0.1 Python Tkiner

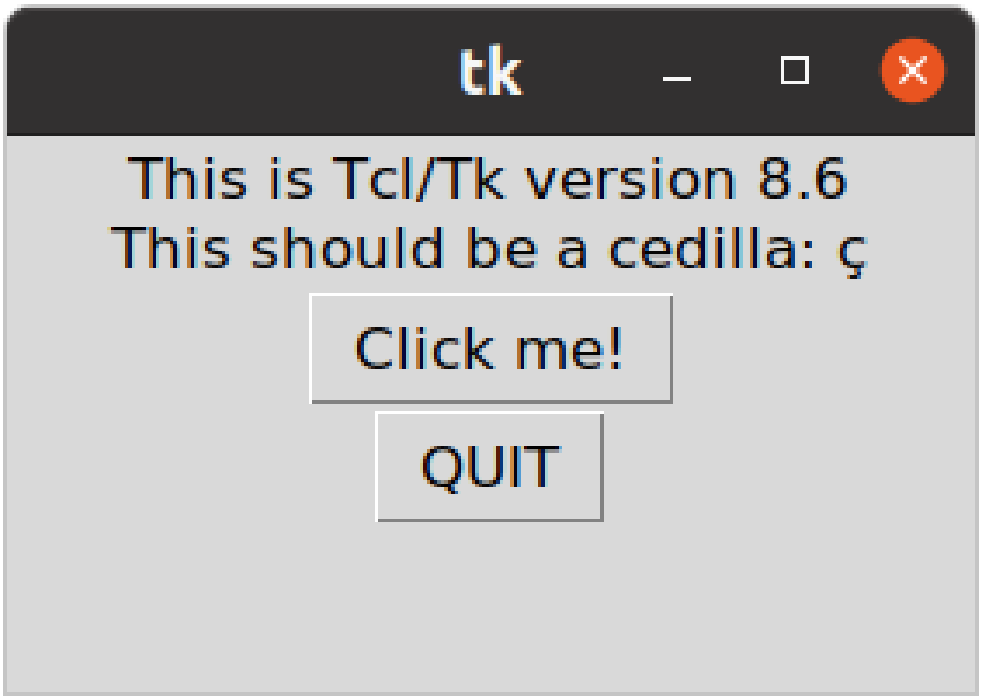
Tkinter este un modul Python pentru crearea interfețelor grafice(GUI) și este inclus în toate distribuțiile standard Python. Acest modul oferă o interfață pentru setul de instrumente Tk și funcționează pe modelul orientat obiect. Setul de instrumente Tk este o colecție multiplatformă de elemente de control grafic, cunoscute sub numele de widget-uri, pentru construirea interfețelor grafice. Pentru verificarea versiunii Tcl/Tk în terminal introduceți:

```
$ tclsh
% info patchlevel
8.6.10
```

Pentru a verifica instalarea corectă a modulului Tkinter în terminal introduceți comanda:

```
$ python3 -m tkinter
```

Această comanda deschide următoarea fereastră:



Acest modul oferă utilizatorilor Python o modalitate simplă de a crea elemente grafice folosind widgeturile găsite în setul de instrumente Tk. Widget-urile Tk pot fi folosite pentru a construi butoane, meniuri, câmpuri de date etc. într-o aplicație Python. Odată create, aceste elemente grafice pot fi

asociate sau pot interacționa cu caracteristici, funcționalități, metode, date sau chiar alte widgeturi. De exemplu, un widget de buton poate accepta clicuri de mouse și poate fi, de asemenea, programat pentru a efectua un fel de acțiune, cum ar fi ieșirea din aplicație.

Aceasta este un program Python cu Tkinter și rezultatul după rularea programului:

```
#!/usr/bin/python3
# -*- coding: utf-8 -*-
# FILE : main01.py
# RUN  : python3 main01.py
#-------------------------------------------------------------BEGIN file
from tkinter import *
from tkinter import ttk

root = Tk()

fereastra = ttk.Frame(root, padding=10)
fereastra.grid()

ttk.Label(fereastra, text="Salut").grid(column=0, row=0)
ttk.Button(fereastra, text="Inchide", command=root.destroy).grid(
    column=1, row=0)

root.mainloop()
#---------------------------------------------------------------END file
```

0.2 Python tkinter.ttk

Vechile widget-uri clasice Tkinter le-a introdus în 1991. Cele mai noi widget-uri tematice ttk au fost adăugate în 2007 cu Tk 8.5. Cele mai noi widget-uri tematice ttk înlocuiesc multe (dar nu toate) widget-uri clasice. Există diferite moduri de a importa modulul ttk. Dacă preferați ca toate widget-urile și alte caracteristici ale Tkinter și ttk să fie în spațiul dvs. de nume global, utilizați această formă de import:

```
from tkinter import *
from ttk import *
```

Este important să faceți aceste două importuri în această ordine, astfel încât toate tipurile de widget-uri de la ttk să înlocuiască widget-urile echivalente de la Tkinter. De exemplu, toate widgeturile Button vor proveni de la ttk și nu de la Tkinter. În aplicații mai complexe, unde utilizați mai multe module importate, poate îmbunătăți foarte mult lizibilitatea codului dvs. dacă practicați o igienă sigură a spațiului de nume: importați toate modulele dvs. folosind sintaxa „import modulename”. Acest lucru necesită doar un pic mai mult de tastare, dar are marele avantaj că puteți privi o referință la ceva și puteți spune de unde a venit.

```
import ttk
```

După acest import, ttk.Label este constructorul widge-tului Label, ttk.Button este un Button și așa mai departe. Dacă trebuie să faceți referire la articole din modulul Tkinter, acesta este disponibil sub numele de ttk.Tkinter. De exemplu, codul de ancorare pentru „nord-est" este ttk.Tkinter.NE. În schimb, puteți importa Tkinter separat în acest fel:

```
import Tkinter as tk
```

După această formă de import, codul pentru „nord-est" este tk.NE.

0.2.1 Button

Widget-ul Button reprezintă un element care poate fi apăsat (clik) în aplicații. De obicei, utilizați un text sau o imagine pentru a afișa acțiunea care va fi efectuată atunci când faceți clic.

Butoanele pot afișa text într-un singur font. Cu toate acestea, textul poate cuprinde mai multe linii. În plus, puteți face unul dintre caracterele subliniate pentru a marca o comandă rapidă de la tastatură.

Pentru a invoca automat o funcție sau o metodă a unei clase atunci când se face clic pe buton, atribuiți opțiunea de comandă funcției sau metodei. Aceasta se numește legarea comenzii în Tkinter.

Pentru a crea un buton, utilizați constructorul ttk.Button după cum urmează:

```
buton = ttk.Button(container, **option)
```

Un buton are multe opțiuni. Cu toate acestea, cele tipice sunt acestea:

```
buton = ttk.Button(container, text, command)
```

În această sintaxă:

* container este componenta părinte pe care plasați butonul.
* text este eticheta butonului.
* command specifică o funcție de apel invers care va fi apelată automat atunci când butonul a fost apăsat....

Buton simplu

```
#!/usr/bin/python3
#-----------------------------------------------------BEGIN file
import tkinter as tk
from tkinter import ttk
# Ferastra root
```

```
fereastra = tk.Tk()
fereastra.geometry("300x200")
fereastra.resizable(False, False)
fereastra.title("Fereastra cu button")
# Exit buton
exit_btn = ttk.Button(fereastra, text="Exit",
                      command=fereastra.quit)
exit_btn.pack(ipadx=5, ipady=5, expand=True)
fereastra.mainloop()
#-----------------------------------------------END file
```

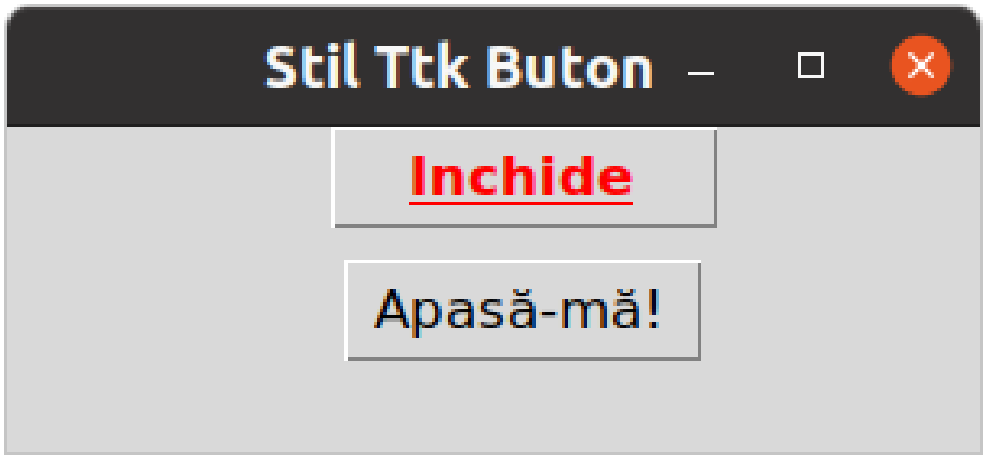
Stilizarea unui buton ttk

Stilizarea unui buton ttk, presupune crearea unui obiect Style după cum urmează:

```
stil = ttk.Style()
```

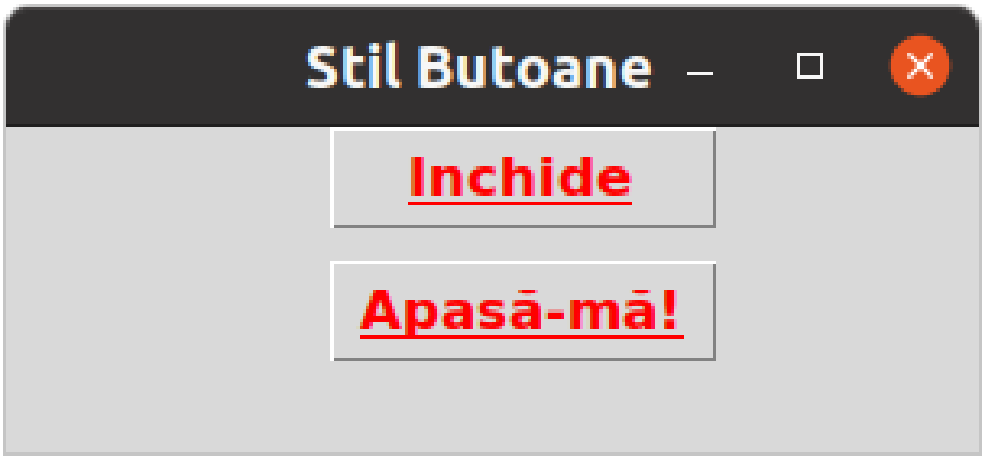
Codul de mai jos adauga stilizare doar butoanelor selectate în care trecem optiunea de stil.

```
#!/usr/bin/python3
# -*- coding: utf-8 -*-
# FILE: stilttkbuton.py
# RUN : python3 stilttkbuton.py
#-------------------------------------------------------------BEGIN file
from tkinter import *
from tkinter.ttk import *

root = Tk()
root.geometry("300x100")
root.title("Stil Ttk Buton")

stil = Style()
stil.configure("W.TButton", font =( 'calibri',
                10, 'bold', 'underline'), foreground='red')

btn1 = Button(root, text="Inchide",
               style='W.TButton',
               command=root.destroy)
btn1.grid(row=0, column=3,padx=100)

btn2 = Button(root, text="Apas -m !", command=None)
btn2.grid(row=1, column=3, pady=10, padx=100)

root.mainloop()
#---------------------------------------------------------------END file
```



Aplicarea stilului tuturor butoanelor

```
#!/usr/bin/python3
# -*- coding: utf-8 -*-
# FILE: stilttkbuton02.py
# RUN : python3 stilttkbuton02.py
#-------------------------------------------------------------BEGIN file
from tkinter import *
from tkinter.ttk import *

root = Tk()
root.geometry("300x100")
root.title("Stil Butoane")

stil = Style()
stil.configure("TButton", font =( 'calibri',
                10, 'bold', 'underline'), foreground='red')

btn1 = Button(root, text="Inchide",
               style='TButton',
               command=root.destroy)
btn1.grid(row=0, column=3,padx=100)

btn2 = Button(root, text="Apas -m !", command=None)
btn2.grid(row=1, column=3, pady=10, padx=100)

root.mainloop()
#---------------------------------------------------------------END file
```

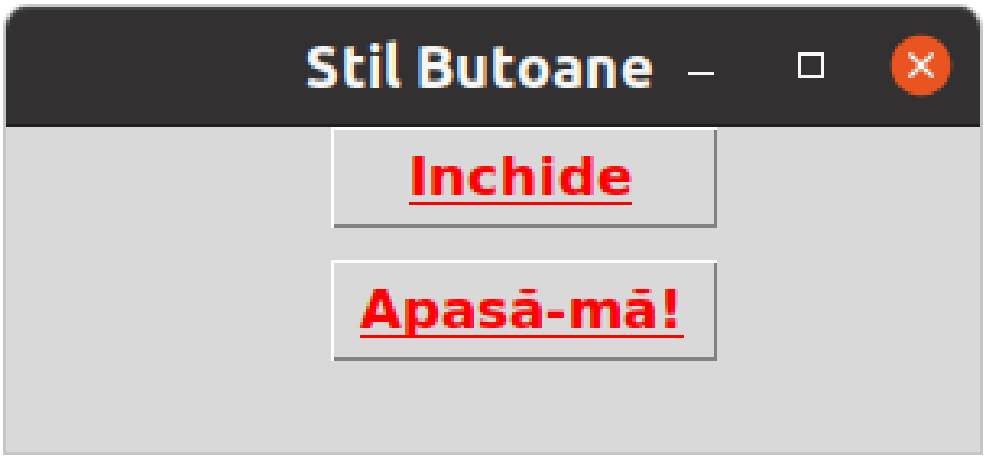


Schimbarea culorii butoanelor la trecera mouse-ului

Acum, dacă doriți să schimbați aspectul butoanelor prin mișcarea mouse-ului, adică, când trecem mouse-ul peste buton, acesta își va schimba culoarea atunci când îl apăsăm, va schimba culoarea și așa mai departe.

```
#!/usr/bin/python3
# -*- coding: utf-8 -*-
# FILE: stilttkbuton03.py
# RUN : python3 stilttkbuton03.py
#-------------------------------------------------------------BEGIN file
from tkinter import *
from tkinter.ttk import *

root = Tk()
root.geometry("300x100")
root.title("Stil Butoane")

stil = Style()
stil.configure("TButton", font =( 'calibri',
                10, 'bold', 'underline'), foreground='red')
stil.map('TButton', foreground=[('active', '!disabled', 'green')],
          background=[('active', 'black')])

btn1 = Button(root, text="Inchide",
               style='TButton',
               command=root.destroy)
btn1.grid(row=0, column=3,padx=100)
```

```
btn2 = Button(root, text="Apas -m !", command=None)
btn2.grid(row=1, column=3, pady=10, padx=100)

root.mainloop()
#------------------------------------------------------END file
```



0.2.2 Checkbox

O casetă de selectare este un widget care vă permite să bifați și debifați.

O casetă de selectare poate conține o valoare și poate invoca un apel invers atunci când este bifată sau debifată.

De obicei, utilizați o casetă de selectare atunci când doriți să cereți utilizatorilor să aleagă între două valori.

Pentru a crea o casetă de selectare, utilizați constructorul ttk.Checkbutton:

```
checkbox_var = tk.StringVar()

def check_changed():
   #...

checkbox = ttk.Checkbutton(container,
               text='<checkbox label>',
               command=check_changed,
               variable=checkbox_var,
               onvalue='<value_when_checked>',
               offvalue='<value_when_unchecked>')
```

Următorul program ilustrează modul de utilizare a unui widget casetă de selectare. După ce bifați sau debifați caseta de selectare, o casetă de mesaj va afișa valoarea activată și valoarea oprită în consecință:

```
#!/usr/bin/python3
#-----------------------------------------BEGIN file
import tkinter as tk
from tkinter import ttk
from tkinter.messagebox import showinfo

root = tk.Tk()
root.geometry('300x250')
root.resizable(False, False)
root.title("Checkbox Demo")

agreement = tk.StringVar()

def agreement_changed():
    tk.messagebox.showinfo(title="Rezultat",
                           message=agreement.get())

ttk.Checkbutton(root,
                text="De acord",
                command=agreement_changed,
                variable=agreement,
                onvalue="Acord",
                offvalue="Dezacord").pack()
root.mainloop()
#-------------------------------------------END file
```

0.2.3 Entry

Widget-ul Entry vă permite să introduceți o linie de text. În Tkinter, pentru a crea o casetă text se utilizează widget-ul Entry.

```
textbox = ttk.Entry(container, **options)
```

- container este fereastra părinte, unde este plasat widget-ul
- options sunt unul sau mai multe argumente cheie pentru a configura widget-ul

Pentru a obține textul curent al unui widget Entry se utilizează metoda get():

```
textbox.get()
```

De obicei, se asociază valoarea curentă a casetei de text cu un obiect StringVar astfel:

```
# Instanta clasei StringVar care va contine textul
text = tk.StringVar()
# Atribuirea variabilei text widget-ului Entry
textbox = ttk.Entry(root, textvariable=text)
```

În acest caz, putem utiliza apelarea metodei get() obiectului StringVar() pentru a obține valoarea curentă a widget-ului Entry:

```
text.get()
```

0.2.4 Frame

0.2.5 Label

0.2.6 LabelFrame

0.2.7 MenuButton

0.2.8 PanedWindow

0.2.9 Radiobutton

0.2.10 Checkbox

0.2.11 Scale

0.2.12 Scrollbar

0.2.13 Spinbox

0.2.14 Combobox

0.2.15 Notebook

0.2.16 Progressbar

0.2.17 Separator

0.2.18 Sizegrip

0.2.19 Treeview

Tkinter ttk `https://www.pythontutorial.net/tkinter/tkinter-ttk/`

```
print(\Hello, World!")
```

@onlineexample, title = Example.com, url = http://www.example.com,