

## Interfata grafica Tkinter in Python

Python ofera mai multe optiuni in ceea ce priveste realizarea interfețelor grafice cu utilizatorul (GUI). Cele mai importante sunt:

- **Tkinter** - este o interfață grafică care este deja prezenta in pachetul software al Python.
- **WxPython** - aceasta este o interfață open-source pentru wxWindows (<http://www.wxpython.org>).
- **JPython** - este modulul care acordă acces scripturilor Python la librariile Java de pe masina locala (<http://www.jython.org>).

Sunt mult mai multe interfete grafice pentru Python, însă vă lăsăm plăcerea să le descoperiți dumneavoastră pe net. În acest capitol o sa ne referim la interfata grafica Tkinter care este după cum am spus, modulul integrat in pachetul Python.

Python împreună cu modulul Tkinter oferă o cale ușoară și rapidă pentru crearea interfetelor grafice cu utilizatorul, fiind necesari urmtorii pași:

- Importarea modulului Tkinter;
- Crearea ferestrei principale a interfetei grafice;
- Adaugarea de elemente (widget-uri) in fereastra principala;
- Crearea evenimentului care afișează fereastra principală cu elementele pe care le conține.

### 13.1 Elemente (widget-uri) Tkinter

Button	Afișează butoane în aplicație.
Canvas	Utilizat pentru afișarea diferitelor forme ca de exemplu linii, ovale, poligoane și dreptunghiuri în aplicatie.

Checkbox	Afișează un număr de opțiuni ca checkbox-uri în care utilizatorul poate selecta mai multe opțiuni.
Entry	Utilizat la afișarea unei linii de text care accepta input de la utilizator.
Frame	Widget-ul Frame este utilizat drept container organizator pentru alte widget-uri.
Label	Este utilizat pentru a eticheta într-o singură linie de text alte widget-uri. Poate conține imagini.
Listbox	Este utilizat pentru a oferi o listă de opțiuni utilizatorului.
Menubutton	Utilizat pentru afișarea Menu-urilor în aplicație.
Menu	Folosit pentru a oferi comenzi variate utilizatorului în cadrul Menubutton-ului.
Message	Folosit pentru afișarea de text multilinie pentru acceptarea valorilor de către utilizator.
Radiobutton	Afișează opțiunile sub forma unor butoane radio, din care utilizatorul poate selecta o singură opțiune.
Scale	O fereastră care glisează în jos sau în sus.
Scrollbar	Utilizat pentru scrolling-ul diferitelor widget-uri, spre exemplu Listbox.
Text	Afișarea mai multor linii de text.
Toplevel	Oferă o fereastră container pentru widget-uri.
Spinbox	Este asemenea unui Entry, dar care oferă spre selecție un număr limitat de opțiuni.
PanedWindow	Este un container care conține un număr de panouri aranjate orizontal sau vertical.
LabelFrame	Este un container al cărui scop este să ofere un layout mai complex în ferestre.
tkMessageBox	Afișează ferestre cu mesaje în aplicație.

### 13.2 Atributele standard ale elementelor (widget)

Elementele enumerate mai sus pot lua un număr de atribute pe care le vom enumera în continuare: Dimensions, Colors, Fonts, Anchors, Relief Styles, Bitmaps, Cursors.

### 13.3 Managementul geometriei ferestrelor

Pentru afișarea acestor widget-uri în fereastra principală avem nevoie

de un manager de geometrie a ferestrelor, iar Tkinter utilizeaza clasele: pack, grid si place.

Metoda **pack()** - acest manager organizează widget-urile în blocuri înainte de a le plasa in fereastra.

Syntaxa:

```
widget.pack (pack_options)
```

Widget-ul este orice element din cle din tabelul cu elemente, iar pack\_options sunt:

- **expand** - daca este true, umple tot spațiul din fereastră.
- **fill** - umple extraspatiu alocat de catre pack() sau îi păstrează dimensiunile minime: NONE(default), X (umple doar vertical), Y (umple doar orizontal), BOTH (umple vertical și orizontal).
- **side** - determină unde plasează widget-ul in fereastră: TOP (default), BOTTOM, LEFT sau RIGHT.

Exemplu:

```
#!/usr/bin/env python

from Tkinter import *

root = Tk()
frame = Frame(root)
frame.pack()

bottomframe = Frame(root)
bottomframe.pack()

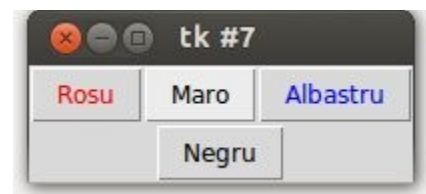
redbutton = Button(frame, text="Rosu", fg="red")
redbutton.pack(side = LEFT)

greenbutton = Button(frame, text="Maro", fg="brown")
greenbutton.pack( side = LEFT)

bluebutton = Button(frame, text="Albastru", fg="blue")
bluebutton.pack( side=LEFT)

blackbutton = Button(bottomframe, text="Negru", fg="black")
blackbutton.pack( side = BOTTOM)

root.mainloop()
```



Rularea codului de mai sus genereaza imaginea din fereastra codului.

Metoda **grid()** - organizează widgeturile intr-o structura asemanatoare unui tabel.

Sintaxa:

```
Widget.grid( grid_options )
```

Grid\_options poate lua valorile:

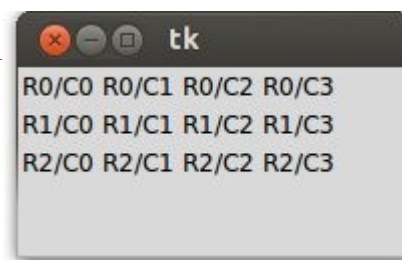
- **column** - coloana in care se pune widget-ul, default este 0;
- **columnspan** - câte coloane ocupa widget-ul, default este 1;
- **ipadx, ipady** - câți pixeli ocupă widget-ul, orizontal și vertical, in marginile widget-ului;
- **padx, pady** - câți pixeli ocupă widget-ul, orizontal si vertical, în afara widgetului;
- **row** - randul in care este pus widget-ul, default primul rand care este liber;
- **rowspan** - câte randuri ocupa widget-ul, default este 1;
- **sticky** - daca celula este mai mare decât widget-ul, cu "sticky=" , este afisat in centru, cu această opțiune widget-ul poate fi orientat N, S, W, E, NE, SE si SW si se refera la coltul celulei.

Exemplu:

```
#!/usr/bin/env python
```

```
import Tkinter
root=Tkinter.Tk()
for r in range(3):
    for c in range(4):
        Tkinter.Label(root, text ='R%s/C%s'%(r,c),
                       borderwidth =1).grid(row=r, column=c)

root.mainloop()
```



Imaginea din cadrul codului afisează 12 etichete intr-un tabel de 3x4.

Metoda **place()** - afisează widget-urile intr-o pazație specificată.

Sintaxa:

```
Widget.place( place_options)
```

- **Anchor** - locul exact al widget-ului care poate fi N, E, S, W, NW, SE sau SW care indică colțurile ferestrei, setat este NW (stanga sus);
- **bordermode** - **INSIDE** (default) in interiorul ferestrei aplicației, **OUTSIDE** afara;
- **height, width** - valori in pixeli;
- **relheight, relwidth** - sunt valori intre 0.0 si 0.1 ca fracție

- din fereastra aplicației;
- rslx, rely - orizontal si vertical cu valori tot intre 0.0 si 0.1 ca fractie din fereastra aplicatiei;
- x, y - orizontal si vertical in pixeli.

Exemplu:

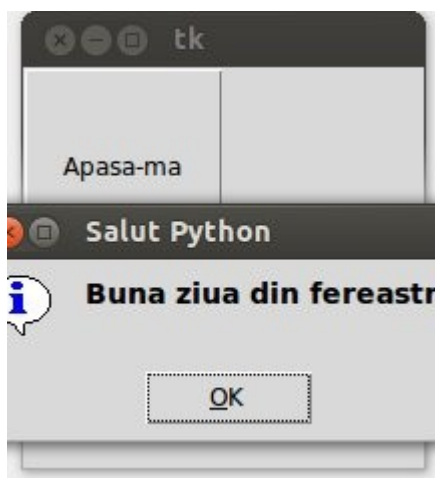
```
#!/usr/bin/env python
"""
File fereastabuton.py
Autor Mihai Cornel
Email mhcrnl@gmail.com
Exemplificare a utilizarii butoanelor in ferestrele Tkinter.
"""
from Tkinter import *
import tkMessageBox
import Tkinter

top=Tkinter.Tk()

def accesMesaj():
    tkMessageBox.showinfo("Salut Python", "Buna ziua din
fereastra!")

buton = Tkinter.Button(top, text="Apasa-ma", command= accesMesaj)

buton.pack()
buton.place(bordermode=OUTSIDE, height=100, width=100)
top.mainloop()
```



Acestea sunt ferestrele pe care le genereaza codul de mai sus, apasarea butonului "Apasa-ma" deschide o fereasta de mesaj.

## 13.4 Widget-ul Button

Este utilizat pentru adaugarea de butoane in aplicatiile Python.

Acestea pot afișa text sau imagini care subliniază scopul butonului. Se poate atașa o funcție sau o metodă care va fi apelată automat când butonul este apăsător.

Sintaxa:

```
Buton = Button( master, option = value, ... )
```

Parametri:

- master - reprezintă fereastra în care va fi afișat butonul;
- option - mai jos puteți consulta o listă cu opțiuni pentru butoane. Acestea sunt folosite sub forma cheie=valoare, separate prin virgulă.

Opțiuni	Descriere
activebackground	Culoarea fundalului butonului când se află sub cursor.
activeforeground	Culoarea textului din buton când se află sub cursor.
bd	Bordura din jurul butonului, presetată la valoarea 2.
bg	Culoarea normală a butonului.
command	Funcția sau metoda care va fi apelată în momentul acționării butonului.
fg	Culoarea normală a textului butonului.
font	Tipul de font folosit în textul butonului.
height	Înălțimea butonului în linii de text sau în pixeli pentru imagini.
highlightcolor	Culoarea de focus.
image	Imaginea utilizată în cadrul butonului în detrimentul textului.
justify	Cum sunt afișate liniile de text: LEFT, CENTER sau RIGHT.
padx	Distanțarea textului în stânga sau dreapta.
pady	Distanțarea textului în sus sau jos.
relief	Specifică tipul de bordură: SUNKEN, RAISED, GROOVE și RIDGE.
state	NORMAL este starea presetată. DISABLED face butonul să nu răspundă. ACTIVE când mouse-ul este deasupra să devină activ.
underline	Valoarea -1 nici o literă a textului nu este subliniată, dacă este o valoare pozitivă litera

	corespunzatoare va fi subliniata.
width	Lungimea butonului.
wraptlength	Daca valoarea este un numar pozitiv, textul este afisat pe mai multe linii.

Exemplu:

```
#!/usr/bin/env python
"""
File fereastabuton.py
Autor Mihai Cornel
Email mhcrnl@gmail.com
Exemplificare a utilizarii butoanelor in ferestrele Tkinter.
"""
from Tkinter import *
import tkMessageBox
import Tkinter

top=Tkinter.Tk()

def accesMesaj():
    tkMessageBox.showinfo("Salut Python", "Buna ziua din fereastra!")

buton = Tkinter.Button(top,
                        text="Apasa-ma te rog!",
                        command= accesMesaj,
                        activebackground= "red",
                        activeforeground= "green",
                        bd = 4,
                        bg = "white",
                        fg = "blue",
                        font="italic",
                        height = 200,
                        highlightcolor= "green",
                        underline = 3)

buton.pack()
buton.place(bordermode=OUTSIDE, height=150, width=150)
top.mainloop()
```

### 13.5 Widget-ul canvas

Este o arie dreptunghiulara utilizata pentru desenarea de figuri complexe. De asemenea poate afisa grafica, text, widget-uri sau ferestre.

## Sintaxa:

```
Canvas1 = Canvas( master, option= value, ...)
```

## Parametri:

- master - reprezinta fereastra in care va fi afisat butonul;
- option - mai jos puteti consulta o lista cu optiuni pentru butoane. Acestea sunt folosite sub forma cheie=valoare, separate prin virgula.

Option	Description
bd	Border width in pixels. Default is 2.
bg	Normal background color.
confine	If true (the default), the canvas cannot be scrolled outside of the scrollregion.
cursor	Cursor used in the canvas like <i>arrow</i> , <i>circle</i> , <i>dot</i> etc.
height	Size of the canvas in the Y dimension.
highlightcolor	Color shown in the focus highlight.
relief	Relief specifies the type of the border. Some of the values are SUNKEN, RAISED, GROOVE, and RIDGE.
scrollregion	A tuple (w, n, e, s) that defines over how large an area the canvas can be scrolled, where w is the left side, n the top, e the right side, and s the bottom.
width	Size of the canvas in the X dimension.
xscrollincrement	If you set this option to some positive dimension, the canvas can be positioned only on multiples of that distance, and the value will be used for scrolling by scrolling units, such as when the user clicks on the arrows at the ends of a scrollbar.
xscrollcommand	If the canvas is scrollable, this attribute should be the .set() method of the horizontal scrollbar.
yscrollincrement	Works like xscrollincrement, but governs vertical movement.
yscrollcommand	If the canvas is scrollable, this attribute should be the .set() method of the vertical scrollbar.

Widget-ul canvas poate suporta urmatoarele elemente:

### arc

```
coord = 10, 50, 240, 210  
arc = canvas.create_arc(coord, start=0, extent=150, fill="blue")
```

### image

```
filename = PhotoImage(file = "sunshine.gif")  
image = canvas.create_image(50, 50, anchor=NE, image=filename)
```

### Line

```
line = canvas.create_line(x0, y0, x1, y1, ..., xn, yn, options)
```



## Oval

```
oval = canvas.create_oval(x0, y0, x1, y1, options)
```

## Polygon

```
oval = canvas.create_polygon(x0, y0, x1, y1,...xn, yn, options)
```

## Exemplu:

```
#!/usr/bin/env python
"""
File fereastacanvas.py
Autor Mihai Cornel
Email mhcrnl@gmail.com
Exemplificare a utilizarii butoanelor in ferestrele Tkinter.
"""
import Tkinter
import tkMessageBox

top= Tkinter.Tk()

canvas1 = Tkinter.Canvas(top,
                          bg="blue",
                          height = 250,
                          width=300)

coord = 10,50,240,210
arc = canvas1.create_arc(coord, start = 0, extent=150, fill="red")

canvas1.pack()
top.mainloop()
```

## 13.6 Widget-ul Checkbutton

Checkbutton este utilizat pentru a afisa mai multe optiuni sub forma de butoane. Utilizatorul poate selecta unul sau mai multe optiuni printr-un click corespunzator optiunilor.

### Syntaxa:

```
Checkbuton = Checkbutton( master, option, ...)
```

Ca si in cazul celorlalte widget-uri master se refera la fereastra principala iar optiunile sunt urmatoarele:

Option	Description
activebackground	Background color when the checkbutton is under the cursor.
activeforeground	Foreground color when the checkbutton is under the cursor.

bg	The normal background color displayed behind the label and indicator.
bitmap	To display a monochrome image on a button.
bd	The size of the border around the indicator. Default is 2 pixels.
command	A procedure to be called every time the user changes the state of this checkbox.
cursor	If you set this option to a cursor name ( <i>arrow, dot etc.</i> ), <i>the mouse cursor will change to that pattern when it is over the checkbox.</i>
disabledforeground	The foreground color used to render the text of a disabled checkbox. The default is a stippled version of the default foreground color.
font	The font used for the text.
fg	The color used to render the text.
height	The number of lines of text on the checkbox. Default is 1.
highlightcolor	The color of the focus highlight when the checkbox has the focus.
image	To display a graphic image on the button.
justify	If the text contains multiple lines, this option controls how the text is justified: CENTER, LEFT, or RIGHT.
offvalue	Normally, a checkbox's associated control variable will be set to 0 when it is cleared (off). You can supply an alternate value for the off state by setting offvalue to that value.
onvalue	Normally, a checkbox's associated control variable will be set to 1 when it is set (on). You can supply an alternate value for the on state by setting onvalue to that value.
padx	How much space to leave to the left and right of the checkbox and text. Default is 1 pixel.
pady	How much space to leave above and below the checkbox and text. Default is 1 pixel.
relief	With the default value, relief=FLAT, the checkbox does not stand out from its background. You may set this option to any of the other styles
selectcolor	The color of the checkbox when it is set. Default is selectcolor="red".
selectimage	If you set this option to an image, that image will appear in the checkbox when it is set.
state	The default is state=NORMAL, but you can use state=DISABLED to gray out the control and make it unresponsive. If the cursor is currently over the checkbox, the state is ACTIVE.
text	The label displayed next to the checkbox. Use newlines ("\n") to display multiple lines of text.
underline	With the default value of -1, none of the characters of the text label are underlined. Set this option to the index of a character in the text (counting from zero) to underline that character.
variable	The control variable that tracks the current state of the checkbox. Normally this variable is an <i>IntVar</i> , and 0 means cleared and 1 means set, but see the offvalue and onvalue options above.
width	The default width of a checkbox is determined by the size of the displayed image or text. You can set this option to a number of characters and the checkbox will always have room for that many characters.

wraplength Normally, lines are not wrapped. You can set this option to a number of characters and all lines will be broken into pieces no longer than that number.

Acest widget are si metodele:

Method	Description
deselect()	Clears (turns off) the checkbutton.
flash()	Flashes the checkbutton a few times between its active and normal colors, but leaves it the way it started.
invoke()	You can call this method to get the same actions that would occur if the user clicked on the checkbutton to change its state.
select()	Sets (turns on) the checkbutton.
toggle()	Clears the checkbutton if set, sets it if cleared.

Exemplu:

```
#!/usr/bin/env python

from Tkinter import *

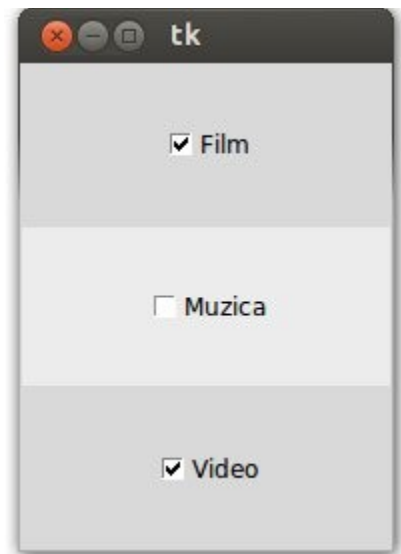
import tkMessageBox
import Tkinter

top = Tkinter.Tk()
radbut1 = IntVar()
radbut2 = IntVar()
radbut3 = IntVar()

but1 = Checkbutton(top,
                    text = "Film",
                    variable = radbut1,
                    onvalue = 1,
                    offvalue = 0,
                    height = 5,
                    width = 20)

but2 = Checkbutton(top,
                    text = "Muzica",
                    variable = radbut2,
                    onvalue = 1,
                    offvalue = 0,
                    height = 5,
                    width = 20)

but3 = Checkbutton(top,
                    text = "Video",
                    variable = radbut3,
                    onvalue = 1,
                    offvalue = 0,
```



```
        height = 5,  
        width = 20)  
but1.pack()  
but2.pack()  
but3.pack()  
top.mainloop()
```

### 13.7 Widget-ul Entry

Entry este utilizat pentru a prelua date de la utilizator, de exemplu text. Acest widget permite utilizatorului sa scrie o singura linie de text, daca textul este mai lung acesta nu va fi afisat in intregime.

Syntaxa:

```
W = Entry( master, option, ...)
```

Option	Description
bg	The normal background color displayed behind the label and indicator.
bd	The size of the border around the indicator. Default is 2 pixels.
command	A procedure to be called every time the user changes the state of this checkbutton.
cursor	If you set this option to a cursor name ( <i>arrow, dot etc.</i> ), the mouse cursor will change to that pattern when it is over the checkbutton.
font	The font used for the text.
exportselection	By default, if you select text within an Entry widget, it is automatically exported to the clipboard. To avoid this exportation, use exportselection=0.
fg	The color used to render the text.
highlightcolor	The color of the focus highlight when the checkbutton has the focus.
justify	If the text contains multiple lines, this option controls how the text is justified: CENTER, LEFT, or RIGHT.
relief	With the default value, relief=FLAT, the checkbutton does not stand out from its background. You may set this option to any of the other styles
selectbackground	The background color to use displaying selected text.
selectborderwidth	The width of the border to use around selected text. The default is one pixel.
selectforeground	The foreground (text) color of selected text.
show	Normally, the characters that the user types appear in the entry. To make a .password. entry that echoes each character as an asterisk, set show="*".
state	The default is state=NORMAL, but you can use state=DISABLED to gray out the control and make it unresponsive. If the cursor is currently over the checkbutton, the state is ACTIVE.
textvariable	In order to be able to retrieve the current text from your entry widget, you must set this option to an instance of the StringVar class.
width	The default width of a checkbutton is determined by the size of the displayed image or text. You can set this option to a number of characters and the checkbutton will

always have room for that many characters.

xscrollcommand If you expect that users will often enter more text than the onscreen size of the widget, you can link your entry widget to a scrollbar.

Acestea sunt metodele care pot fi aplicate widget-ului entry:

Method	Description
delete ( first, last=None )	Deletes characters from the widget, starting with the one at index first, up to but not including the character at position last. If the second argument is omitted, only the single character at position first is deleted.
get()	Returns the entry's current text as a string.
icursor ( index )	Set the insertion cursor just before the character at the given index.
index ( index )	Shift the contents of the entry so that the character at the given index is the leftmost visible character. Has no effect if the text fits entirely within the entry.
insert ( index, s )	Inserts string s before the character at the given index.
select_adjust ( index )	This method is used to make sure that the selection includes the character at the specified index.
select_clear()	Clears the selection. If there isn't currently a selection, has no effect.
select_from ( index )	Sets the ANCHOR index position to the character selected by index, and selects that character.
select_present()	If there is a selection, returns true, else returns false.
select_range ( start, end )	Sets the selection under program control. Selects the text starting at the start index, up to but not including the character at the end index. The start position must be before the end position.
select_to ( index )	Selects all the text from the ANCHOR position up to but not including the character at the given index.
xview ( index )	This method is useful in linking the Entry widget to a horizontal scrollbar.
xview_scroll ( number, what )	Used to scroll the entry horizontally. The what argument must be either UNITS, to scroll by character widths, or PAGES, to scroll by chunks the size of the entry widget. The number is positive to scroll left to right, negative to scroll right to left.

Exemplu:

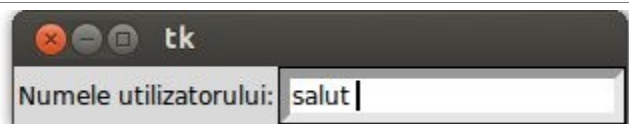
```
#!/usr/bin/env python
```

```
from Tkinter import *
```

```
top = Tk()
```

```
l1 = Label(top,  
            text = "Numele utilizatorului:")  
l1.pack(side=LEFT)
```

```
E1 = Entry(top,  
            bd=5)
```



```
E1.pack(side = RIGHT)

top.mainloop()
```

Am creat o fereastră în care avem un câmp de input, următoarea întrebare care se pune este cum putem prelua datele din el? Metoda care ne accesează datele din entry este `get()`, vom scrie o funcție `afiseazaDatele()` pe care o vom lega de butonul `Afiseaza`, la apăsarea acestuia în consola deschisă la rularea programului va apărea scrisă valoarea din entry.

Exemplu:

```
#!/usr/bin/env python

from Tkinter import *

def afiseazaDatele():
    print("Numele: %s" %
(E1.get()))

top = Tk()

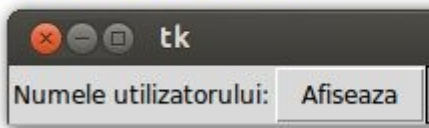
l1 = Label(top,
            text = "Numele utilizatorului:")
l1.pack(side=LEFT)

E1 = Entry(top,
            bd=5)
E1.pack(side = RIGHT)

Button(top,
        text = "Afiseaza",
        command=afiseazaDatele).pack(side=BOTTOM)

top.mainloop()
```

>>>  
Numele: salut  
Numele: salut cornel



### 13.8 Widget Frame

Frame este un widget foarte important pentru gruparea și aranjarea celorlalte elemente necesare într-o aplicație. Lucrează asemenea unui container pentru poziționarea celorlalte elemente.

Syntaxa:

```
W = Frame (master, option, ...)
```

Acestea sunt optiunile:

Option	Description
bg	The normal background color displayed behind the label and indicator.
bd	The size of the border around the indicator. Default is 2 pixels.
cursor	If you set this option to a cursor name ( <i>arrow, dot etc.</i> ), the mouse cursor will change to that pattern when it is over the checkbutton.
height	The vertical dimension of the new frame.
highlightbackground	Color of the focus highlight when the frame does not have focus.
highlightcolor	Color shown in the focus highlight when the frame has the focus.
highlightthickness	Thickness of the focus highlight.
relief	With the default value, relief=FLAT, the checkbutton does not stand out from its background. You may set this option to any of the other styles
width	The default width of a checkbutton is determined by the size of the displayed image or text. You can set this option to a number of characters and the checkbutton will always have room for that many characters.

Exemplu:

```
#!/usr/bin/env python

from Tkinter import *

class Application(Frame):
    def say_hi(self):
        print "hi there, everyone!"

    def createWidgets(self):
        self.QUIT = Button(self)
        self.QUIT["text"] = "QUIT"
        self.QUIT["fg"] = "red"
        self.QUIT["command"] = self.quit

        self.QUIT.pack({"side": "left"})

        self.hi_there = Button(self)
        self.hi_there["text"] = "Hello",
        self.hi_there["command"] = self.say_hi

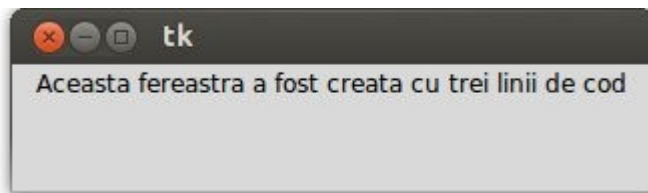
        self.hi_there.pack({"side": "left"})

    def __init__(self, master=None):
        Frame.__init__(self, master)
        self.pack()
        self.createWidgets()
```

```
root = Tk()
app = Application(master=root)
app.mainloop()
root.destroy()
```

Aceasta interfata grafica exceptand comentariile a fost scrisa in trei linii de cod si este cea mai simpla fereastră posibila:

```
#!/usr/bin/env python
#importam componentele din Tkinter
from Tkinter import Label, mainloop
#creem o eticheta cu linia de text si utilizam managerul pack()
Label(text="Aceasta fereastră a fost creata cu trei linii de
cod").pack()
# tine activata fereastră din display
mainloop()
```



Codul scris mai sus la rulare genereaza fereastră din partea stanga. Este o fereastră simpla generata cu foarte putine linii de cod dar ce este mai interesant este ca modulul care genereaza

aceasta fereastră este prezent in pachetul de programare Python si poate fi folosit imediat dupa instalarea sa.