatDeck script with the second line in Python using the MD Python Library. Here is what it looks like with its value assigned.



Tables

To set the widget value of a Table you'll need to use an array, just like a 2D graph. The column will automatically be assigned, you can change this using the `table_create()` function.



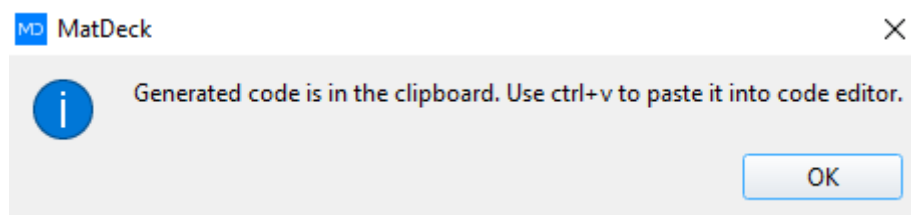
```
30 set_widget_value(table1,[1, 2, 3;1, 2, 3])  
31 set_widget_value(self.table1,[[1, 2, 3], [1, 2, 3]])
```

The first line is in MatDeck script with the second line in Python using the MD Python Library. Here is what it looks like with its value assigned.

1	2	3	4
Column1	1	2	3
Column2	1	2	3

Finishing Up

Once you are finished working on your widget design, you can press the 'Done' button and the designer will generate the code for the widget and paste it into the script you are working in, if you are using a MatDeck Document the code will be copied into your clipboard and can then be pasted anywhere. If you have created a new widget and generated the code by pressing 'Done' for the first time, the notification window (Picture 8) will open. Once the 'Done' button has been pressed for the first time, any new code is generated and it will be updated without showing this notification.

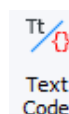


Picture 8: Notification Window

As mentioned before, when you click done in a script, your code will automatically be generated into the script.

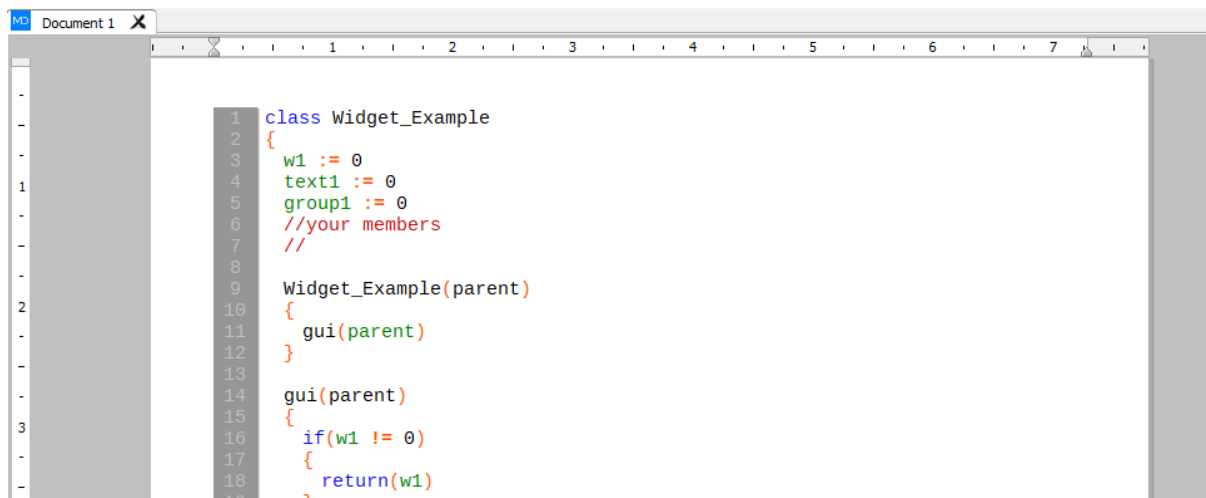
Using GUI Designer Code in a Document

To place the generated code on a MatDeck document, we can place the cursor on the preferred position and turn on Code mode. We can turn this mode on from Programming tab using:



or by using the *Ctrl. + i* key combination. When Text Code mode is inserted, paste the code from the clipboard on document.

The generated code is copied to your clipboard as soon as you press 'OK'. To paste the generated code into a MatDeck script, you need to first open a new script file. Do this by opening the drop down list on the 'new' page button and pressing on the 'new script' option. In the new script file, you can paste your generated code and use it as you wish.



Picture 9: Widget Code

For every Event function we have created on the GUI Designer, the function will appear with an empty body for the user to enter any code they want.

```
30
31 Example_function()
32 {
33
34 }
35 }
36
37 a := Widget_Example(0)
38 show(a.w1)
```

Picture 10: Event Function In Code

At the end of code, a constructor for the created code is added to allows the program to be shown. The purpose of this is to create the widget as a stand-alone application it is evaluated.

Notice: The preferred order of creating/changing widget designs and generating or updating code originates from the GUI Designer. If you change code directly and then, open and save some changes in the GUI Designer afterwards, all changes that were made directly in the code will be lost.

Interacting with generated GUI code with MatDeck Script

The GUI Designer generates code which is used to create the visual aspects of the GUI. This means the generated code will output the GUI as it was designed by the users. The position of the GUI elements, their titles and event functions will all be coded as they are in the GUI Designer. In short, the code created by the GUI designer is for creating the individual GUI elements and their properties, any interactions or changes need to be added by the user.

Functions and code that can be used on GUIs are all included and explained in another manual. However, in this segment of the manual we will cover two important functions that are most commonly used. These functions are also included in the example at the end.

[Retrieving the value of a widget / get\(\)](#)

To retrieve and store the value of a widget present in a GUI, the function `get()` needs to be used. It has only one argument, the name of the widget.

```
1 A := widget_value(ExampleWidget)
```

In the example above, the function retrieves the value of the widget stored in the variable “`ExampleWidget`” and stores the value in the variable “`A`”.

[Connecting widget values with other widgets/ set_widget_value\(\)](#)

To display the value of a widget on a separate widget, the function `set_widget_value()` needs to be used. It has only two arguments, the widget name where the value will be set and the value that will be displayed on that widget.

```
1 set_widget_value(instrument1, widget_value(vslider1))
```

In the example above, the first argument (the widget where the value will be applied) is a circular instrument gauge. The second argument is the value of the vertical slider present in the same GUI application. The value of the slider is retrieved using the function `widget_value()`. Once the function has been used, the value of the instrument will be that of the slider.

When altering or interacting with the value of a GUI element, it should be done in its event function. In this case, when interacting with the value of the slider, the function `set_widget_value()` is used in the slider’s event function.

Below is an example of this.

```
1 SliderEventFunction()  
2 {  
3     set_widget_value(instrument1, widget_value(vslider1))  
4 }
```

Executing and Outputting GUIs

Once GUI designs are completed, the generated code is copied to the clipboard or pasted directly into the script. This generated code can be used in different formats which include: MatDeck documents, Python Scripts, and MatDeck Script.

GUIs can be outputted using the following methods:

- Evaluation of generated code
- Embedding the widget

Evaluation of generated code

GUI applications can be executed by evaluating the generated code. By default, the GUI application will appear as a standalone window separate to the MatDeck document in which it was evaluated. To execute generated code in MatDeck, first, the code must be placed in the appropriate format. This means the generated code can be copied to a programming block, MatDeck Script and Python Script.

Once the code is placed in the appropriate location, press the 'Evaluate' icon to evaluate the current code. Evaluate is located in the 'Programming' toolbar.

