



Tema 6.

Bara de meniu, Canvas și ferestre de dialog în Tkinter

Ce ne așteaptă?

1. Bara de meniu
2. Componenta Canvas
3. Ferestre de dialog

1. **Ce reprezintă bara de meniu și cum se creează?**
2. **Cum se plasează componentele de gradul I?**
3. **Cum se plasează componentele de gradul II și III?**
4. **Ce reprezintă componenta Canvas și cum se creează?**
5. **Cum se creează o linie, un oval și un dreptunghi în Canvas?**
6. **Ce reprezintă ferestrele FileDialog și cum se creează?**
7. **Ce reprezintă fereastra ColorChooser și cum se creează?**

1. Bara de meniu

Crearea și plasarea barei de meniu

- Componenta Menu se utilizează pentru a permite utilizatorului să gestioneze mai ușor funcționalitățile aplicației

- Bara de meniu se creează utilizând clasa Menu()

```
menu_bar=Menu(root)
```

- Plasarea barei de meniu

```
root.config(menu=menu_bar)
```

- Exemplu

```
from tkinter import *  
root=Tk()  
root.geometry('300x200')
```

```
menu_bar=Menu(root)  
root.config(menu=menu_bar)  
root.mainloop()
```

Componente de gradul I pe bara de meniu

- Componentele de gradul I sunt componentele plasate pe bara de meniu
- Componentele de gradul I se creează tot utilizând clasa Menu()
`meniu_culoare=Menu(meniu_bara)`
- Parametrul `tearoff` setează meniul să fie flotant: valoarea 0 blochează meniul flotant
- Plasarea componentei de gradul I pe bara de meniu cu metoda `add_cascade()`
`meniu_bara.add_cascade(menu=meniu_culoare, label='Culoare')`
- Parametrul `menu` – specifică componeta de gradul I ce va fi plasată pe bara de meniu
- Parametrul `label` – specifică denumirea componentei
- Exemplu de utilizare

```
meniu_bara=Menu(root)
root.config(menu=meniu_bara)
```

```
meniu_culoare=Menu(meniu_bara, tearoff=0)
meniu_bara.add_cascade(menu=meniu_culoare, label='Culoare')
```

Componente de gradul II pe bara de meniu

- Componentele de gradul II sunt componentele plasate pe componentele de gradul I
- Dacă componentele de gradul II sunt ultimele în ierarhia de componente acestea se adaugă cu metoda `add_command()` apelată de componenta de gradul mai superior

```
meniu_culoare.add_command(command=rosu, label='Rosu')
```

- Parametrul `command` – specifică funcția ce se va apela la accesarea componentei
- Parametrul `label` – specifică denumirea componentei
- Pentru plasare liniilor de separare între componentele de gradul 2 se utilizează metoda `add_separator()` apelată de componenta de gradul mai superior

```
meniu_culoare.add_separator()
```

Exemplu de meniu cu componente de gradul I și II

```
from tkinter import *
root=Tk()
root.geometry('300x200')

def rosu():
    root.config(bg='red')
def verde():
    root.config(bg='green')

meniu_bar=Menu(root)
root.config(menu=meniu_bar)

meniu_culoare=Menu(meniu_bar, tearoff=0)
meniu_bar.add_cascade(menu=meniu_culoare, label='Culoare')

meniu_culoare.add_command(command=rosu, label='Rosu')

meniu_culoare.add_separator()

meniu_culoare.add_command(command=verde, label='Verde')

root.mainloop()
```

Componente de gradul II și III

- Dacă componentele de gradul II nu sunt ultimele în ierarhia de componente acestea se creează și se adaugă identic componentelor de gradul I

```
meniu_bară=Menu(root)
root.config(menu=meniu_bară)
```

```
meniu_dimensiune1=Menu(meniu_bară, tearoff=0)
meniu_bară.add_cascade(menu=meniu_dimensiune1, label='Dimensiune')
```

```
meniu_dimensiune2=Menu(meniu_dimensiune1, tearoff=0)
meniu_dimensiune1.add_cascade(menu=meniu_dimensiune2, label='Alege dimensiunea')
```

- Dacă componentele de gradul III sunt ultimele în ierarhia de componente acestea se adaugă cu metoda `add_command()` apelată de componenta de gradul II

```
meniu_dimensiune2.add_command(label='600x500', command=mare)
meniu_dimensiune2.add_command(label='450x300', command=mediu)
meniu_dimensiune2.add_command(label='300x200', command=mic)
```


Exemplu de meniu cu componente de gradul I , II și III

```
from tkinter import *
root=Tk()
root.geometry('300x200')
def rosu():
    root.config(bg='red')
def verde():
    root.config(bg='green')
def mare():
    root.geometry('600x500')
def mediu():
    root.geometry('450x300')
def mic():
    root.geometry('300x200')
menu_bar=Menu(root)
root.config(menu=menu_bar)
menu_culoare=Menu(menu_bar, tearoff=0)
menu_bar.add_cascade(menu=menu_culoare, label='Culoare')
menu_culoare.add_command(command=rosu, label='Rosu')
menu_culoare.add_command(command=verde, label='Verde')
menu_dimensiune1=Menu(menu_bar, tearoff=0)
menu_bar.add_cascade(menu=menu_dimensiune1, label='Dimensiune')
menu_dimensiune2=Menu(menu_dimensiune1, tearoff=0)
menu_dimensiune1.add_cascade(menu=menu_dimensiune2, label='Alege dimensiunea')
menu_dimensiune2.add_command(command=mare, label='600x500')
menu_dimensiune2.add_command(command=mediu, label='450x300')
menu_dimensiune2.add_command(command=mic, label='300x200')
root.mainloop()
```

2. Componenta Canvas

Crearea și plasarea componentei - Canvas

- Reprezintă o suprafață dreptunghiulară unde pot fi desenate diferite figuri sau importate imaginii
- Se creează utilizând clasa `Canvas()`

```
canvas = Canvas (root)
```

- Plasarea componentei

```
canvas.pack ()
```

Desenarea unei linii

- Desenarea unei linii în Canvas se realizează cu metoda `create_line()`:

`linie=canvas.create_line(x0,y0,x1,y1,...xn,yn, width, fill, dash, arrow,...)`

- `x0, y0` – coordonatele punctului de start
- `x1, y1` – coordonatele punctului final
- `xn, yn` – coordonatele punctului final pentru linii cu mai multe segmente
- `width` – grosimea linii în pixeli
- `fill` – culoarea linii
- `dash` – modelului stilului linii, de exemplu (7,3,3,3) linie continua 7 pixeli apoi pauza 3 pixeli din nou 3 pixeli linie continua și 3 pixeli pauza și apoi se reia de la început
- `arrow` – prezența săgeților: FIRST, LAST, BOTH

```
canvas = Canvas(root, height=250, width=300)
linie=canvas.create_line(10,10,100,100,120, 200, width=3, fill='red',
                        dash=(30, 3, 3, 3), arrow=BOTH)
```

Desenarea unui oval

- Desenarea unui oval în Canvas se realizează cu metoda `create_oval()`:

`linie=canvas.create_oval(x0,y0,x1,y1, width, fill, dash, outline,...)`

- `x0, y0` – coordonatele punctului stânga-sus a suprafeței unde va fi plasat ovalul
- `x1, y1` – coordonatele punctului dreapta-jos a suprafeței unde va fi plasat ovalul
- `fill` – culoarea interioară a ovalului liniei
- `outline` – culoarea liniei
- `width` – grosimea liniei în pixeli
- `dash` – modelului stilului liniei, de exemplu (7,3,3,3) linie continua 7 pixeli apoi pauza 3 pixeli din nou 3 pixeli linie continua și 3 pixeli pauza și apoi se reia de la început

```
canvas = Canvas(root, height=250, width=300)
oval = canvas.create_oval(20, 20, 200, 200, width=3, fill='red',
dash=(30, 3, 3, 3), outline="blue")
```

Desenarea unui dreptunghi

- Desenarea unui dreptunghi în Canvas se realizează cu metoda `create_rectangle()`:

`linie=canvas.create_rectangle(x0,y0,x1,y1, width, fill, dash, outline,...)`

- `x0, y0` – coordonatele colțului stânga-sus a dreptunghiului
- `x1, y1` – coordonatele colțului dreapta-jos a dreptunghiului
- `fill` – culoarea interioară a ovalului liniei
- `outline` – culoarea liniei
- `width` – grosimea liniei în pixeli
- `dash` – modelului stilului liniei, de exemplu (7,3,3,3) linie continua 7 pixeli apoi pauza 3 pixeli din nou 3 pixeli linie continua și 3 pixeli pauza și apoi se reia de la început

```
canvas = Canvas(root, height=250, width=300)
```

```
oval = canvas.create_rectangle(50, 50, 150, 150, width=3, fill='red',  
dash=(30, 3), outline="blue")
```

Plasarea unei imagini

- Pentru plasare unei imagini este necesară importul acesteia cu modulul PIL (pillow) care se instalează și se importă
- Pentru importul imaginii se vor utiliza clasele ImageTK și Image

```
from PIL import ImageTk, Image
```

```
img=ImageTk.PhotoImage(Image.open('cats.png'))
```

- Plasarea imaginii în Canvas se realizează cu metoda create_image():

```
image=canvas.create_image(x0,y0, image,...)
```

- **x0, y0** – coordonatele punctului central al imaginii
- **image** – specifică variabila imaginii importate

Exemplu de plasare a imaginii

```
from tkinter import *  
  
from PIL import ImageTk, Image  
  
root=Tk()  
  
canvas = Canvas(root, height=700, width=700)  
  
img=ImageTk.PhotoImage(Image.open('cats.png'))  
  
canvas.create_image(350, 350, image = img)  
  
canvas.pack()  
  
root.mainloop()
```

3. Ferestre de dialog

Apelarea ferestrelor- `filedialog`

- `FileDialog` – permit selectarea locației de deschide sau salvare a fișierelor
- Necesită importul implicit al modulului `filedialog`

```
from tkinter import filedialog
```

- Pentru selectarea locației de deschidere se utilizează metoda `askopenfilename()`:

```
locatie=filedialog.askopenfilename(initialdir, title, filetypes)
```

- `initialdir` – locația inițială când se deschide fereastra de dialog
- `title` – titlul ferestrei de dialog
- `filetypes` – tipul fișierelor ce vor apărea în fereastra de dialog

Exemplu de selectare a unui fisier

```
from tkinter import *
from tkinter import filedialog
root=Tk()
root.geometry('600x300')

def deschide():
    calea_doc = filedialog.askopenfilename(initialdir='e:/',
        title='Selectare document word',
        filetypes=(('fisiere doc', '*.doc'),
                    ('fisiere docx', '*.docx'),
                    ('toate fisierele', '*.*')))
    label.config(text=calea_doc)
    label.pack()

buton = Button(root, text = 'Deschide un fisier word', command=deschide)
buton.pack()

label=Label(root, font='Arial 16')

root.mainloop()
```

Apelarea ferestrei- colorchooser

- ColorChooser – permit deschiderea paletei de culori și selectarea culorii dorite
- Necesită importul implicit al modulului colorchooser

```
from tkinter import colorchooser
```

- Pentru deschiderea paletei de culori se utilizează metoda askcolor():

```
culoare=colorchoser.askcolor()
```

```
from tkinter import *  
from tkinter import colorchooser  
root=Tk()  
root.geometry('400x300')
```

```
def select_color():  
    culoare = colorchooser.askcolor()  
    print(culoare)  
    root.config(bg=culoare[1])
```

```
buton = Button(root, text = 'Schimba culoarea', command=select_color)  
buton.pack()
```

```
root.mainloop()
```