

Tkinter GUI Designer

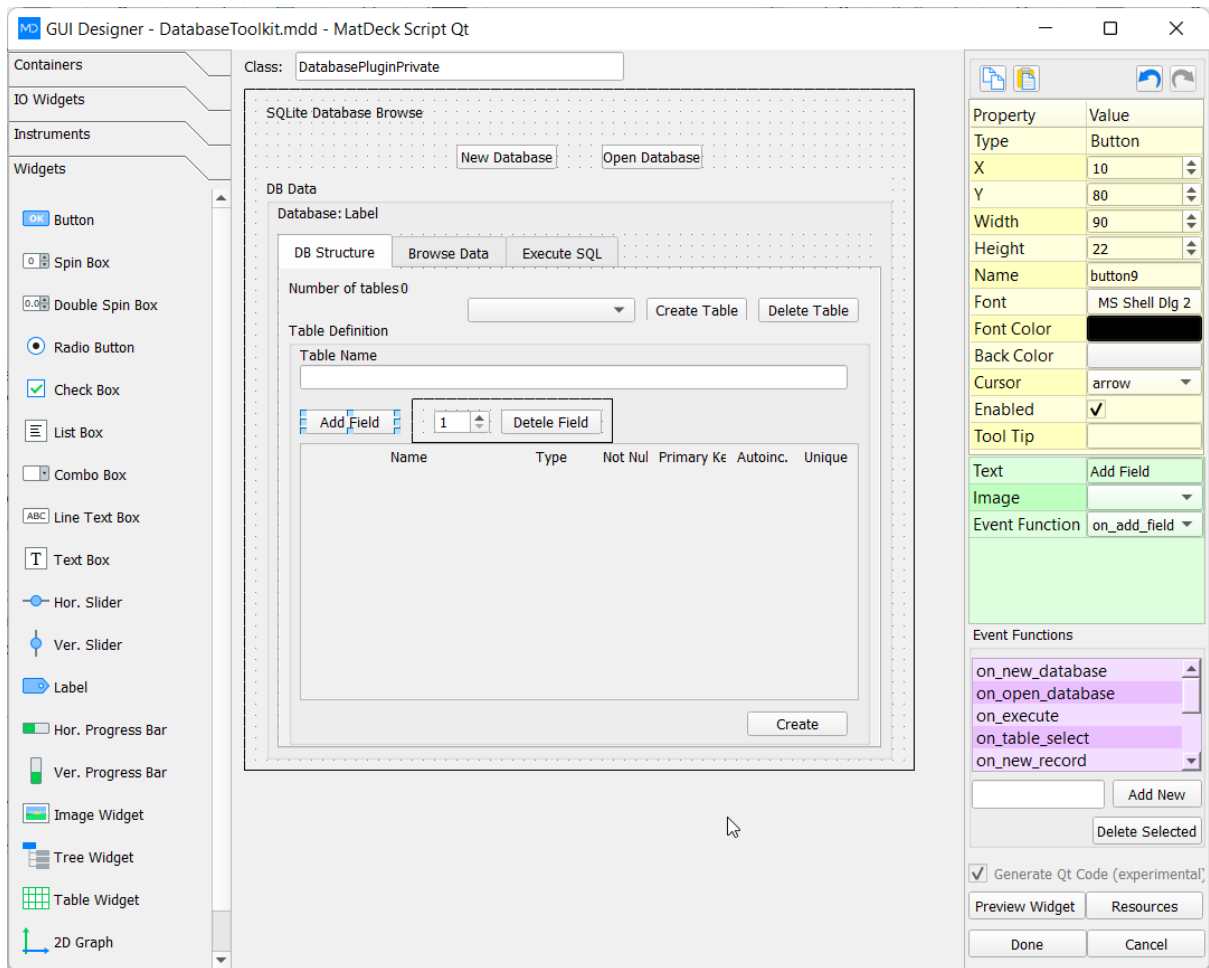
Introduction

The Unlimited 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: Tkinter, Kivy, 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 Tkinter. As mentioned before all the necessary libraries and modules come preinstalled.



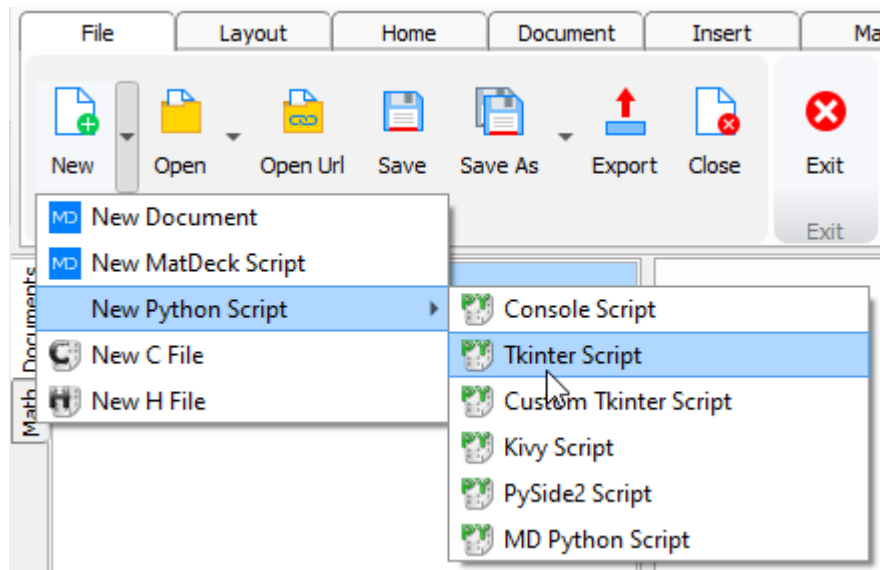
This is the what a MD GUI Designer looks like. As you can see the from itself, contains no code and was created solely through the use of dragging and dropping GUI elements into the work area.

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.

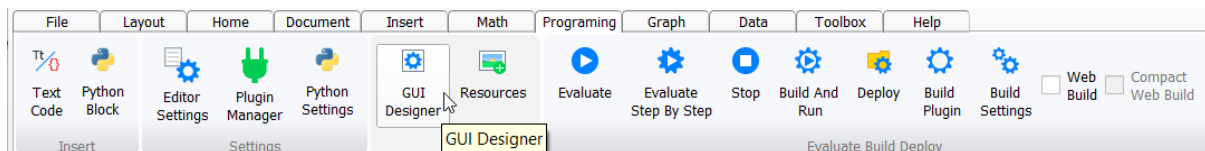
Name	Date modified	Type	Size
Example.mpy	14/11/2022 18:52	MPY File	3 KB
Example.py	14/11/2022 18:52	Python File	3 KB

Opening a MD GUI Designer

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

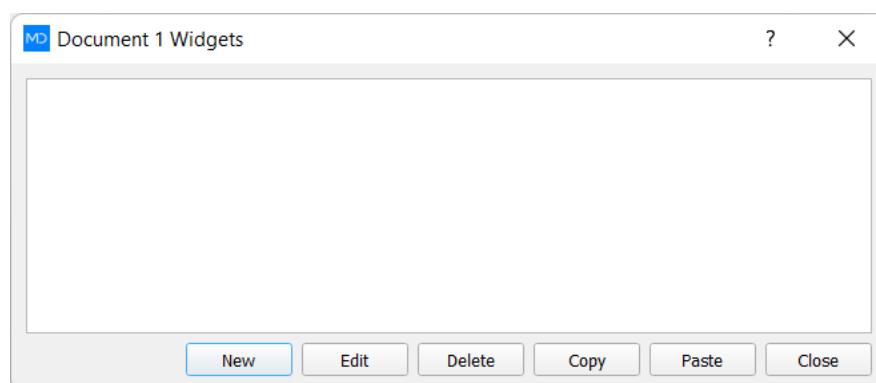


To open the Tkinter GUI Designer, go to Programming tab and press the 'GUI Designer' button (Picture 1).



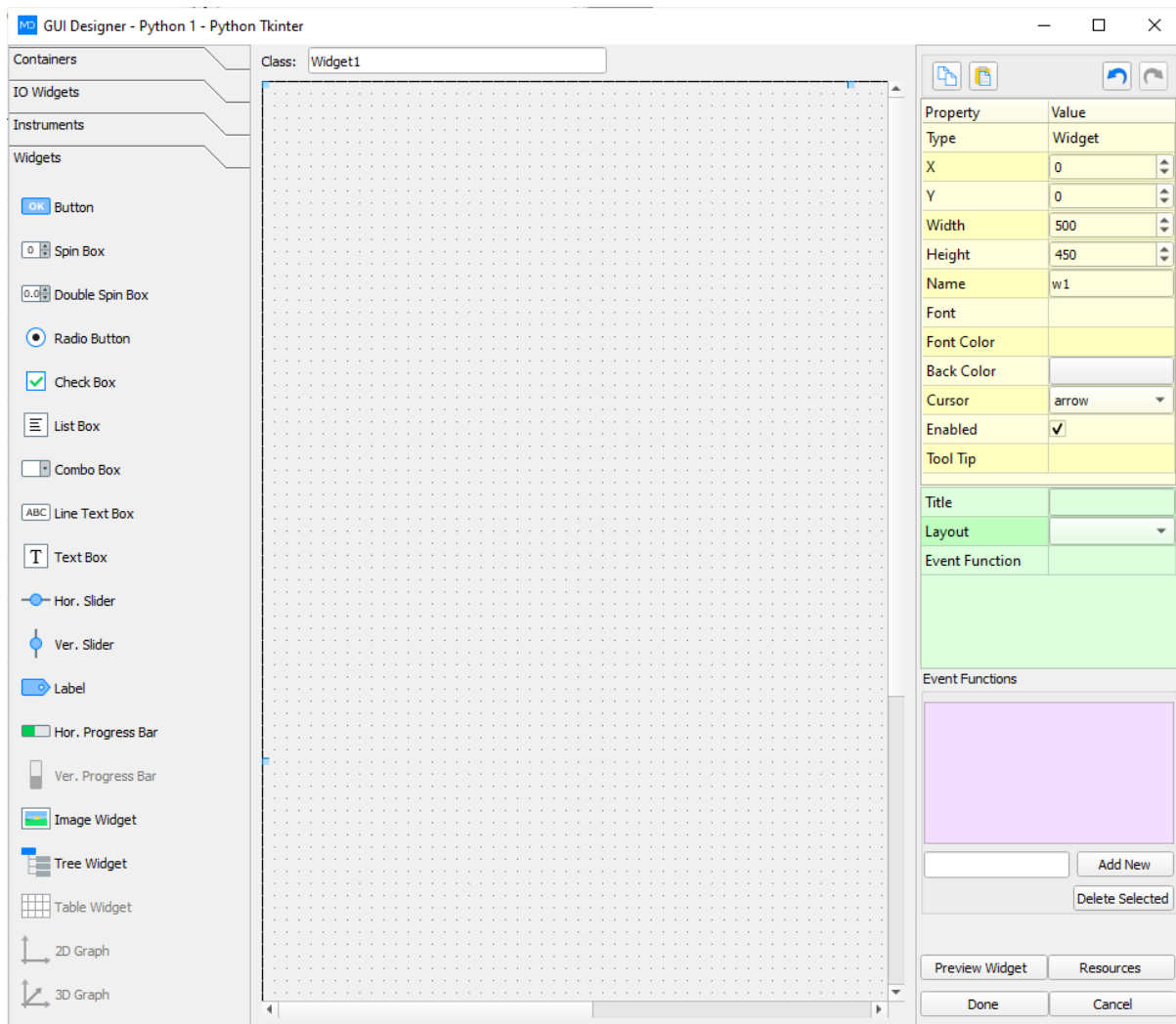
Picture 1: Widgets button

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



Picture 2: Document widgets editor

To create a GUI, simply, press 'New' and an empty 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 Tkinter GUI Designer is shown on picture below.

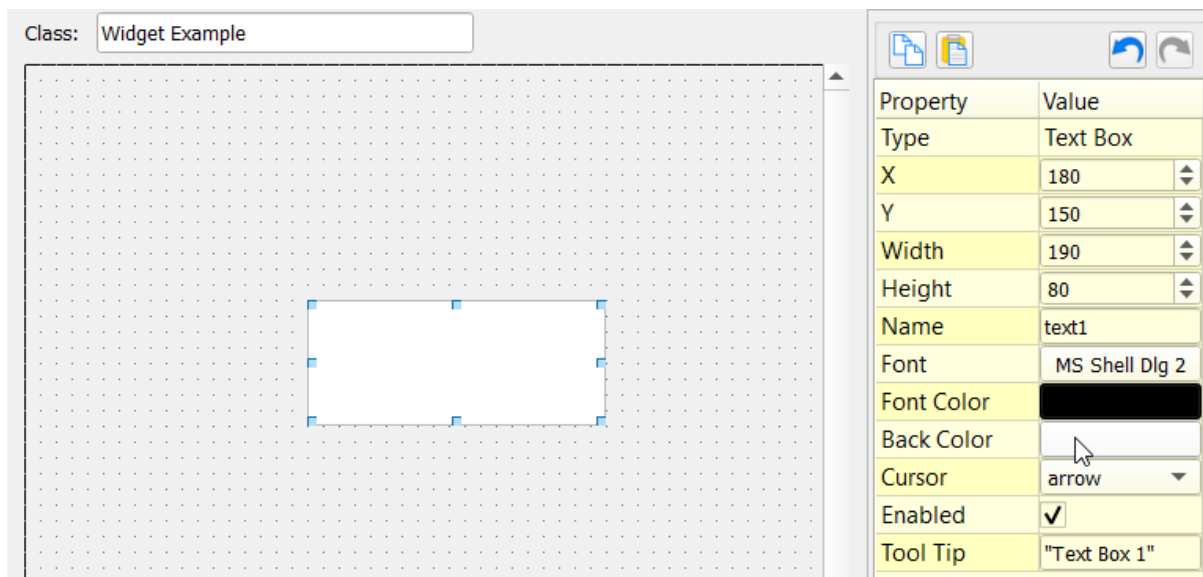


Picture 3: GUI Designer

On the left side of the 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.

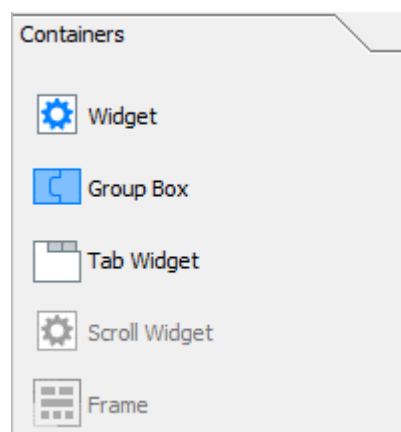
On the right-hand side of the Tkinter GUI Designer, the properties of the selected element are displayed and these properties can be modified here (Picture 4). In the example below, we have also edited the name of the widget to 'Widget Example'. To do this, you just have to write your preferred name 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 limited to what your code will do and so we do not want to ruin your imagination.

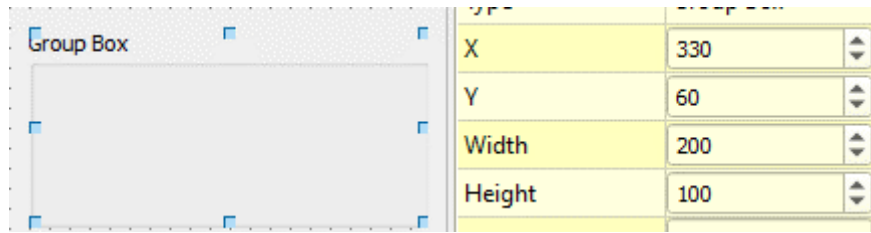
If we open the Containers menu, from the top left part of the Tkinter GUI Designer, a new list of GUI elements will appear. These elements are called containers and can be placed on the main widget area. They act as carriers for other GUI elements.



Picture 5: Containers

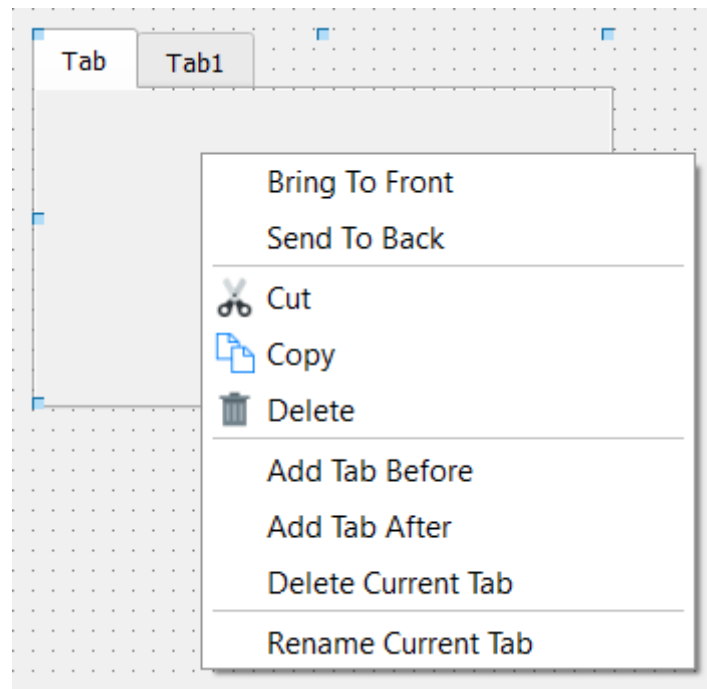
To insert any of these containers on the main form, use the same method as any other GUI element. First, select the preferred container and then, select where on the working area you want to place it.

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.



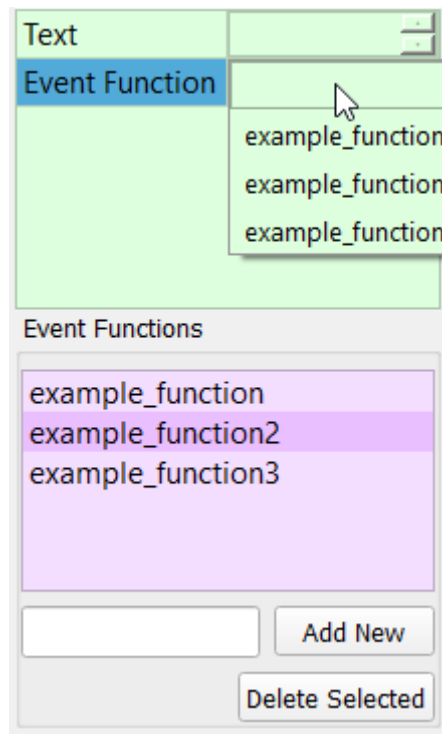
Picture 6: Resizing of object

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).

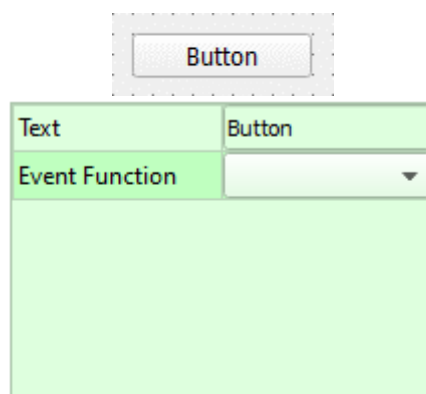


Picture 7: Event Functions

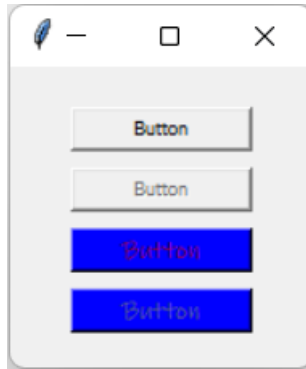
The Event Functions container is a global container which contains all the created event functions for the Widget that we are currently creating/editing. In the field named 'Event Function', you can select one of the defined functions by choosing one of the options that appears in the drop down list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

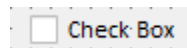


Generated Button



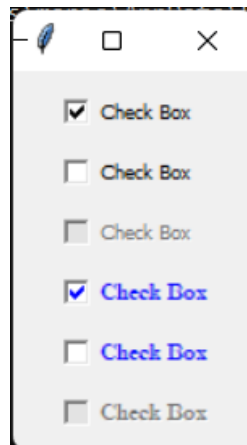
When the Button GUI element is inserted into the working area, the green box above will be visible on the left hand side. The 'Text' field can be used to change and set the visible label on the button. Additionally, you can choose the Event Function that will be used on the Button's event.

Check Box



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

Generated Check Box



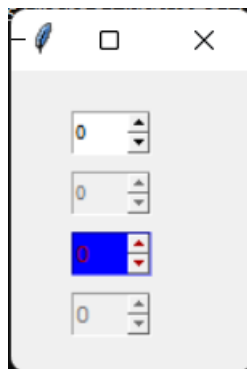
When a Check Box GUI element is inserted, the settings above will be visible. The text of the check box can be altered by writing in the 'Text' field. By default, the label used is 'Check Box'. The 'Value' attribute can be used to define the default value of the selected check box (whether the check box is checked or not). The Event Function will occur when the Check Box is checked.

Spin box



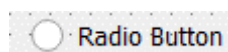
Minimum	0	▲▼
Maximum	99	▲▼
Step	1	▲▼
Value	0	▲▼
Event Function	▼	

Generated Spin box



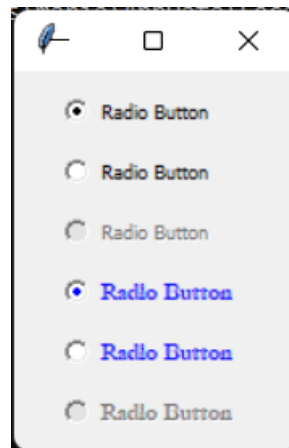
When a spin box GUI element is inserted, element specific settings that can be changed will appear. For example, the Minimum and Maximum value of the spin box can be set by using the 'Minimum' and 'Maximum' fields respectively. The step at which the spin box's value increments by can also be set by using 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



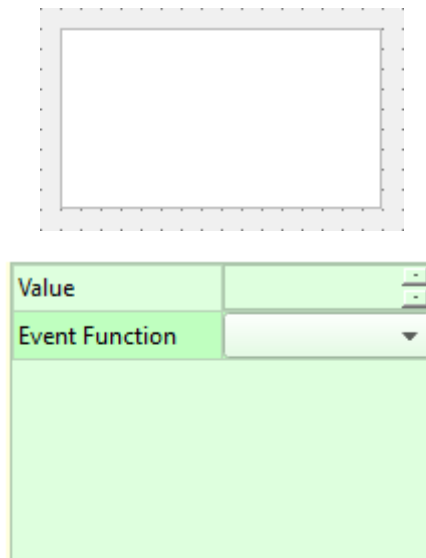
Text	Radio Button
Value	<input type="checkbox"/>
Event Function	▼

Generated Radio Button

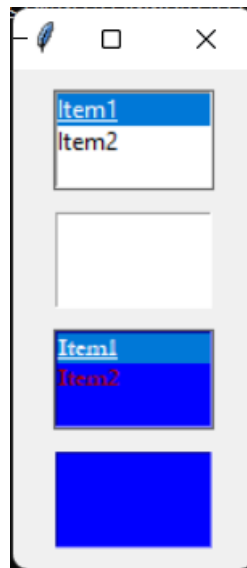


When a Radio button GUI element is inserted, the settings above will be visible. The text of the radio button can be altered by writing in the 'Text field. By default, the label used is 'Radio Button'. The 'Value' attribute can be used to define the default value of the selected radio button (whether 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

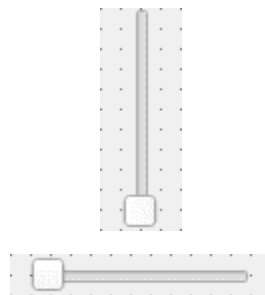


Generated List box



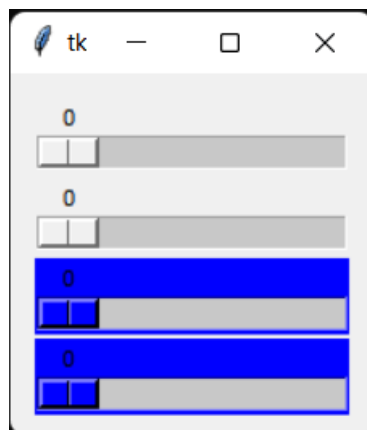
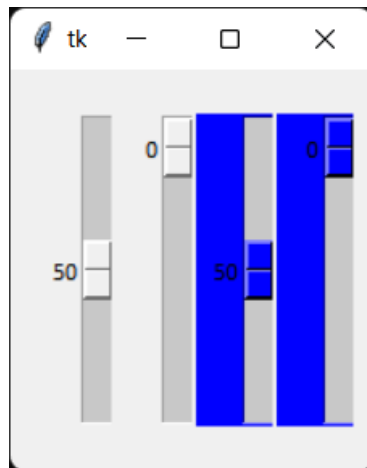
When a List box GUI element is inserted, the following options will be visible. Using the 'Value' attribute, the default value of the list box can be defined. Event Functions will occur when one of the list boxes containing rows is selected.

Sliders



Minimum	0	▲▼
Maximum	99	▲▼
Step	1	▲▼
Value	0	▲▼
Orientation	horizontal	▼
Event Function		▼

Generated Sliders



When a Slider is inserted, the following options will be visible. 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.

Label



Generated Label is identical to Tkinter GUI Designer look.

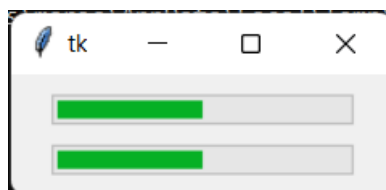
When a Label is inserted, it will only have one customizable setting. Only the Label's visible text can be edited. This is done by writing the preferred text in the 'Text' field at the top.

Progress Bars



Minimum	0
Maximum	100
Value	25
Orientation	horizontal

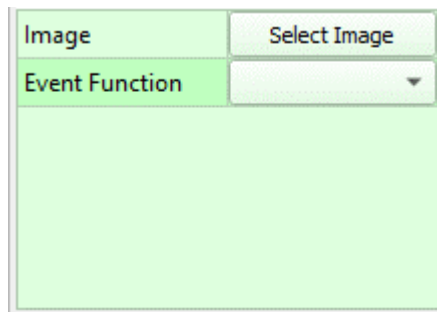
Generated Progress Bars



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 the horizontal orientation can be used on the Tkinter Progress bar.

Image

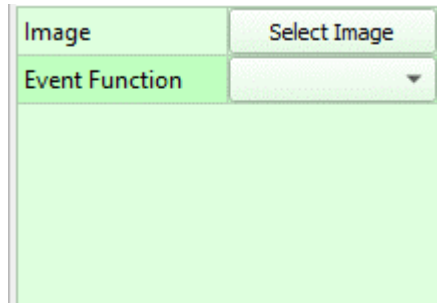
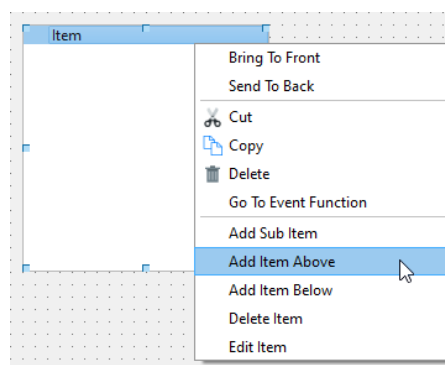




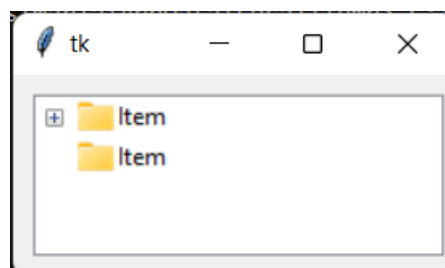
Generated Image is identical to Tkinter GUI Designer look.

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.

Tree



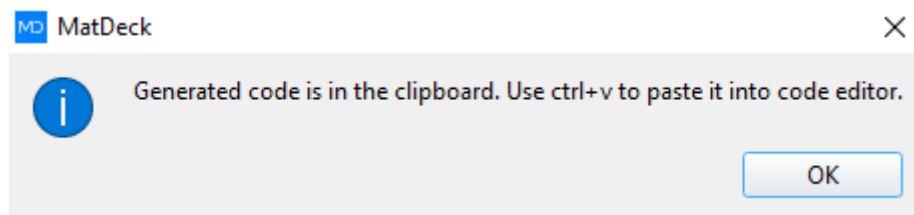
Generated Tree Image



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 an 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.

Finishing Up

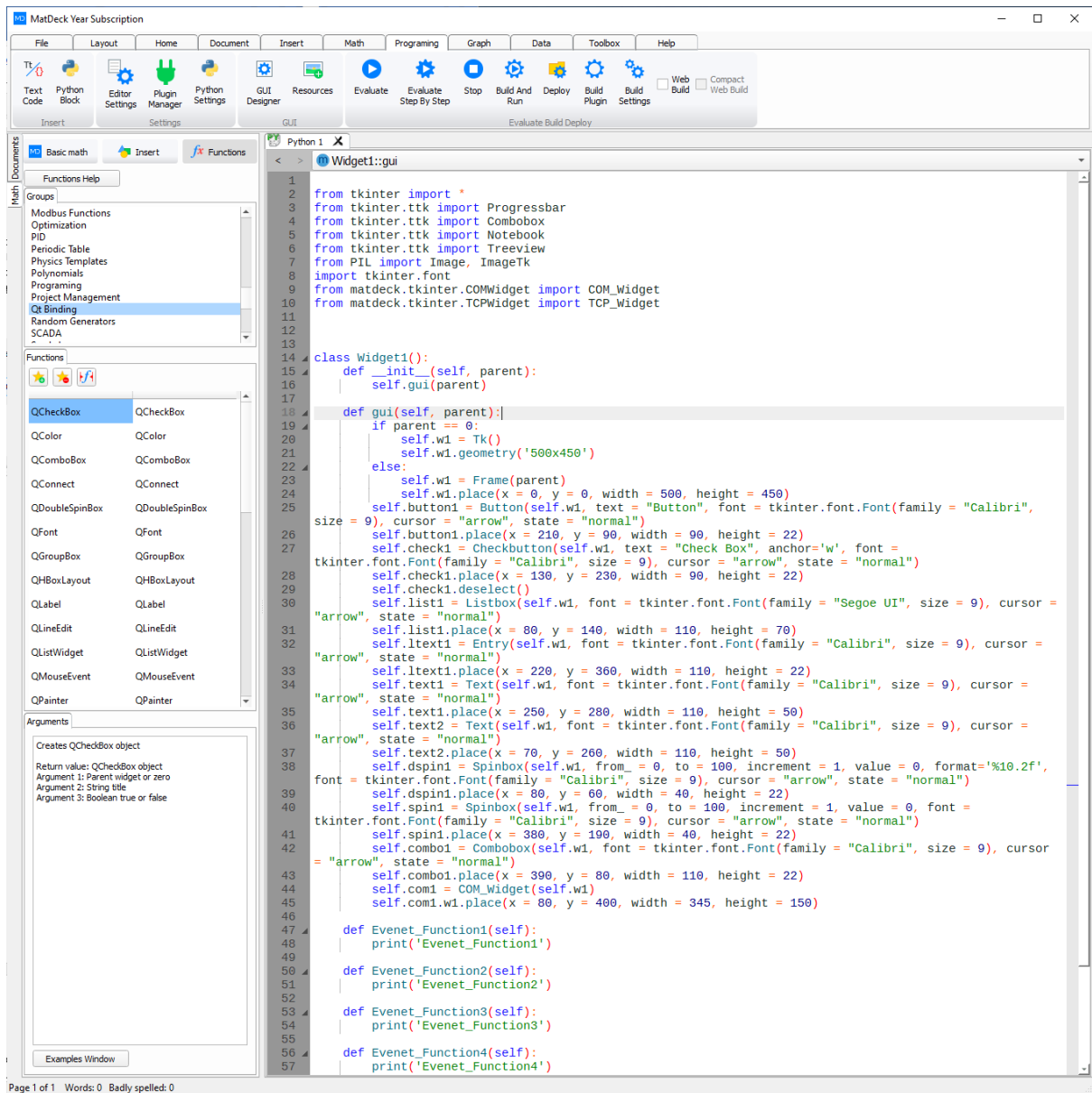
Once you are finished working on your widget design, you can press the 'Done' button and the Tkinter GUI Designer will generate the code for the widget and paste it into the If you have created a new widget for the first time and by pressed 'Done', the notification window (Picture 8) will open. Once the 'Done' button has been pressed for the first time, it will not show up again and code will be updated without showing this notification.



Picture 8: Notification Window

Using GUI Designer Code in a Tkinter Script

The code in the Tkinter script can now be changed and saved as a normal .py file. We can also use the Deploy button to package the Tkinter application into a .exe file and from there we can distribute it. Our Tkinter GUI Designer saves you time, stress and effort on your GUI so you can focus on what the app does instead of being lost in tedious GUI code.



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.


```

47  def Evenet_Function1(self):
48      | print('Evenet_Function1')
49
50  def Evenet_Function2(self):
51      | print('Evenet_Function2')
52
53  def Evenet_Function3(self):
54      | print('Evenet_Function3')
55
56  def Evenet_Function4(self):
57      | print('Evenet_Function4')
58
59  def Evenet_Function5(self):
60      | print('Evenet_Function5')

```

Picture 10: Event Functions 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.

```

62  if __name__ == '__main__':
63      a = Widget1(0)
64      a.w1.mainloop()

```

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 Tkinter Designer.

In short, the code created by the Tkinter GUI Designer is for creating the individual GUI elements and their properties, any interactions or changes need to be added by the user.

In this segment of the manual we will cover the two ,most important functions that are most commonly used.

[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.

```

1  A = self.hslider1.get()

```

In the example above, the function retrieves the value of the widget stored in the widget “hprogress1” and stores the value in the variable “A”.

Connecting widget values with other widgets/ set()

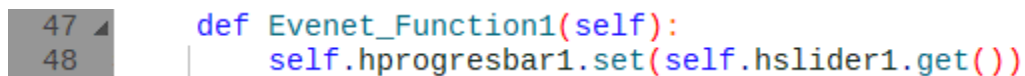
To display the value of a widget on a separate widget, the function set () needs to be used.

```
1 self.hprogressbar1.set(50)
```

The only argument needed is the value we would like to set, in this case it is 50.

When altering or interacting with the value of a GUI element, it should be done in its event function. For example, when interacting with the value of the slider, the function get() is used in the slider’s event function to get the value of the slider, then we use the set() function to set the value of the progress bar.

Below is an example of this.

The image shows a small GUI element on the left, which is a slider with two visible values, 47 and 48. To its right is a line of Python code:

```
def Evenet_Function1(self):  
    self.hprogressbar1.set(self.hslider1.get())
```

Executing and Outputting GUIs

Once GUI designs are completed, the generated code is placed directly into the script.

GUIs can be outputted using the following methods:

- Evaluation of generated code
- Deploy .EXE
- Build and Run
- Preview 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 Tkinter script in which it was evaluated

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.

