DOCUMENTATION OF THE FRAMEWORK LIBRARIES

THE FRAMEWORK'S LABEL WIDGET:

The widget is used to display a text or image on the screen.

Label(master=None, **options) (class)

master : Parent widget

**options : widget options

text=

The text to display in the label

side=

The side of the widget in the parent (master widget). Use 'top', 'bottom', 'left' or 'right'. The default is 'top'

Other keyword options can be found at:
http://effbot.org/tkinterbook/label.htm#Tkinter.Label.config-method

change_text(string)

Replace the current text of the label to the input string

get_label()

Get the Tkinter.Label object of the label, so it can be customized

Returns: Tkinter.Label object

THE FRAMEWORK'S ENTRY WIDGET:

This widget is used to enter text strings. This widget allows the user to enter one line of text, in a single font.

Entry(master=None, **options) (class)

master :parent widget

**options : widget options

side=

The side of the widget in the parent (master widget). Use 'top', 'bottom', 'left' or 'right'. The default is 'top'

defaultvalue=

Default text to be inserted in the entry

command=

A function or method that is called when enter key is pressed in the entry. The callback can be a function, bound method, or any other callable Python object. If this option is not used, nothing will happen.

Other keyword options can be found at:

http://effbot.org/tkinterbook/entry.htm#Tkinter.Entry.config-method

get()

Get the current content of the entry field

Returns: Entry content as a string

clear()

Delete all text in the entry

change_value(string)

Replace the current content with the input string

get_entry()

Get the Tkinter.Entry object of the entry.

Return: Tkinter.Entry object

THE FRAMEWORK 'S LABELENTRY WIDGET:

The widget is used to display a text and an entry to enter text strings in one line.

LabelEntry(master=None, **options) (class)

master : Parent widget

**options : widget options

text=

The text to display in the label

orient=

The orientation of the label and the entry. Use 'horizontal' or 'vertical'. The default is 'horizontal'.

side=

The side of the widget in the parent (master widget). Use 'top', 'bottom', 'left' or 'right'. The default is 'top'

labelwidth=

The width of the label. The size is given in text units. If the size is set to 0, or omitted, it is calculated based on the label contents.

entrywidth=

Width of the entry field, in character units. It does not limit the number of characters that can be typed into the entry field. The default width is 20 character

command=

A function or method that is called when enter key is pressed in the entry. The callback can be a function, bound method, or any other callable Python object. If this option is not used, nothing will happen.

defaultvalue=

Insert default text to the entry

get()

Gets the current contents of the entry field.

Returns: The entry contents, as a string.

clear()

delete all text in the widget.

change_label(string)

Change the text of the label

change_value(string)

Replace the text of the entry with the string

set_invisible()

Hide the widget. Other widgets can replace its position

set_visible()

Pack the widget if it is hidden

get_label()

Get the Tkinter.Label object of the label, so it can be more customized

Returns: Tkinter.Label object

get_entry()

Get the Tkinter.Entry object of the entry, so it can be more customized

Returns: Tkinter.Entry object

THE FRAMEWORK BUTTON WIDGET:

The Button widget is used to implement various kinds of buttons. Buttons contain text, and you can associate a Python function or method with each button. When the button is pressed, it automatically calls that function or method.

Button(master=None, **options) (class)

master : Parent widget

**options : widget options

text=

The text to display in the button

side=

The side of the widget in the parent (master widget). Use 'top', 'bottom', 'left' or 'right'. The default is 'top'

command=

A function or method that is called when the button is pressed. The callback can be a function, bound method, or any other callable Python object. If this option is not used, nothing will happen.

Other keyword options can be found at:

http://effbot.org/tkinterbook/button.htm#Tkinter.Button.config-method

get_button()

Get the Tkinter.Button object of the button, so it can be more customized

Returns: Tkinter.Button object

THE FRAMEWORK CHECKBUTTON WIDGET:

The Checkbutton widget is used to implement on-off selections. Checkbuttons can contain text, and you can associate a Python function or method with each button. When the button is pressed, Tkinter calls that function or method. The Checkbutton widget is used to choose between two distinct values (usually switching something on or off). Groups of Checkbuttons can be used to implement "many-of-many" selections. To handle "one-of-many" choices, use Radiobutton and Listbox widgets.

Checkbutton(master=None, list=[], **options) (class)

master: Parent widget

list: The list of texts displayed along with the check buttons

**options: widget options, note that this apply to all the checkbuttons

side=

The side of the widget in the parent (master widget). Use 'top', 'bottom', 'left' or 'right'. The default is 'top'

orient=

The orientation of the list of checkbuttons. Use 'horizontal' or 'vertical'. The default is 'horizontal'.

command=

A function or method that is called when the button is pressed. The callback can be a function, bound method, or any other callable Python object. If this option is not used, nothing will happen.

defaultvalue=

List of Default values of the checkbuttons. Note that on value is 1 and off value is 0 by default. So if the argument is omitted, the defaultvalues is [0,0,…,0], all buttons are deselected.

Other keyword options can be found at:

http://effbot.org/tkinterbook/checkbutton.htm#Tkinter.Checkbutton.config-method

get_value()

Get the list of current values of the checkbuttons. By default, on value is 1 and off value is 0

Return: List type

get_checkbutton(int)

Get single Tkinter.Checkbutton object with the input as the id in the checkbutton list

Return: Tkinter.Checkbutton object

THE FRAMEWORK LISTBOX WIDGET

The Listbox widget is used to display a list of alternatives. The Listbox can only contain text items, and all items must have the same font and color. Depending on the widget configuration, the user can choose one or more alternatives from the list.

listbox(master=None, list=[], **options) (class)

Master: Parent widget

List: The list of text items displayed in the list box

**options: widget options

side=

The side of the widget in the parent (master widget). Use 'top', 'bottom', 'left' or 'right'. The default is 'top'

height=

The height of the widget, default value is length of the text items. If the height is less than length of the text items, there would be a scrollbar on the left.

selectmode=

The listbox offers four different selection modes through the selectmode option. These are **'single'** (just a single choice), '**browse'** (same, but the selection can be moved using the mouse), '**multiple'** (multiple item can be choosen, by clicking at them one at a time), or '**extended'** (multiple ranges of items can be chosen, using the Shift and Control keyboard modifiers). The default is '**browse'**. Use '**multiple'** to get "checklist" behavior

label=

The description of the listbox displayed above the listbox

defaultvalue=

Default selection or set of selection (index)

command=

A function or method that is called when the item is selected or deselected. The callback can be a function, bound method, or any other callable Python object. If this option is not used, nothing will happen.

Other keyword options can be found at:

http://effbot.org/tkinterbook/listbox.htm#Tkinter.Listbox.config-method

get_selection()

If selectmode is default (‘browse’), return the currently selected item as index (integer), else return list of currently selected items as list of indexes.

Return: integer (if selectmode==’browse’), list type (if selectmode==’mutiple’,…)

get_listbox()

Get the Tkinter.Listbox object, so it can be more customized

Return: Tkinter.Listbox object

change_list(list)

Replace all the current text items with a new set of text items specified in list argument

get_name_selection()

If selectmode is ‘browse’, get the text of item currently selected, return as string

Return: string

THE FRAMEWORK'S RADIOBUTTON WIDGET:

The Radiobutton is used to implement one-of-many selections. Radiobuttons can contain text or images, and you can associate a Python function or method with each button. When the button is pressed, Tkinter automatically calls that function or method. It's almost always used in groups, where all group members use the same variable. The Radiobutton widget is very similar to the check button.

Listbox(master=None, list=[], **options) (class)

master: Parent widget

list: The list of text items displayed in the list box

**options: widget options

side=

The side of the widget in the parent (master widget). Use 'top', 'bottom', 'left' or 'right'. The default is 'top'

orient=

The orientation of the list of radiobuttons. Use 'horizontal' or 'vertical'. The default is 'horizontal'.

command=

A function or method that is called when the item is selected. The callback can be a function, bound method, or any other callable Python object. If this option is not used, nothing will happen.

defaultvalue=

Default selection (index)

Other keyword options can be found at:

http://effbot.org/tkinterbook/radiobutton.htm#Tkinter.Radiobutton.config-method

get_selection()

Get the current selected item as index (integer)

Return: integer

get_radiobutton(int)

Get single Tkinter.Radiobutton object with the input as the id in the radiobutton list

Return: Tkinter.Radiobutton object

set_invisible()

Hide the widget. Other widgets can replace its position

set_visible()

Pack the widget if it is hidden

THE FRAMEWORK'S ENTRYSCALE WIDGET

The EntryScale comprises an Entry and a Scale to adjust a value, whenever entry or scale value is changing, the other component changes as well. The Scale widget allows the user to select a numerical value by moving a "slider" knob along a scale. You can control the minimum and maximum values, as well as the resolution.

EntryScale(master=None, **options) (class)

master: Parent widget

**options: widget options

side=

The side of the widget in the parent (master widget). Use 'top', 'bottom', 'left' or 'right'. The default is 'top'

text=

Description text of the Entry

defaultvalue=

Defaultvalue of the value

from_=

Minimum value of the value (Default is -50) on Scale

to=

Maximum value of the value (Default is 50) on Scale

orient=

The orientation of the Slider (Scale), use 'horizontal' or 'vertical'. Default is 'horizontal'.

command=

A function or method that is called when the value is changed in either Entry or Scale. The callback can be a function, bound method, or any other callable Python object. If this option is not used, nothing will happen.

get_input()

Get the current value of the entry and scale

Return: integer

THE FRAMEWORK'S FIGURE WIDGET

The widget is used to display graph, plotting based on matplotlib visualization tool.

figure(master=None, **options) (class)

master: Parent widget

**options: widget options

side=

The side of the widget in the parent (master widget). Use 'top', 'bottom', 'left' or 'right'. The default is 'top'

figsize=

Figure size in tuple (width, height)

dpi=

Dots per inch

grid=

Subplots grid in tuple (width, height)

Plot(x=array[], y=array[], *args, **keywordargs)

X: array of X coordinates

Y: array of Y coordinates

*args: options

'update'

If you want to clear all the Figure to blank and redraw a new plot (X,Y), use argument 'update'. Like a hard reset on Figure object

**keywordargs: options

id=

If multiple subplots are available in initization, grid=(width, height), id arguments specifies the subplot to display the plotting, id is from 1 to width*height

xlim=

Tuple (min, max) specifies the minimum and maximum values in the plot (not the data)

ylim=

Tuple (min, max) specifies the minimum and maximum values in the plot (not the data)

title=

Text string to display above the plotting about the description of the plotting.

xlabel=

Text string to display along the x-axis, description about x-axis

ylabel=

Text string to display along the y-axis, description about y-axis

plot_type=

Use ‘stem’ or ‘scatter’ if stem or scatter plot is used. Otherwise, default is line

color=

Choose the color of markerline

markeredgecolor=

Choose the color of marker edge color

markerfacecolor=

Choose the color of marker face color

baseline=

Choose the color of baseline

stemline=

Choose the color of stemline

Return: line of the (X,Y) plotting used

get_lines(numberOfLines=int, *options)

numberOfLines: Number of lines to be generated

*options: option arguments

'update':

If you want to clear all the Figure to blank and redraw a new plot, use argument 'update'. Like a hard reset on Figure object

'stem':

If stem plot is used. Otherwise, the plotting is line.

Return: list type: list of lines created

plot_line(X=array[], Y=array[], line, **options)

X: array of X coordinates

Y: array of Y coordinates

**options: keyword options

xlim=

Tuple (min, max) specifies the minimum and maximum values in the plot (not the data)

ylim=

Tuple (min, max) specifies the minimum and maximum values in the plot (not the data)

title=

Text string to display above the plotting about the description of the plotting.

xlabel=

Text string to display along the x-axis, description about x-axis

ylabel=

Text string to display along the y-axis, description about y-axis

reset()

Clear all the plotting of the Figure

startAnimation(callback, iterable, **options)

callback: Callback function used to generate new set of data (X,Y)

iterable: A iterable list or array as input for callback function to gererate data

**options: keyword arguments

interval=

Integer, time between each time an element in the list is used

xlim=

Tuple (min, max) specifies the minimum and maximum values in the plot (not the data)

ylim=

Tuple (min, max) specifies the minimum and maximum values in the plot (not the data)