

Aula 08

- ***Interfaces Gráficas do Usuário***

- Widgets
- Container
- Eventos

- ***Tkinter***

- Fundamentos
- Frame, Label, Entry e Button
- Tratamento de Eventos
- MessageBox

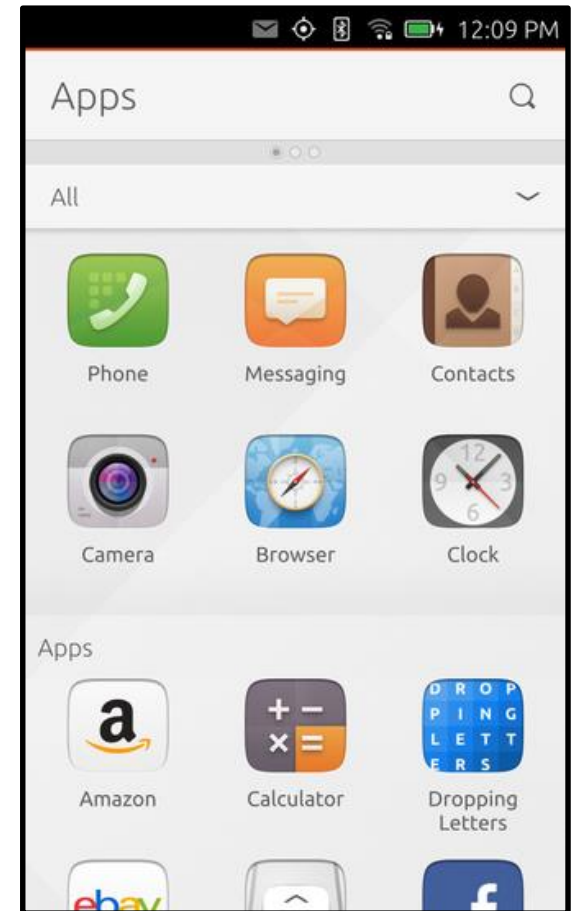


INTERFACES GRÁFICAS



Interfaces Gráficas

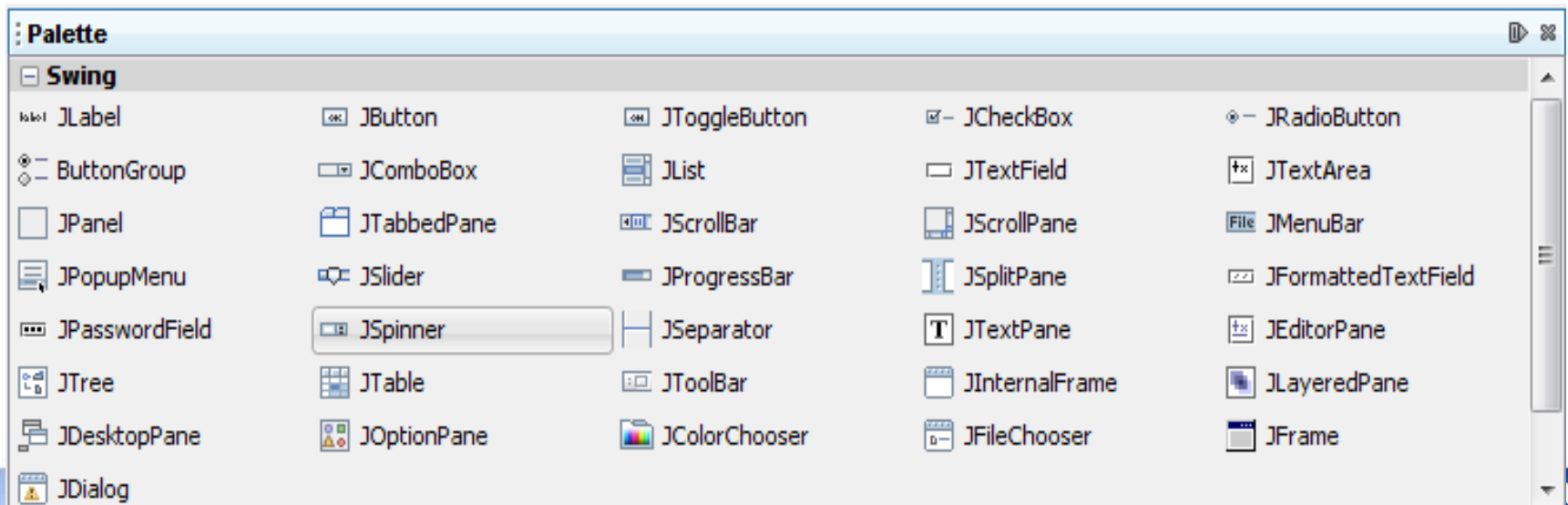
- ***Interface Gráfica do Usuário***
 - *Graphical User Interface (GUI)*
 - Tipo de interface em que a utilização ocorre por meio da interação com elementos gráficos, tais como ícones, botões, caixas de texto, entre outros.



Fonte: https://pt.wikipedia.org/wiki/Interface_gr%C3%A1fica_do_utilizador#/media/File:Ubuntu_Touch_2015-02-06_Apps_screen.png

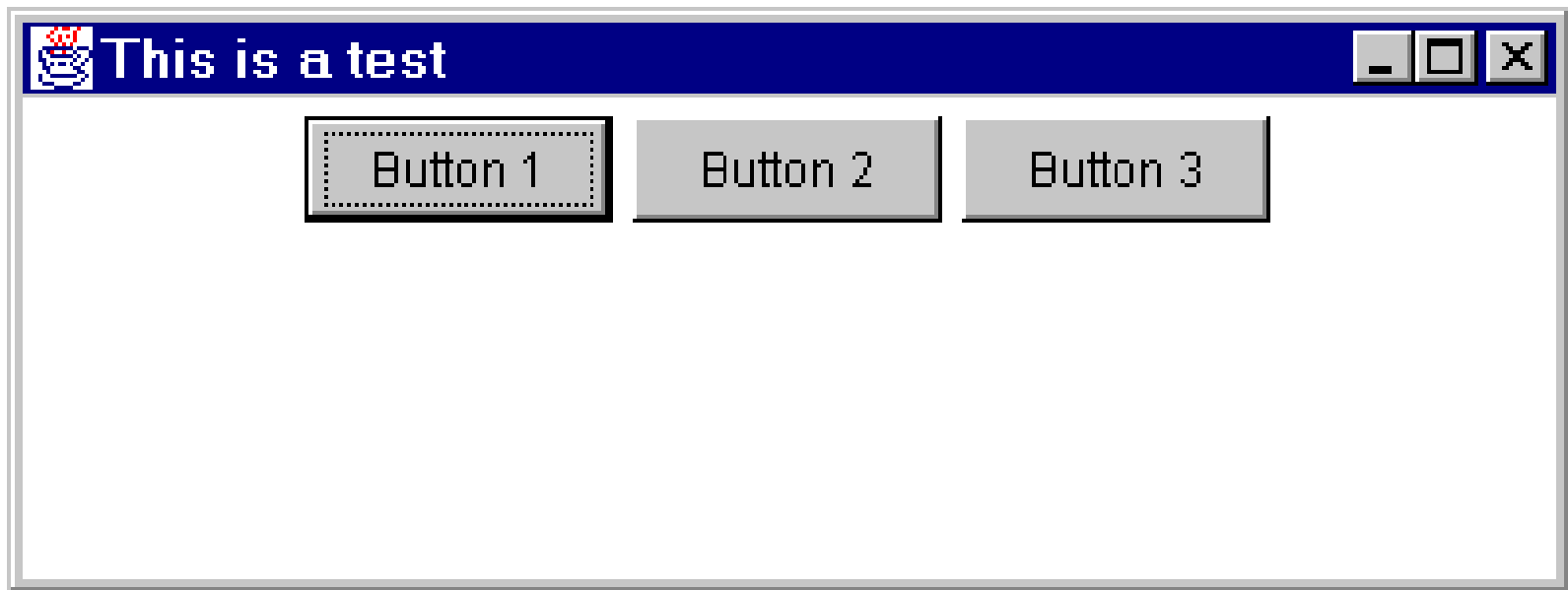
- ***Widgets***

- é um termo sem tradução que designa “***componentes de interface gráfica com o usuário***” de um modo geral.
- inclui elementos como janelas, botões, menus, ícones, listas rolantes, barras de rolagem, etc.



- ***Container***

- Elemento gráfico usado para organizar e armazenar os widgets da interface.



- ***Eventos***

- Representa o resultado da interação do usuário com a interface gráfica ou algum tipo de rotina executada pela aplicação.
- Tipos
 - Teclado (Key)
 - Pressionar uma tecla, Soltar uma tecla e outros.
 - Mouse
 - Clique, Clique Duplo e outros.
 - Janela (Window)
 - Maximizar, Minimizar, Fechar e outros.

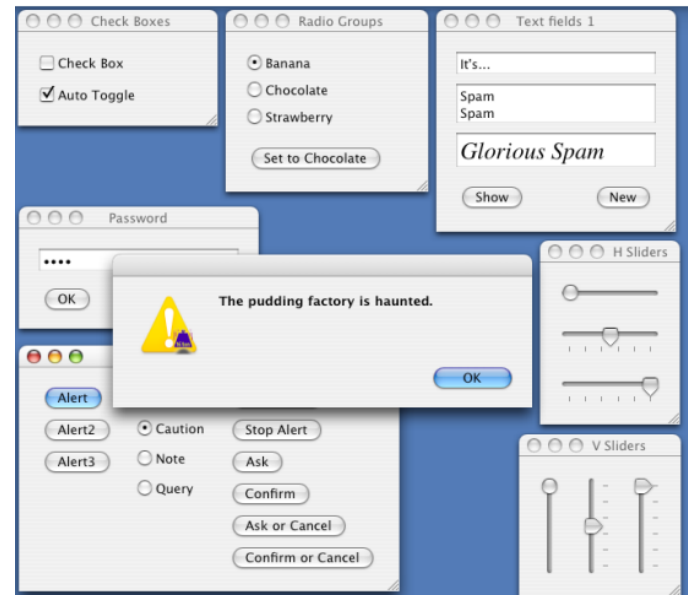
- ***Event Handler (Tratador de Eventos)***
 - Rotina responsável por especificar o que acontecerá quando um evento ocorrer.
 - Por exemplo, o que acontecerá quando o usuário clicar botão?
- ***Event Loop***
 - Procedimento que verifica, constantemente, se um evento foi disparado.
 - Caso afirmativo, busca a rotina tratadora de eventos correspondente.

Interfaces Gráficas

- ***Interfaces Gráficas em Python***

- grande número de framework para construção de interfaces.
- permite a construção de soluções *nativas* e *cross-plataforma*.

wiki.python.org/moin/GuiProgramming



- ***Interfaces Gráficas em Python***

- Principais Frameworks Gráficos

- Tkinter
- WxWidgets
- Kivy
- PyGTK
- PySide
- QT





TKINTER



- ***O que é Tkinter?***

- Biblioteca da linguagem *Python* para desenvolvimento de interfaces gráficas.
- O módulo da biblioteca acompanha a instalação padrão da linguagem.

`wiki.python.org/moin/TkInter`

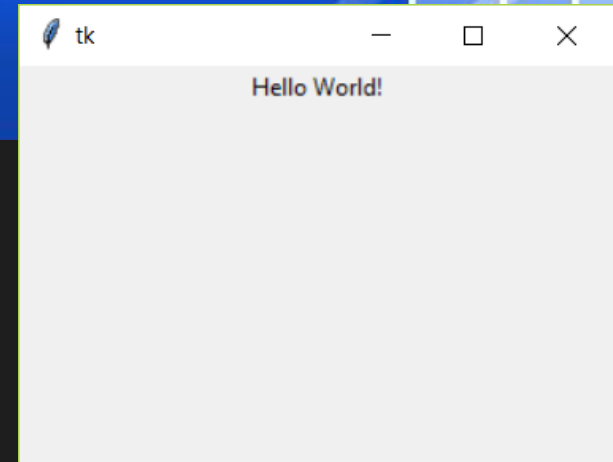
TKinter

```
from tkinter import *
```

```
class FramePrincipal(Frame):
```

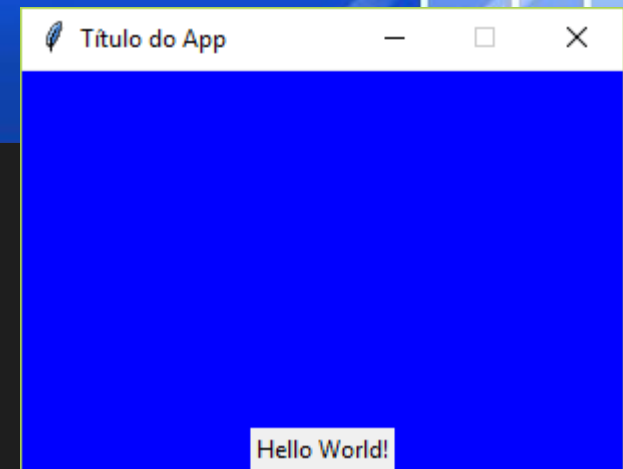
```
    def __init__(self, master=None):  
        super().__init__()  
        self.master.geometry("300x200")  
        self.pack()  
  
        self.msg = Label(self, text="Hello World!")  
        self.msg.pack()
```

```
app = FramePrincipal()  
app.mainloop()
```



TKinter

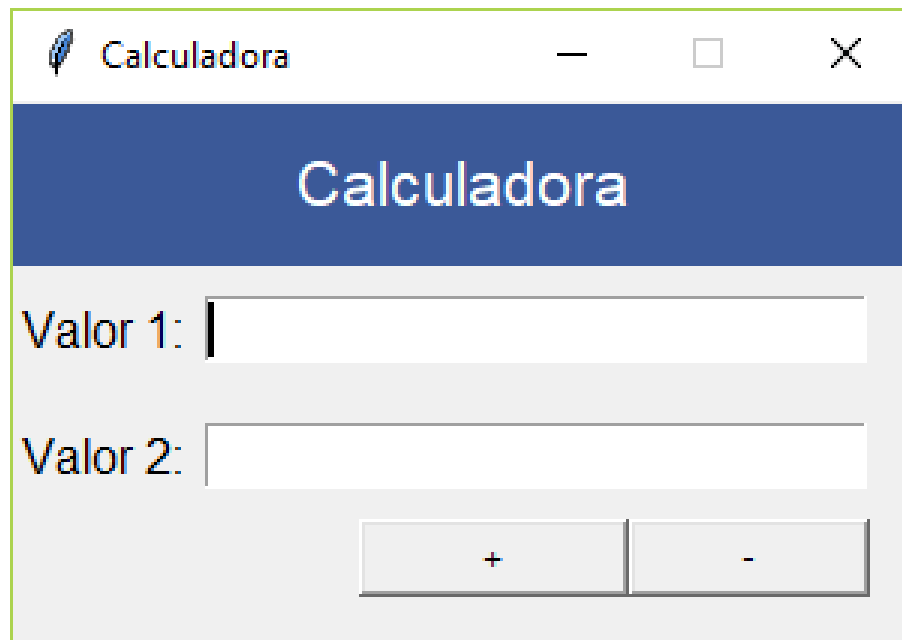
```
4  from tkinter import *
5
6  class FramePrincipal(Frame):
7
8      def __init__(self, master=None):
9          super().__init__()
10         self.master.geometry("300x200")
11         self.master.title("Título do App")
12         self.master.resizable(False, False)
13         self.pack()
14
15         self.frame1 = Frame(bg="#0000ff")
16         self.frame1.pack(expand=YES, fill=BOTH)
17
18         self.msg = Label(self.frame1, text="Hello World!")
19         self.msg.pack(side=BOTTOM)
20
21 app = FramePrincipal()
22 app.mainloop()
```



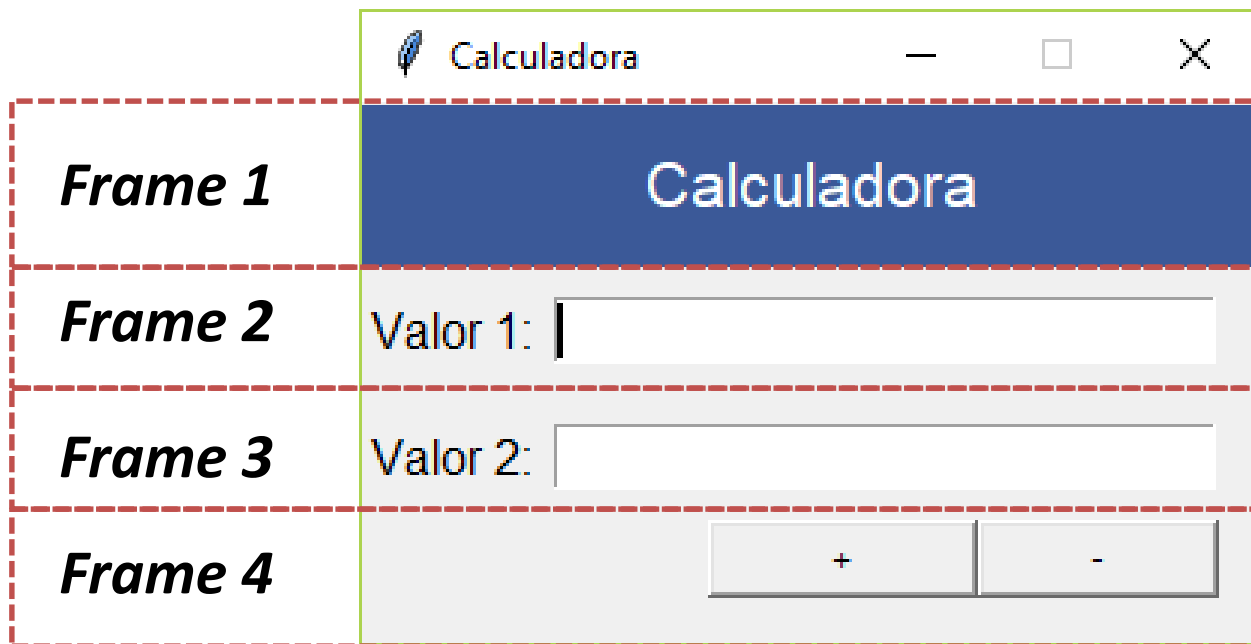
LEFT, RIGHT, TOP, BOTTOM

TKinter

- *Construindo uma **Calculadora***



- *Construindo uma **Calculadora***
 - *Gerenciador de Layout **PACK***



TKinter

- *Construindo uma Calculadora*
 - *fill = {X, Y ou BOTH}*

```
def __init__(self, master=None):  
    super().__init__()   
    self.master.geometry("300x180")  
    self.master.title("Calculadora")  
    self.master.resizable(False, False)  
    self.pack()  
  
    self.frame1 = Frame(bg="#3B5998", height=60)  
    self.frame1.pack(fill=X)  
  
    self.frame2 = Frame(bg="#FFFFFF", height=40)  
    self.frame2.pack(fill=X)  
  
    self.frame3 = Frame(bg="#FFFFFF", height=40)  
    self.frame3.pack(fill=X)  
  
    self.frame4 = Frame(bg="#FFFFFF", height=40)  
    self.frame4.pack(fill=X)
```

- *Construindo uma **Calculadora***
 - *Label*

```
self.frame1 = Frame(bg="#3B5998", height=60)
self.frame1.pack(fill=X)

self.titulo = Label(self.frame1)
self.titulo["text"] = "Calculadora"
self.titulo["bg"] = "#3B5998"
self.titulo["fg"] = "#FFFFFF"
self.titulo["font"] = "Helvetica 16 bold"
self.titulo.pack(side=TOP, ipady=15)
```

- *Construindo uma **Calculadora***

- ***Configurações Comuns***

- Width – Largura do widget;
 - Height – Altura do widget;
 - Text – Texto a ser exibido no widget;
 - Font – Família da fonte do texto;
 - Fg – Cor do texto do widget;
 - Bg – Cor de fundo do widget;
 - Side – posicionamento (Left, Right, Top, Bottom).

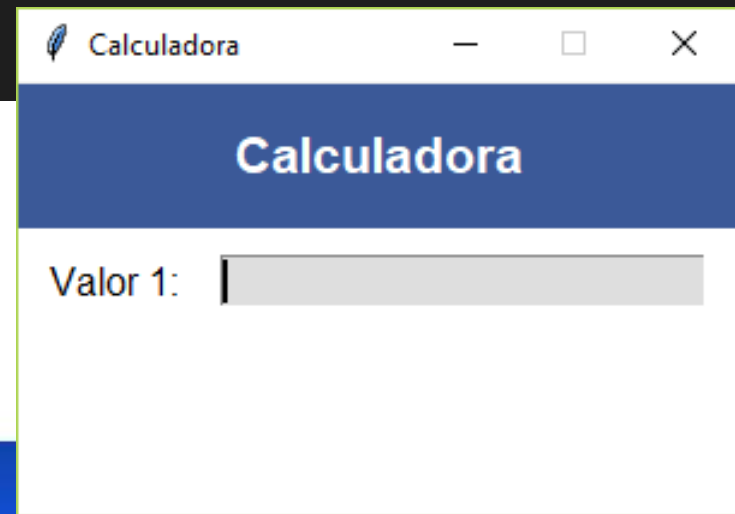
TKinter

- *Construindo uma **Calculadora***
 - ***Entry** = Entrada de Dados*

```
self.frame2 = Frame(bg="#FFFFFF", height=40)
self.frame2.pack(fill=X)

self.lbl1 = Label(self.frame2, text="Valor 1: ", bg="#FFFFFF", font=("Helvetica", 12))
self.lbl1.pack(side=LEFT, ipadx=10, ipady=10)

self.valor1 = Entry(self.frame2, width=22, bg="#DEDEDE", font=("Helvetica", 12))
self.valor1.pack(side=LEFT)
self.valor1.focus_set()
```



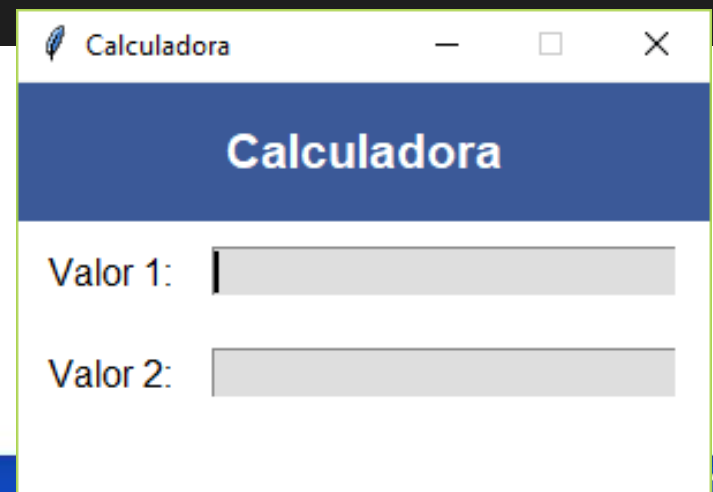
TKinter

- *Construindo uma **Calculadora***
 - ***Entry** = Entrada de Dados*

```
self.frame3 = Frame(bg="#FFFFFF", height=40)
self.frame3.pack(fill=X)

self.lbl2 = Label(self.frame3, text="Valor 2: ", bg="#FFFFFF", font=("Helvetica", 12))
self.lbl2.pack(side=LEFT, padx=10, pady=10)

self.valor2 = Entry(self.frame3, width=22, bg="#DEDEDE", font=("Helvetica", 12))
self.valor2.pack(side=LEFT)
```



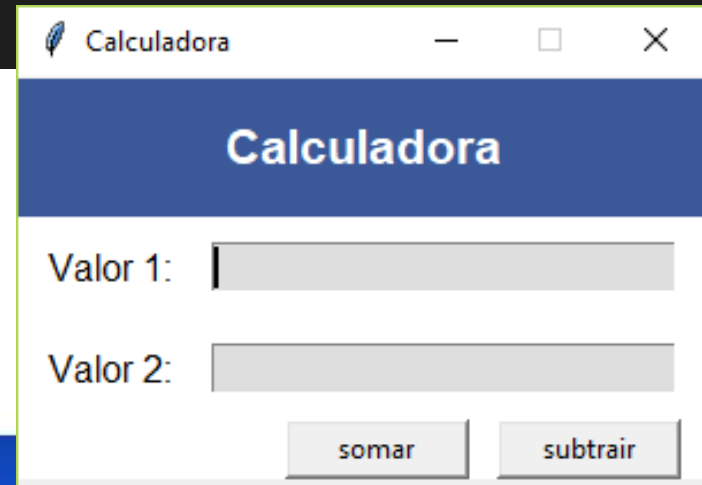
TKinter

- *Construindo uma **Calculadora***
 - ***Button** = Botão de comando*

```
self.frame4 = Frame(bg="#FFFFFF", height=40)
self.frame4.pack(fill=X)

self.btnsubtrair = Button(self.frame4, text=" subtrair ", width=10)
self.btnsubtrair.pack(side=RIGHT, padx=12)

self.btnsomar = Button(self.frame4, text=" somar ", width=10)
self.btnsomar.pack(side=RIGHT)
```



Adicionando Eventos

```
self.frame4 = Frame(bg="#FFFFFF", height=40)
self.frame4.pack(fill=X)

self.btnsubtrair = Button(self.frame4, text=" subtrair ", width=10)
self.btnsubtrair.bind("<Button-1>", self.btnSubtrairClick)
self.btnsubtrair.pack(side=RIGHT, padx=12)
```

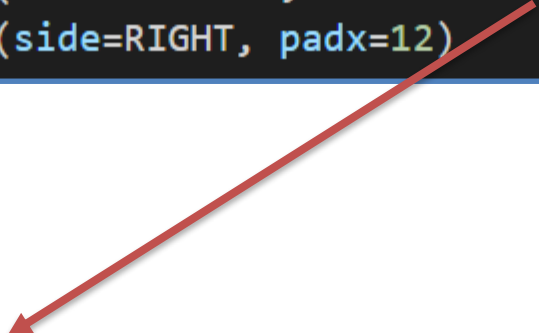
```
def btnSubtrairClick(self, event):
    v1 = float(self.valor1.get())
    v2 = float(self.valor2.get())
    mbox.showinfo("Calculadora", "Resultado %s - %s = %s" %(v1,v2,(v1-v2)))
    self.limpar_campos()
```

from tkinter **import** messagebox as *mbox*

Adicionando Eventos

```
self.frame4 = Frame(bg="#FFFFFF", height=40)
self.frame4.pack(fill=X)

self.btnsubtrair = Button(self.frame4, text=" subtrair ", width=10)
self.btnsubtrair.bind("<Button-1>", self.btnSubtrairClick)
self.btnsubtrair.pack(side=RIGHT, padx=12)
```




```
def btnSubtrairClick(self, event):
    v1 = float(self.valor1.get())
    v2 = float(self.valor2.get())
    mbox.showinfo("Calculadora", "Resultado %s - %s = %s" %(v1,v2,(v1-v2)))
    self.limpar_campos()
```

<Button-1> Botão Esquerdo, **<Button-2>** Botão do Meio e **<Button-3>** Botão Direito

Adicionando Eventos

```
self.btnsomar = Button(self.frame4, text=" somar ", width=10)
self.btnsomar.bind("<Button-1>", self.btnSomarClick)
self.btnsomar.pack(side=RIGHT)
```



```
def btnSomarClick(self,event):
    v1 = float(self.valor1.get())
    v2 = float(self.valor2.get())
    mbox.showinfo("Calculadora","Resultado %s + %s = %s" %(v1,v2,(v1+v2)))
    self.limpar_campos()

def limpar_campos(self):
    self.valor1.delete(0, 'end')
    self.valor2.delete(0, 'end')
    self.valor1.focus_set()
```

<Button-1> Botão Esquerdo, **<Button-2>** Botão do Meio e **<Button-3>** Botão Direito

Adicionando Eventos

```
self.valor2 = Entry(self.frame3, width=22,bg="#DEDEDE", font=("Helvetica", 12))
self.valor2.bind("<Return>", self.btnSomarClick)
self.valor2.pack(side=LEFT)
```

<Return> Enter

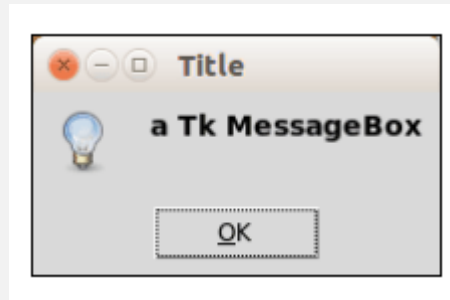
<Return>

The user pressed the Enter key. You can bind to virtually all keys on the keyboard. For an ordinary 102-key PC-style keyboard, the special keys are **Cancel** (the Break key), **BackSpace**, **Tab**, **Return**(the Enter key), **Shift_L** (any Shift key), **Control_L** (any Control key), **Alt_L** (any Alt key), **Pause**, **Caps_Lock**, **Escape**, **Prior** (Page Up), **Next** (Page Down), **End**, **Home**, **Left**, **Up**, **Right**, **Down**, **Print**, **Insert**, **Delete**, **F1**, **F2**, **F3**, **F4**, **F5**, **F6**, **F7**, **F8**, **F9**, **F10**, **F11**, **F12**, **Num_Lock**, and **Scroll_Lock**.

MessageBox

```
from tkinter import messagebox as mbox
```

```
mbox.showerror("Title", "Message")  
mbox.showwarning(" Title", "Message")  
mbox.showinfo(" Title ", "Message")
```



```
result = mbox.askokcancel("Title", "Message?")  
result = mbox.askyesno("Title", "Message?")  
result = mbox.askretrycancel("Title", "Message?")
```



ATIVIDADE PRÁTICA

- **Exercício 1**

- Desenvolver uma tela de Login que verifica o usuário e senha digitados

Window Title

Login

Usuário

Senha

entrar

→ Campo de senha

→ exibir caixa de diálogo com mensagem seja bem-vindo quando usuário=joaodasilva e senha=123456

Atividade Prática

- **Exercício 2**

- Construir uma interfaces gráfica para Cadastro de Livros

Window Title

Cadastro de Livros

Título

Autor

Editora

gravar

limpar

exibir em uma caixa de diálogo o conteúdo informado nos campos

limpar o conteúdo dos campos de texto

Atividade Prática

- **Exercício 3**

- Elaborar uma GUI para registro de notas fiscais.

The image shows a screenshot of a GUI window titled "Registro de Notas Fiscais". The window has a blue title bar with standard Windows window controls. The main content area is light gray. At the top, the title "Registro de Notas Fiscais" is displayed in bold black text. Below the title, there is a section labeled "Valor R\$" with a text input field containing "890.00" and an "OK" button. To the right of the "OK" button, an arrow points to the text "Armazenar um número indeterminado de notas". Below this section, there is a section labeled "Informações sobre as Notas". This section contains four rows, each with a label and a text input field: "Qtde. de Notas", "Maior valor R\$", "Menor valor R\$", and "Soma total R\$". To the right of each input field, an arrow points to the text "somente leitura". At the bottom of the window, there is a large button labeled "exibir".

Window Title

Registro de Notas Fiscais

Valor R\$

890.00

OK → Armazenar um número indeterminado de notas

Informações sobre as Notas

Qtde. de Notas → somente leitura

Maior valor R\$ → somente leitura

Menor valor R\$ → somente leitura


Soma total R\$ → somente leitura

exibir

TKinter

- *Entrada de Dados com **Password***

```
self.txtSenha = Entry(self.frame3, width=22)
self.txtSenha["bg"] = "#DEDEDE"
self.txtSenha["font"] = "Helvetica 12"
self.txtSenha["show"] = "*"
self.txtSenha.pack(side=LEFT)
```



Login

Usuário: joao

Senha: *****

entrar

- *Centralizar Frame*

```
def centralizar_janela(self, larg, alt):  
    px = int(self.master.winfo_screenwidth()/2 - larg/2)  
    py = int(self.master.winfo_screenheight()/2 - alt/2)-50  
    self.master.geometry("{}x{}+{}+{}".format(larg,alt,px, py))
```

Chamada do Método

```
self.centralizar_janela(400,500)
```

- ***CheckBox***


- Usado para seleções do tipo: sim ou não (ligado/desligado)

Selecione os adicionais:

☐ Bacon ☐ Molho ☐ Queijo ☐ Tomate Seco

CheckBox

```
def addCheckBox(self, chaves):  
    self.frame = Frame(bg=self._bg, height=20)  
    self.frame.pack(side=TOP, fill=X)  
  
    self.estados = []  
  
    for item in chaves:  
        var = IntVar()  
        ckb = Checkbutton(self.frame, bg=self._bg, font=self._f2)  
        ckb["text"] = item  
        ckb["variable"] = var  
        ckb["command"] = self.onPress  
        self.estados.append(var)  
        ckb.pack(side=LEFT)
```



```
def onPress(self):  
    print([var.get() for var in self.estados])
```

- ***RadioButton***

- Usado para seleções mutuamente exclusivas.

Selecione o tamanho:

☐ Pequeno ☐ Médio ☐ Grande ☒ Extra Grande

RadioButton

```
def addRadio(self, chaves):
    self.frame = Frame(bg=self._bg, height=20)
    self.frame.pack(side=TOP, fill=X)

    self.var = StringVar()
    for item in chaves:
        rdb = Radiobutton(self.frame, bg=self._bg, font=self._f2)
        rdb["text"] = item
        rdb["command"] = self.onRadioSelect
        rdb["variable"] = self.var
        rdb["value"] = item
        rdb.pack(side=LEFT)

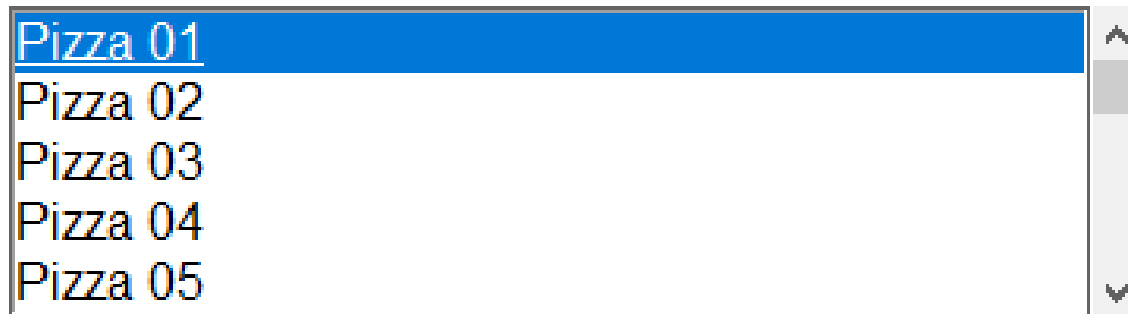
    self.var.set("Pequeno")

def onRadioSelect(self):
    print(self.var.get())
```

- **ListBox**

- Utilizado para exibir lista de objetos

Selecione a pizza desejada:



- **Listbox**

```
scrollY = Scrollbar(self.frameLista,orient=VERTICAL)

self.listaPizza = Listbox(self.frameLista,yscrollcommand=scrollY.set, height=5)
self.listaPizza["font"] = "Helvetica 12"
self.listaPizza["selectmode"] = SINGLE
self.listaPizza.bind("<<ListboxSelect>>", self.listSelection)

for i in range(1,21):
    self.listaPizza.insert(END,"Pizza %02d" %i)

self.listaPizza.pack(side=LEFT, fill=X,expand=True)

scrollY["command"] = self.listaPizza.yview
scrollY.pack(side=LEFT, fill=Y)
```

TKinter

- **Listbox**

```
scrollY = Scrollbar(self.frameLista,orient=VERTICAL)

self.listaPizza = Listbox(self.frameLista,yscrollcommand=scrollY.set, height=5)
self.listaPizza["font"] = "Helvetica 12"
self.listaPizza["selectmode"] = SINGLE
self.listaPizza.bind("<<ListboxSelect>>", self.listSelection)

for i in range(1,21):
    self.listaPizza.insert(END,"Pizza")

self.listaPizza.pack(side=LEFT, fill=X)

scrollY["command"] = self.listaPizza.yview
scrollY.pack(side=LEFT, fill=Y)
```

selectmode

Determines how many items can be selected, and how mouse drags affect the selection –

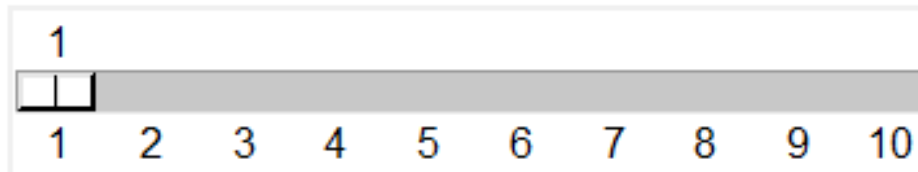
- **BROWSE** – Normally, you can only select one line out of a listbox. If you click on an item and then drag to a different line, the selection will follow the mouse. This is the default.
- **SINGLE** – You can only select one line, and you can't drag the mouse. wherever you click button 1, that line is selected.
- **MULTIPLE** – You can select any number of lines at once. Clicking on any line toggles whether or not it is selected.
- **EXTENDED** – You can select any adjacent group of lines at once by clicking on the first line and dragging to the last line.

- ***ListBox***
 - Retornar elemento selecionado

```
def listSelection(self, event):  
    pos = self.listaPizza.curselection()  
    print(self.listaPizza.get(pos))
```

- ***Scale***
 - Usado para selecionar um valor a partir de um intervalo.

Selecione a quantidade:



A Tkinter Scale widget is shown. It consists of a horizontal slider bar with a small rectangular handle on the left. Above the slider, the number '1' is displayed. Below the slider, a series of numbers from 1 to 10 are listed, corresponding to the scale's range. The slider bar is currently positioned at the value 1.

```
def addScale(self):
    self.frame = Frame(bg=self._bg, height=20)
    self.frame.pack(side=TOP, fill=X)

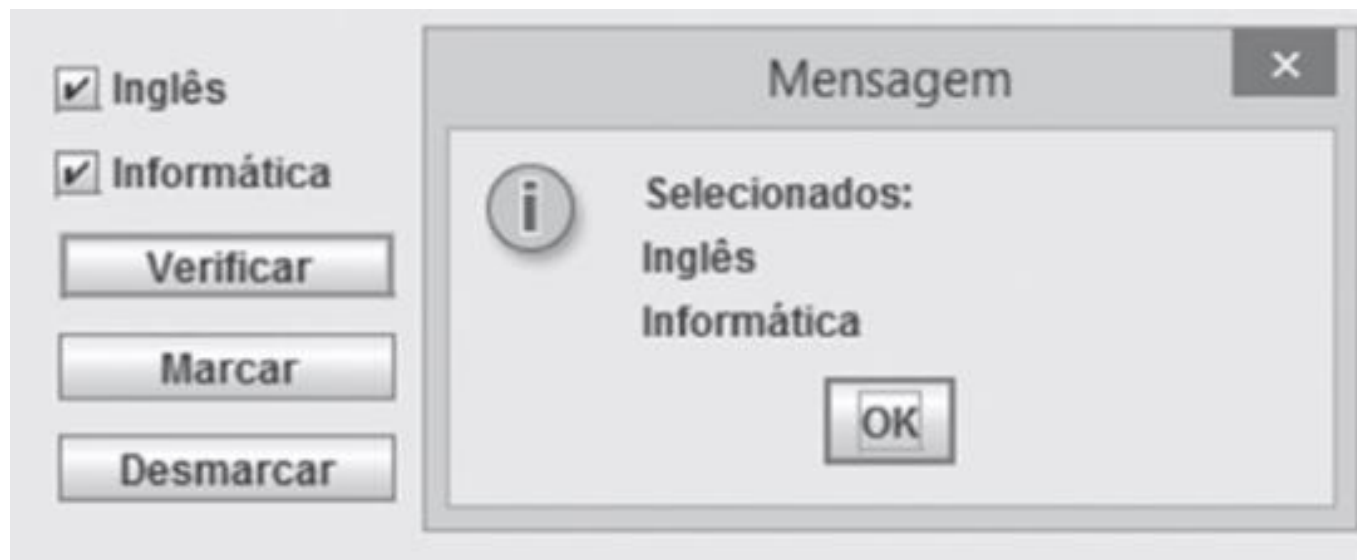
    self.escala = IntVar()
    qtde = Scale(self.frame, bg=self._bg, font=self._f2)
    qtde["from"] = 1
    qtde["to"] = 10
    qtde["variable"] = self.escala
    qtde["command"] = self.onScaleChange
    qtde["tickinterval"] = 1
    qtde["orient"] = HORIZONTAL
    qtde.pack(side=TOP, fill=X)

def onScaleChange(self, event):
    print(self.escala.get())
```

ATIVIDADE PRÁTICA

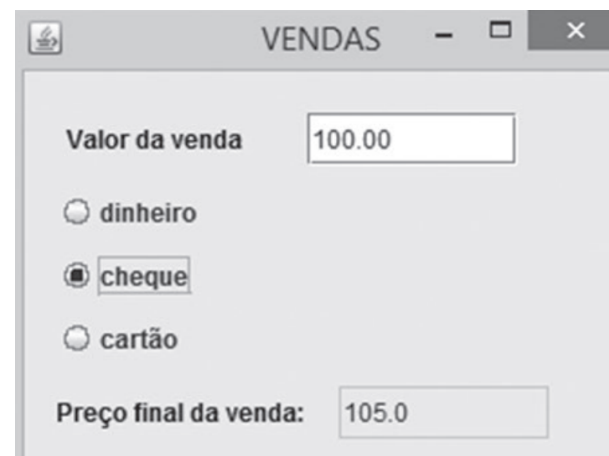
- **Exercício 1**

- Elabore uma aplicação baseada na seguinte interface.



• **Exercício 2**

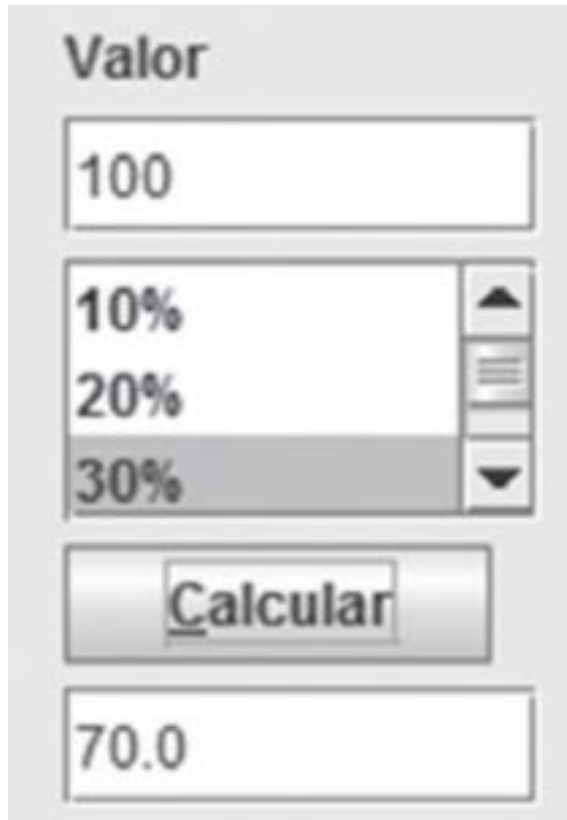
- Crie uma classe que simule vendas contendo três formas de pagamento, de acordo com a Figura.
- O usuário entra com um valor, escolhe a forma de pagamento, e o cálculo do preço final é realizado conforme os seguintes critérios:
 - dinheiro, desconto de 5%;
 - cheque, acréscimo de 5%;
 - cartão, acréscimo de 10%.



The screenshot shows a Java Swing window titled "VENDAS". Inside the window, there is a text field labeled "Valor da venda" containing the value "100.00". Below this, there are three radio buttons for payment methods: "dinheiro", "cheque", and "cartão". The "cheque" radio button is selected. At the bottom, there is a text field labeled "Preço final da venda:" containing the value "105.0".

- **Exercício 3**

- Elabore uma aplicação baseada na seguinte interface.



The image shows a Java Swing window titled "Valor". It contains the following elements:

- A text input field at the top containing the value "100".
- A dropdown menu below the input field. The menu is open, showing three options: "10%", "20%", and "30%". The "30%" option is currently selected and highlighted.
- A button labeled "Calcular" (Calculate) below the dropdown menu.
- A text output field at the bottom containing the result "70.0".