

Aula 15

Resumo

O pacote tkinter (Tk interface) é a interface padrão do Python e faz parte do kit de ferramentas Tcl/Tk GUI. Tanto o Tk quanto o tkinter estão disponíveis na maioria das plataformas Unix, incluindo MacOS, bem como em sistemas Windows.

Cada widget do Tkinter possui funcionalidades próprias e comuns. Por exemplo, a configuração `activebackground` só faz sentido usar em widgets que possuem alguma interação com o usuário, como botões.

Veja um exemplo completo abaixo:

```
# importando apenas as classes que serão usadas
from tkinter import Tk, Button, Label

# instanciando um objeto da classe Tk
# e outros objetos todos de uma vez
janela = Tk() # janela
etiqueta = Label(janela) # rótulo
botao = Button(janela) # botão
```

Primeiro criamos as instâncias das classes, os objetos.

Depois definimos as configurações para cada objeto.

```
# definindo algumas configurações da janela
janela.title('Uma Janela') # título da janela
janela.geometry('300x300') # dimensões da janela
janela.resizable(False, False) # redimensionamento da janela
janela.configure(background='#000088') # cor de fundo da janela
janela.iconbitmap('git.ico') # ícone da janela
```

```
# definindo as configurações da etiqueta
etiqueta.configure(text='Uma Etiqueta') # texto
etiqueta.configure(background='#FF0000') # cor de fundo
etiqueta.configure(foreground='#FFFFFF') # cor da fonte
etiqueta.configure(font=('Verdana', 10, 'bold')) # fonte
etiqueta.configure(width=20) # largura
etiqueta.configure(height=3) # altura
etiqueta.configure(anchor='w') # alinhamento dentro da etiqueta
```

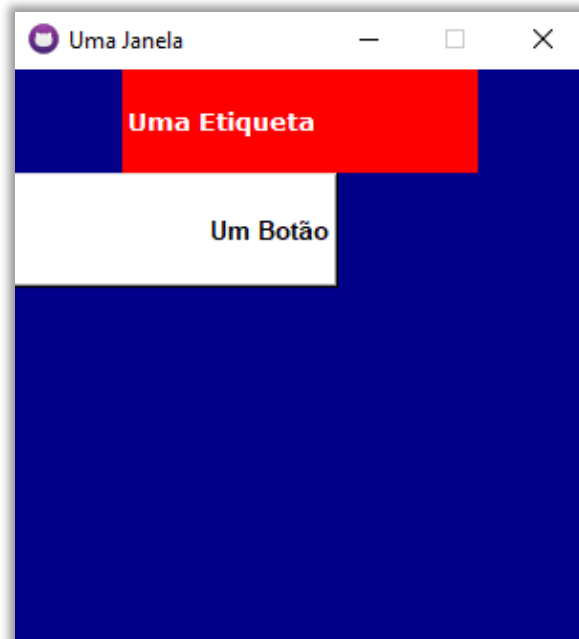
```
# definindo as configurações do botão usando uma sintaxe diferente
botao['text'] = 'Um Botão' # texto
botao['background'] = '#FFFFFF' # cor de fundo
botao['foreground'] = '#000000' # cor da fonte
botao['font'] = ('Arial', 10, 'bold') # fonte
botao['width'] = 20 # largura
botao['height'] = 3 # altura
botao['anchor'] = 'e' # alinhamento dentro do botão
# os comandos activebackground e activeforeground só funcionam quando usados
# em um objeto que possui alguma forma de interação com o usuário, como
# botões, menus, etc.
botao['activebackground'] = 'orange' # cor de fundo quando pressionado
botao['activeforeground'] = 'purple' # cor da fonte quando pressionado
```

Reparou em algo diferente acima? Também podemos definir os valores dos atributos de um objeto widget usando a notação de dicionários. O efeito é o mesmo que usar o método configure.

```
etiqueta.pack() · #·exibindo·a·etiqueta
#·quando·especificamos·o·anchor·no·pack,·estamos·posicionando·o·widget
#·em·relação·ao·seu·container,·que·no·caso·é·a·janela
botao.pack(anchor='sw') · #·exibindo·o·botão

#·método·mainloop()·é·usado·para·exibir·a·janela
janela.mainloop()
```

Isso vai gerar a seguinte janela:



Widget Entry

A classe Entry do Tkinter é utilizada para criar campos de entrada de texto na interface gráfica. Esta classe é um widget do Tkinter que herda as características básicas da classe Tk e adiciona a funcionalidade de permitir que o usuário digite e edite texto. Ele tem um funcionamento similar à função input.

Para criar um campo de entrada de texto, é preciso criar uma instância da classe Entry, passando como argumento o widget pai (geralmente a janela principal) e as opções desejadas. Alguns dos argumentos-chave da classe Entry incluem :

- master: especifica o widget pai do campo de entrada. O padrão é o widget raiz (Tk);
- show: especifica o caractere de substituição para ocultar a entrada, como "*" para senhas;
- state: especifica o estado do campo de entrada, pode ser "normal" (ativado) ou "disabled" (desativado);
- width, font, foreground, background: também são atributos existentes no Entry;

Se por acaso tiver dúvidas sobre quais são os campos de cada widget, clique com o botão esquerdo no nome da classe segurando o ctrl. Isso vai te levar ao módulo de definição da classe e, comentado logo abaixo, terá todos os campos daquele widget.

Ainda há outros comandos mais específicos que veremos em aulas futuras.

Veja um exemplo de implementação abaixo:

```
# importando apenas as classes que serão usadas
from tkinter import Tk, Entry, Label

# instanciando um objeto da classe Tk e a classe Entry
janela = Tk() # janela
entrada = Entry(janela) # campo de entrada
etiqueta = Label(janela) # etiqueta

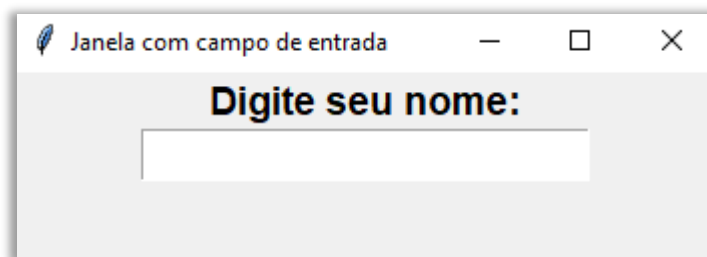
# definindo o título da janela
janela.title('Janela com campo de entrada')

# definindo as especificações da etiqueta e
# do campo de entrada
etiqueta['text'] = 'Digite seu nome:'
etiqueta['font'] = ('Arial', '14', 'bold')
entrada['font'] = ('Arial', '14', 'bold')

# posicionando os objetos na janela
etiqueta.pack()
entrada.pack()

# método mainloop() é usado para exibir a janela
janela.mainloop()
```

Ao ser executado, vai gerar a seguinte janela



Acontece que o campo Entry sozinho não tem muita utilidade, por isso que ele é usado com botões e funções para que ele tenha alguma finalidade.

Mas antes, vamos ver como funciona o comando command.

Comando command

O argumento-chave `command` do `tkinter` do `python` é usado para associar uma função ou método a um botão. Quando o botão é pressionado, a função ou método associado é chamado.

Para usar o argumento-chave `command`, você precisa primeiro criar uma função ou método que será chamado quando o botão for pressionado. Em seguida, você precisará criar o botão usando o método `tkinter Button` e passar a função ou método criado como o valor do argumento `command`.

```
from tkinter import Tk, Button

def mensagem():
    """Função que imprime uma
    mensagem no terminal"""
    print('Vou aparecer no terminal!')

janela = Tk()
botao = Button(janela)

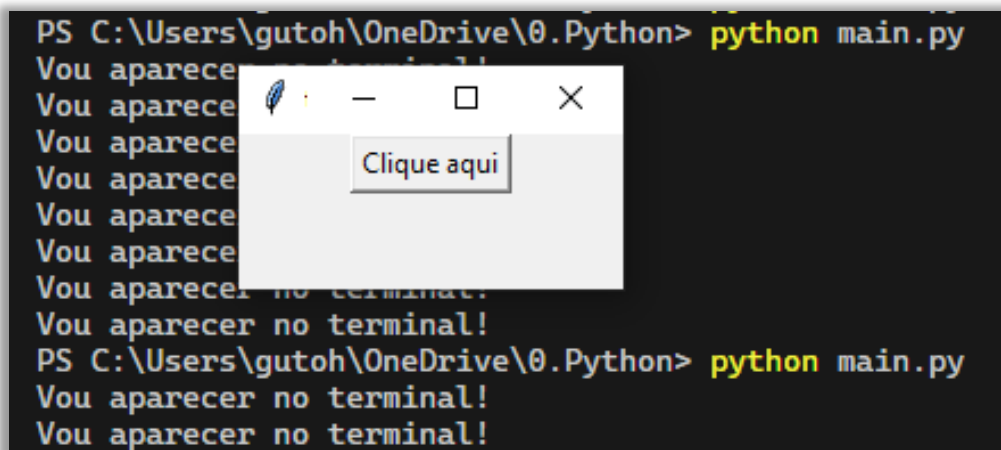
botao['text'] = 'Clique aqui'
botao['command'] = mensagem

botao.pack()

janela.mainloop()
```

Repare que a função não é chamada com os parênteses, para que ela não seja executada ao iniciar o programa.

Resultado da execução:



```
PS C:\Users\gutoh\OneDrive\0.Python> python main.py
Vou aparece
Vou aparece
Vou aparece
Vou aparece
Vou aparece
Vou aparece
Vou aparece
Vou aparece
Vou aparecer no terminal!
Vou aparecer no terminal!
PS C:\Users\gutoh\OneDrive\0.Python> python main.py
Vou aparecer no terminal!
Vou aparecer no terminal!
```

Agora, podemos usar o command com o widget Entry para captar algo que o usuário digitar e colocar em outro lugar.

```
import tkinter as tk

def atualizar_label():
    valor = campo_entrada.get()
    label_exibicao['text'] = valor

janela = tk.Tk()

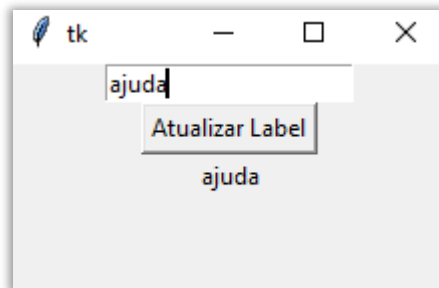
campo_entrada = tk.Entry(janela)
campo_entrada.pack()

botao_atualizar = tk.Button(janela, command=atualizar_label)
botao_atualizar['text'] = "Atualizar Label"
botao_atualizar.pack()

label_exibicao = tk.Label(janela)
label_exibicao.pack()

janela.mainloop()
```

Usando o método get do objeto Entry, podemos captar o conteúdo e colocar em um Label, por exemplo.



Assim como o input, o campo Entry SEMPRE vai retornar uma string.

Exercícios para Praticar

1. Crie uma janela com um botão que imprima uma mensagem de boas-vindas na janela de comando quando clicado.
2. Crie uma janela com um botão que mostre o valor de pi no terminal quando clicado.
3. Crie uma janela com um botão que mostre a soma de 1 até 100 no terminal quando clicado.
4. Crie uma janela com um botão que mostre os 5 primeiros números primos no terminal quando clicado.
5. Crie uma janela com um botão que mostre os 10 primeiros números da sequência de Fibonacci no terminal quando clicado.
6. Crie uma janela com um botão e um Label. Quando o botão for clicado, ele deve mudar a cor do fundo do Label para uma cor aleatória.
7. Crie uma janela com um botão que, quando clicado, altere a cor de fundo da janela para uma das 3 cores azul, vermelho ou verde.
8. Crie uma janela com um botão e um Label. Quando o botão for clicado, ele deve mudar a mensagem do Label e desativar o botão.
9. Crie uma janela com um botão e um Label. Quando o botão for clicado, ele deve mudar a mensagem do Label e desativar o botão, mas apenas se o texto do Label for uma string.
10. Crie uma janela com dois Label (um de cada lado da janela) e um botão. Quando o botão for clicado, a posição dos dois Label deve ser trocada.
11. Crie uma janela com dois campo Entry e um botão. Quando o botão for clicado, deve mostrar a soma dos dois campos Entry em um Label. Se a entrada for inválida, mostre uma mensagem no terminal (realize o mesmo para as operações de subtração, multiplicação e divisão, que não deve permitir divisão por zero).
12. Crie uma janela com um campo Entry e um botão. Quando o botão for clicado, deve calcular o fatorial do número inserido no campo Entry e mostrar o resultado em um Label.
13. Crie uma janela com um campo Entry e um botão. Quando o botão for clicado, deve calcular a área do círculo com raio igual ao número inserido no campo Entry e mostrar o resultado em um Label.
14. Crie uma janela com um campo Entry e um botão. Quando o botão for clicado, deve calcular a área do quadrado com lado igual ao número inserido no campo Entry e mostrar o resultado em um Label.
15. Crie uma janela com dois campos Entry e um botão. Quando o botão for clicado, deve calcular a área do triângulo com base e altura igual ao número inserido no campo Entry e mostrar o resultado em um Label.
16. Crie uma janela com dois campos Entry e um botão. Quando clicado, tudo que estiver no primeiro campo Entry deve ser separado em uma lista com base no separador definido no segundo campo Entry e mostrado no terminal.
17. Crie uma janela com um botão, um campo Entry e um Label. Quando o botão for clicado, o texto do campo Entry deve ser copiado para o Label.
18. Crie uma janela com um botão, um campo Entry e um Label. Quando o botão for clicado, o texto do campo Entry deve ser copiado para o Label, mas apenas se o texto do campo Entry for um número inteiro.
19. Crie uma janela com um botão e um Label. Quando o botão for clicado, ele deve mudar a cor do texto do Label.
20. Crie uma janela com um botão e um Label. Quando o botão for clicado, ele deve verificar se a idade inserida é maior que 18 anos. Se for, deve mudar o texto do Label para "Você é maior de idade", caso contrário, deve mudar o texto do Label para "Você é menor de idade".