Next / Previous / Contents / Shipman's homepage

Tkinter 8.5 reference: a GUI for Python

---
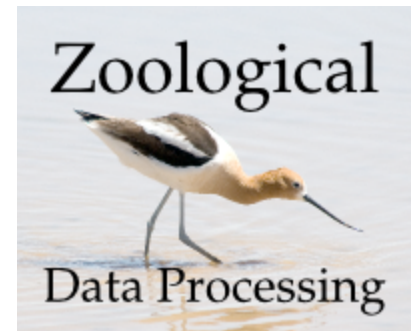
15. The `Menu` widget

"Drop-down" menus are a popular way to present the user with a number of choices, yet take up minimal space on the face of the application when the user is not making a choice.

- A *menubutton* is the part that always appears on the application.

- A *menu* is the list of choices that appears only after the user clicks on the menubutton.

- To select a choice, the user can drag the mouse from the menubutton down onto one of the choices. Alternatively, they can click and release the menubutton: the choices will appear and stay until the user clicks one of them.

- The Unix version of *Tkinter* (at least) supports "tear-off menus." If you as the designer wish it, a dotted line will appear above the choices. The user can click on this line to "tear off" the menu: a new, separate, independent window appears containing the choices.
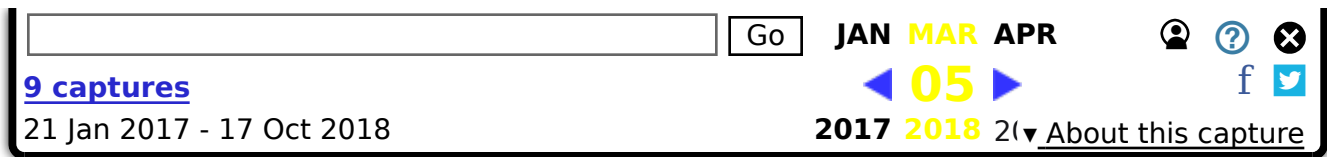
choices.

The choices displayed on a menu may be any of these things:

- A simple command: a text string (or image) that the user can select to perform some operation.

- A *cascade*: a text string or image that the user can select to show another whole menu of choices.

- A checkbutton (see [Section 9, "The Checkbutton widget"](#)).

- A group of radiobuttons (see [Section 20, "The Radiobutton widget"](#)).

To create a menu widget, you must first have created a `Menubutton`, which we will call `mb`:

```
    w = tk.Menu(mb, option, ...)
```

This constructor returns the new `Menu` widget. Options include:

Table 23. `Menu` widget options

activebackground	The background color that will appear on a choice when it is under the mouse. See [Section 5.3, "Colors"](#).
activeborderwidth	Specifies the width of a border drawn around a choice when it is under the mouse. Default is 1 pixel. For possible values, see [Section 5.1, "Dimensions"](#).
activeforeground	The foreground [color](#) that will appear on

	under the mouse.
`bd` or `borderwidth`	The width of the border around all the choices; see Section 5.1, "Dimensions". The default is one pixel.
`cursor`	The cursor that appears when the mouse is over the choices, but only when the menu has been torn off. See Section 5.8, "Cursors".
`disabledforeground`	The color of the text for items whose `state` is `tk.DISABLED`.
`font`	The default font for textual choices. See Section 5.4, "Type fonts".
`fg` or `foreground`	The foreground color used for choices not under the mouse.
`postcommand`	You can set this option to a procedure, and that procedure will be called every time someone brings up this menu.
`relief`	The default 3-D effect for menus is `relief=tk.RAISED`. For other options, see Section 5.6, "Relief styles".
`selectcolor`	Specifies the color displayed in checkbuttons and radiobuttons when they are selected.
`tearoff`	Normally, a menu can be torn off: the first position (position 0) in the list of choices is occupied by the tear-off element, and the additional choices are

|	starting at position 0.
`tearoffcommand`	If you would like your program to be notified when the user clicks on the tear-off entry in a menu, set this option to your procedure. It will be called with two arguments: the window ID of the parent window, and the window ID of the new tear-off menu's root window.
`title`	Normally, the title of a tear-off menu window will be the same as the text of the menubutton or cascade that lead to this menu. If you want to change the title of that window, set the `title` option to that string.

These methods are available on `Menu` objects. The ones that create choices on the menu have their own particular options; see [Section 15.1, "Menu item creation (coption) options"](#).

.add(*kind*, *coption*, ...)

Add a new element of the given *kind* as the next available choice in this menu. The *kind* argument may be any of `'cascade'`, `'checkbutton'`, `'command'`, `'radiobutton'`, or `'separator'`. Depending on the *kind* argument, this method is equivalent to `.add_cascade()`, `.add_checkbutton()`, and so on; refer to those methods below for details.

.add_cascade(*coption*, ...)

Add a new cascade element as the next available choice in
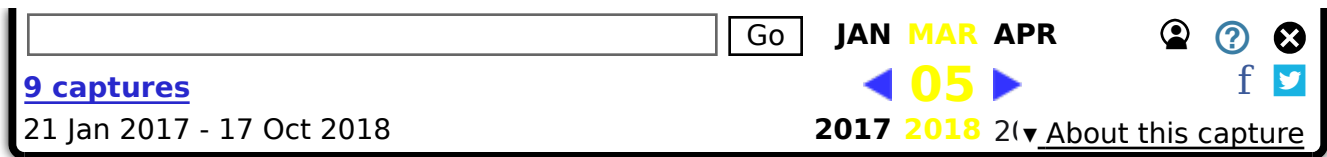
.add_checkbutton(*coption*, ...)

Add a new checkbutton as the next available choice in self.
The options allow you to set up the checkbutton much the
same way as you would set up a [Checkbutton](#) object; see
[Section 15.1, "Menu item creation (coption) options"](#).

.add_command(*coption*, ...)

Add a new command as the next available choice in self.
Use the label, bitmap, or image option to place text or an
image on the menu; use the command option to connect this
choice to a procedure that will be called when this choice
is picked.

**.add_radiobutton(*coption*, ...)**

Add a new radiobutton as the next available choice in self.
The options allow you to set up the radiobutton in much
the same way as you would set up a Radiobutton object;
see [Section 20, "The Radiobutton widget"](#).

.add_separator()

Add a separator after the last currently defined option.
This is just a ruled horizontal line you can use to set off
groups of choices. Separators are counted as choices, so if
you already have three choices, and you add a separator,
the separator will occupy position 3 (counting from 0).

.delete(*index1*, *index2*=None)

This method deletes the choices numbered from *index1*
through *index2*, inclusive. To delete one choice, omit the
index2 argument. You can't use this method to delete a
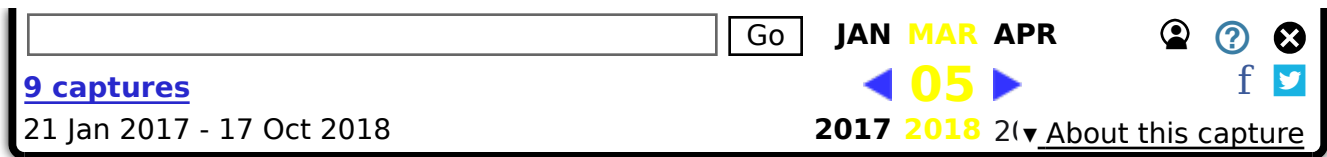
.entrycget(*index*, *coption*)

> To retrieve the current value of some coption for a choice,
> call this method with *index* set to the index of that choice
> and *coption* set to the name of the desired option.

.entryconfigure(*index*, *coption*, ...)

> To change the current value of some *coption* for a choice,
> call this method with *index* set to the index of that choice
> and one or more *coption*=*value* arguments.

.index(*i*)

> Returns the position of the choice specified by index *i*. For
> example, you can use .index(tk.END) to find the index of
> the last choice (or None if there are no choices).

**.insert_cascade(*index*, *coption*, ...)**

> Inserts a new cascade at the position given by *index*,
> counting from 0. Any choices after that position move
> down one. The options are the same as for .add_cascade(),
> above.

.insert_checkbutton(*index*, *coption*, ...)

> Insert a new checkbutton at the position specified by
> *index*. Options are the same as for .add_checkbutton(),
> above.

.insert_command(*index*, *coption*, ...)

> Insert a new command at position *index*. Options are the
> same as for .add_command(), above.
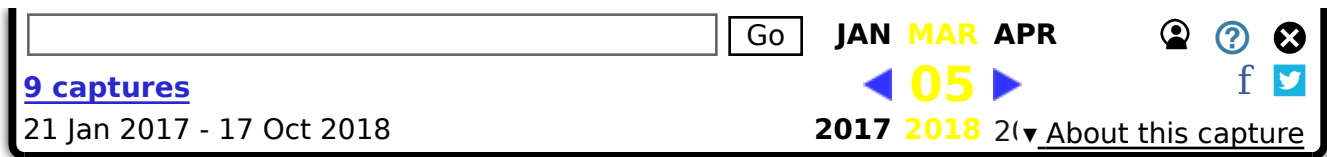
same as for `.add_radiobutton()`, above.

`.insert_separator(index)`

Insert a new separator at the position specified by *index*.

`.invoke(index)`

Calls the `command` callback associated with the choice at position *index*. If a checkbutton, its state is toggled between set and cleared; if a radiobutton, that choice is set.

`.post(x, y)`

Display this menu at position (*x*, *y*) relative to the root window.

`.type(index)`

Returns the type of the choice specified by *index*: either `tk.CASCADE`, `tk.CHECKBUTTON`, `tk.COMMAND`, `tk.RADIOBUTTON`, `tk.SEPARATOR`, or `tk.TEAROFF`.

`.yposition(n)`

For the *n*th menu choice, return the vertical offset in pixels relative to the menu's top. The purpose of this method is to allow you to place a popup menu precisely relative to the current mouse position.

15.1. `Menu` **item creation (coption) options**

Wherever the menu methods described above allow a `coption`,

letters, use "foreground='red'" as an option to the add_command method call.

Table 24. Menu item coption values

accelerator	To display an "accelerator" keystroke combination on the right side of a menu choice, use the option "accelerator=*s*" where *s* is a string containing the characters to be displayed. For example, to indicate that a command has Control-X as its accelerator, use the option "accelerator='^X'". Note that this option does not actually implement the accelerator; use a keystroke binding to do that.
activebackground	The background color used for choices when they are under the mouse.
activeforeground	The foreground color used for choices when they are under the mouse.
background	The background color used for choices when they are *not* under the mouse. Note that this *cannot* be abbreviated as bg.
bitmap	Display a bitmap for this choice; see [Section 5.7, "Bitmaps"](#).
columnbreak	Normally all the choices are displayed in one long column. If you set columnbreak=1, this choice will start a new column to the right of the one containing the previous choice.

command	A procedure to be called when this choice is activated.
compound	If you want to display both text and a graphic (either a bitmap or an image) on a menu choice, use this `coption` to specify the location of the graphic relative to the text. Values may be any of `tk.LEFT`, `tk.RIGHT`, `tk.TOP`, `tk.BOTTOM`, `tk.CENTER`, or `tk.NONE`. For example, a value of "compound=tk.TOP" would position the graphic above the text.
font	The font used to render the `label` text. See [Section 5.4, "Type fonts"](#)
foreground	The foreground [color](#) used for choices when they are *not* under the mouse. Note that this *cannot* be abbreviated as `fg`.
hidemargin	By default, a small margin separates adjacent choices in a menu. Use the `coption` "hidemargin=True" to suppress this margin. For example, if your choices are color swatches on a palette, this option will make the swatches touch without any other intervening color.
image	Display an image for this choice; see [Section 5.9, "Images"](#).
label	The text string to appear for this choice.
menu	This option is used only for cascade choices. Set it to a `Menu` object that displays the next level of choices.

	value by setting this option to the desired value. See Section 52, "Control variables: the values behind the widgets".
onvalue	Normally, the control variable for a checkbutton is set to 1 when the checkbutton is on. You can change the on value by setting this option to the desired value.
selectcolor	Normally, the color displayed in a set checkbutton or radiobutton is red. Change that color by setting this option to the color you want; see Section 5.3, "Colors".
selectimage	If you are using the image option to display a graphic instead of text on a menu radiobutton or checkbutton, if you use selectimage=I, image I will be displayed when the item is selected.
state	Normally, all choices react to mouse clicks, but you can set state=tk.DISABLED to gray it out and make it unresponsive. This coption will be tk.ACTIVE when the mouse is over the choice.
underline	Normally none of the letters in the label are underlined. Set this option to the index of a letter to underline that letter.
value	Specifies the value of the associated control variable (see Section 52, "Control variables: the values behind the widgets") for a radiobutton. This can be an integer if

variable	For checkbuttons or radiobuttons, this option should be set to the control variable associated with the checkbutton or group of radiobuttons. See [Section 52, "Control variables: the values behind the widgets"]().

Next: [15.2. Top-level menus]()
Contents: [*Tkinter* 8.5 reference: a GUI for Python]()
Previous: [14. The `Listbox` widget]()
Home: [John Shipman's Home Sweet Homepage]()

John W. Shipman
Comments welcome: [*john@nmt.edu*]()
Last updated: 2013-09-09 22:58
URL: `http://www.nmt.edu/~shipman/soft/tkinter/web/menu.html`