

Menus

- Podem ser associados a uma janela (menus toplevel), pulldown, popup e em cascata a partir de outro menu
- Todos são instâncias da classe Menu
- Um menu é composto de itens que podem ser
 - command quando pressionado executa uma callback
 - checkbox parecido com command, mas tem um valor booleano associado
 - radiobutton como command, mas representa um de vários estados mutuamente exclusivos
 - cascade ativa um outro menu em cascata
- Para adicionar um item a um menu, use métodos da forma add (“*tipo*”, *opções*) ou add_*tipo*(*opções*)

Menu de janela (oplevel)

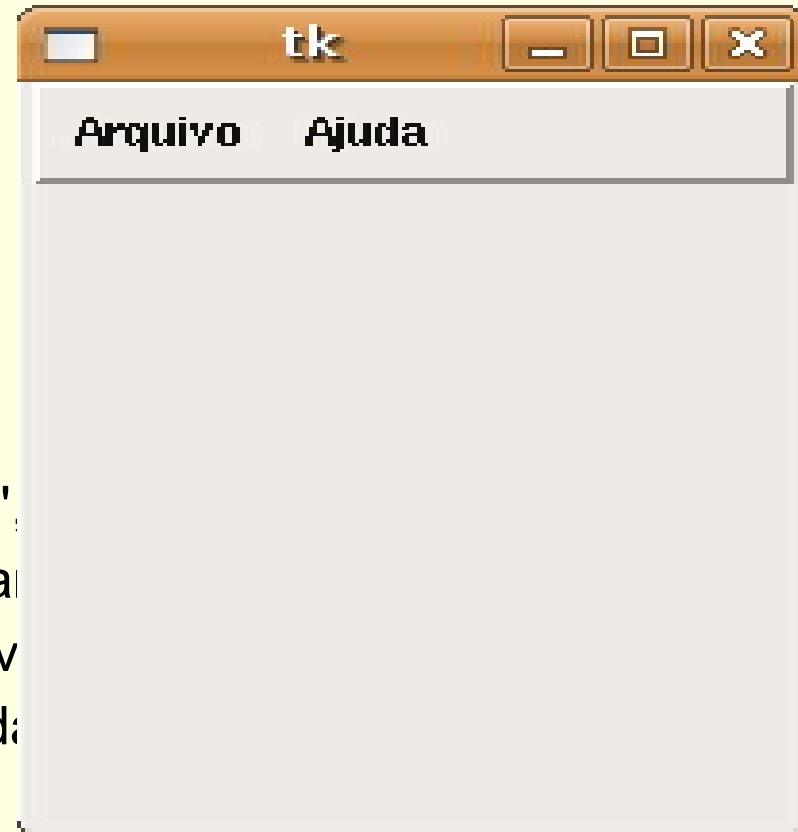
- É tipicamente exibido horizontalmente no topo da janela
 - Aspecto depende do sistema operacional
- Se um outro menu é associado como item cascade, ele é tratado como *pull-down*, isto é, é exibido sob o item do menu de janela
- Assim como outros menus, não necessita ter sua geometria gerenciada (e.g., pack ou grid)
- Para associar a uma janela, usa-se a opção menu do objeto janela.

Exemplo

```
from Tkinter import *
def abrir(): print "abrir"
def salvar(): print "salvar"
def ajuda() : print "ajuda"
top=Tk()
principal=Menu(top)
arquivo=Menu(principal)
arquivo.add_command(label="Abrir",command=abrir)
arquivo.add_command(label="Salvar",command=salvar)
principal.add_cascade(label="Arquivo",menu=arquivo)
principal.add_command(label="Ajuda",command=ajuda)
top.configure(menu=principal)
top.mainloop()
```

Exemplo

```
from Tkinter import *
def abrir(): print "abrir"
def salvar(): print "salvar"
def ajuda() : print "ajuda"
top=Tk()
principal=Menu(top)
arquivo=Menu(principal)
arquivo.add_command(label="Abrir",
arquivo.add_command(label="Salvar",
principal.add_cascade(label="Arquiv
principal.add_command(label="Ajuda
top.configure(menu=principal)
top.mainloop()
```



Exemplo

```
from Tkinter import *
def abrir(): print "abrir"
def salvar(): print "salvar"
def ajuda() : print "ajuda"
top=Tk()
principal=Menu(top)
arquivo=Menu(principal)
arquivo.add_command(label="Abrir"
arquivo.add_command(label="Salvar"
principal.add_cascade(label="Arquiv
principal.add_command(label="Ajuda
top.configure(menu=principal)
top.mainloop()
```



Menus Popup

- Um menu popup é aquele que é exibido numa janela independente
- Para que o menu seja exibido, é preciso invocar o método `post`:
 `post (x, y)`
 - onde `x` e `y` são as coordenadas do canto superior esquerdo do menu com relação ao canto superior esquerdo da tela

Exemplo

```
from Tkinter import *

def alo(): print "Alo!"

root = Tk()
menu = Menu(root, tearoff=0)
menu.add_command(label="Alo 1", command=alo)
menu.add_command(label="Alo 2", command=alo)

def popup(e): menu.post(e.x_root, e.y_root)

frame = Frame(root, width=200, height=200)
frame.pack()
frame.bind("<Button-3>", popup)
root.mainloop()
```

Exemplo

```
from Tkinter import *
```

```
def alo(): print "Alo!"
```

```
root = Tk()
```

```
menu = Menu(root, tearoff=0)
```

```
menu.add_command(label="Alo 1",
```

```
menu.add_command(label="Alo 2",
```

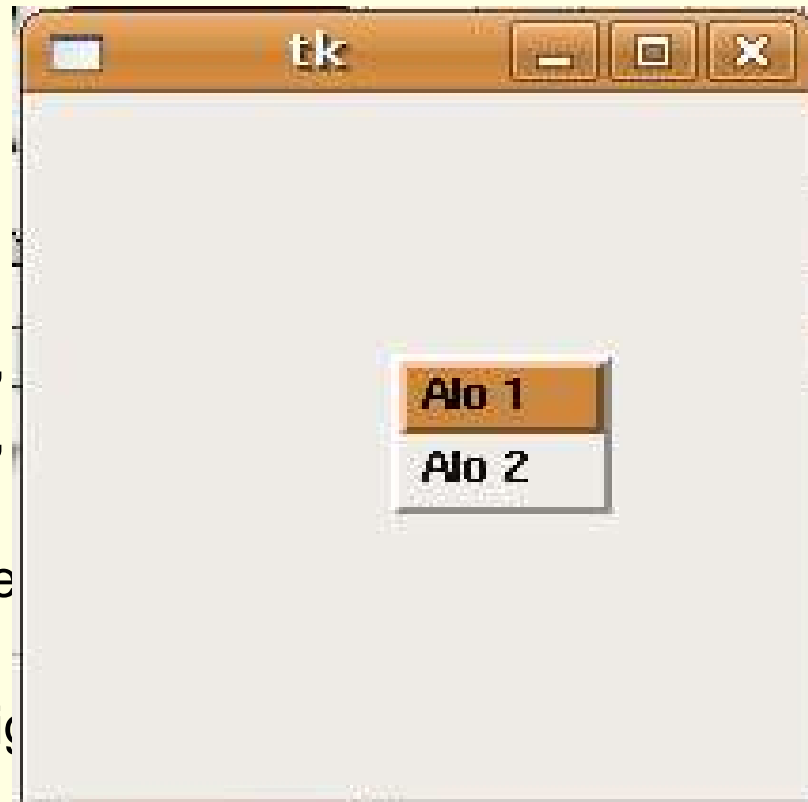
```
def popup(e): menu.post(e.x_root, e
```

```
frame = Frame(root, width=200, height=200)
```

```
frame.pack()
```

```
frame.bind("<Button-3>", popup)
```

```
root.mainloop()
```



Variáveis

- Tk é controlado por um interpretador Tcl (e não diretamente pelo python)
- Em alguns casos, deseja-se usar usar variáveis na interface
 - Por exemplo, é possível especificar que o texto exibido em um Label é o valor de uma variável (e não uma constante)
 - Nesse caso, usa-se a opção `textvar` ao invés de `text`
- Variáveis Tcl são expostas à aplicação Python através das classes `StringVar`, `IntVar` e `DoubleVar`
 - O construtor é da forma `StringVar(master)` onde `master` é uma janela ou widget
- Instâncias dessas classes possuem os métodos `get` e `set` que podem ser usados para acessar os valores armazenados no interpretador Tcl

Exemplo

```
from Tkinter import *

root = Tk()
soma = DoubleVar(root)
parcela = DoubleVar(root)
def aritmetica (e):
    soma.set(soma.get()+parcela.get())

lsoma = Label(textvar=soma)
eparcela = Entry(textvar=parcela)
eparcela.bind("<Return>", aritmetica)
lsoma.pack()
eparcela.pack()
root.mainloop()
```

Exemplo

```
from Tkinter import *
```

```
root = Tk()
soma = DoubleVar(root)
parcela = DoubleVar(root)
def aritmetica (e):
    soma.set(soma.get()+parcela.get())
```



```
lsoma = Label(textvar=soma)
eparcela = Entry(textvar=parcela)
eparcela.bind("<Return>", aritmetica)
lsoma.pack()
eparcela.pack()
root.mainloop()
```

Exemplo

```
from Tkinter import *
```

```
root = Tk()
```

```
soma = DoubleVar(root)
```

```
parcela = DoubleVar(root)
```

```
def aritmetica (e):
```

```
    soma.set(soma.get()+parcela.get())
```

```
lsoma = Label(textvar=soma)
```

```
eparcela = Entry(textvar=parcela)
```

```
eparcela.bind("<Return>", aritmetica)
```

```
lsoma.pack()
```

```
eparcela.pack()
```

```
root.mainloop()
```



Exemplo

```
from Tkinter import *
```

```
root = Tk()
```

```
soma = DoubleVar(root)
```

```
parcela = DoubleVar(root)
```

```
def aritmetica (e):
```

```
    soma.set(soma.get()+parcela.get())
```

```
lsoma = Label(textvar=soma)
```

```
eparcela = Entry(textvar=parcela)
```

```
eparcela.bind("<Return>", aritmetica)
```

```
lsoma.pack()
```

```
eparcela.pack()
```

```
root.mainloop()
```



Checkbuttons

- Checkbutton Representa uma variável que pode ter dois valores distintos (tipicamente um valor booleano). Clicando no botão alterna-se entre os valores
- A callback especificada pela opção `command` é chamada sempre que a variável muda de valor
- Estado é armazenado pela variável Tcl especificada pela opção `variable`
- Se a variável é inteira, o valor correspondente ao checkbutton “desligado”/“ligado” é 0/1
- É possível usar um checkbutton com uma variável string
 - Nesse caso, os valores correspondentes a “desligado”/“ligado” são especificados com as opções `offvalue` e `onvalue`

Exemplo

```
from Tkinter import *
```

```
root = Tk()
```

```
v1 = IntVar(root)
```

```
v2 = StringVar(root)
```

```
def exibe():
```

```
    l.config (text="v1=%d,v2=%s"%(v1.get(),v2.get()))
```

```
c1 = Checkbutton (text="V1", var=v1, command=exibe)
```

```
c2 = Checkbutton (text="V2", var=v2, command=exibe,\n                  onvalue="Sim", offvalue="Nao")
```

```
l = Label()
```

```
for w in (c1,c2,l):w.pack()
```

```
exibe()
```

```
mainloop()
```

Exemplo

```
from Tkinter import *
```

```
root = Tk()
```

```
v1 = IntVar(root)
```

```
v2 = StringVar(root)
```

```
def exibe():
```

```
    l.config (text="v1: " + v1.get() + " e v2: " + v2.get()))
```

```
c1 = Checkbutton (root, text="v1", command=exibe)
```

```
c2 = Checkbutton (root, text="v2", command=exibe, \
```

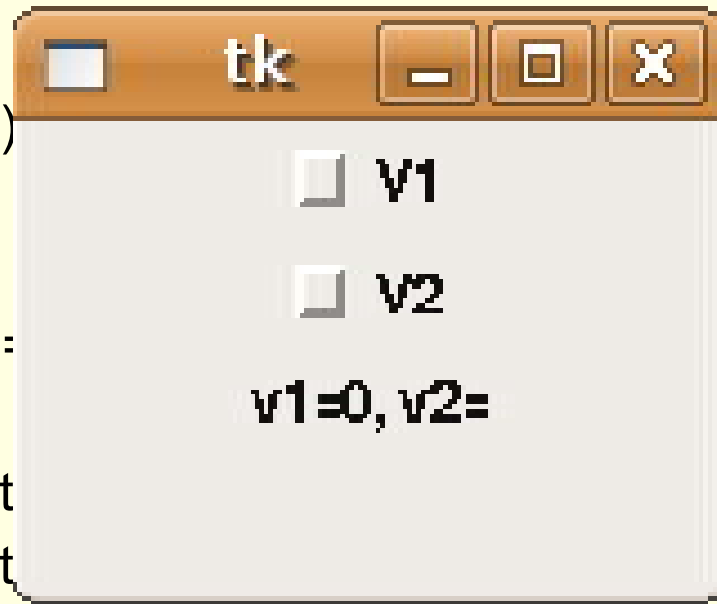
```
                  onvalue="Sim", offvalue="Nao")
```

```
l = Label()
```

```
for w in (c1,c2,l):w.pack()
```

```
exibe()
```

```
mainloop()
```



Exemplo

```
from Tkinter import *
```

```
root = Tk()
```

```
v1 = IntVar(root)
```

```
v2 = StringVar(root)
```

```
def exibe():
```

```
    l.config (text="v1=" + str(v1.get()) + " v2=" + str(v2.get()))
```

```
c1 = Checkbutton (root, text="v1=1", command=exibe)
```

```
c2 = Checkbutton (root, text="v2=1", command=exibe, \
```

```
                  onvalue="Sim", offvalue="Nao")
```

```
l = Label(root)
```

```
for w in (c1,c2,l):w.pack()
```

```
exibe()
```

```
mainloop()
```



Exemplo

```
from Tkinter import *
```

```
root = Tk()
```

```
v1 = IntVar(root)
```

```
v2 = StringVar(root)
```

```
def exibe():
```

```
    l.config (text="v1=
```

```
c1 = Checkbutton (t
```

```
c2 = Checkbutton (t
```

```
        onvalue="Sim", offvalue="Nao")
```

```
l = Label()
```

```
for w in (c1,c2,l):w.pack()
```

```
exibe()
```

```
mainloop()
```



Radiobuttons

- Radiobutton representa um possível valor de uma variável que tem um de muitos valores. Clicando o botão, a variável assume aquele valor
- A variável é especificada com a opção `variable` e o valor associado com a opção `value`
- Os radiobuttons que se referem à mesma variável funcionam em conjunto
 - Ex.: ligar um faz com que outro seja desligado
- Um radiobutton é mostrado com um indicador ao lado
 - Pode-se desabilitar o indicador usando a opção `indicatoron=False`
 - Nesse caso, é mostrado como um botão normal

Exemplo

```
from Tkinter import *
root=Tk()
cor = StringVar(root)
cor.set("black")
l = Label(background=cor.get())
l.pack(fill='both',expand=True)
def pinta(): l.configure(background=cor.get())
for txt,val in (("preto","black"),
                ("vermelho","red"),
                ("azul","blue"),
                ("verde","green")):
    Radiobutton(text=txt,value=val,variable=cor,
                command=pinta).pack(anchor=W)
mainloop()
```

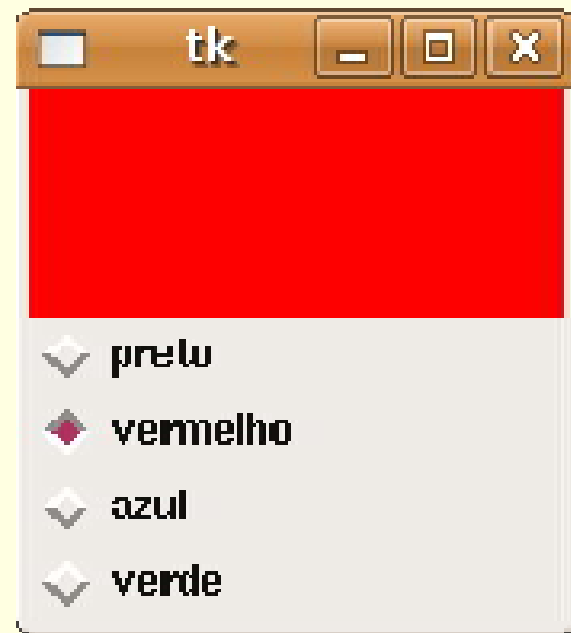
Exemplo

```
from Tkinter import *
root=Tk()
cor = StringVar(root)
cor.set("black")
l = Label(background=cor.get())
l.pack(fill='both',expand=True)
def pinta(): l.configure(background=cor.get())
for txt,val in (("preto","black"),
                ("vermelho","red"),
                ("azul","blue"),
                ("verde","green")):
    Radiobutton(text=txt,value=val,variable=cor,
                command=pinta).pack(anchor=W)
mainloop()
```



Exemplo

```
from Tkinter import *
root=Tk()
cor = StringVar(root)
cor.set("black")
l = Label(background=cor.get())
l.pack(fill='both',expand=True)
def pinta(): l.configure(background=cor.get())
for txt,val in (("preto","black"),
                ("vermelho","red"),
                ("azul","blue"),
                ("verde","green")):
    Radiobutton(text=txt,value=val,variable=cor,
                command=pinta).pack(anchor=W)
mainloop()
```



Exemplo

```
from Tkinter import *
root=Tk()
cor = StringVar(root)
cor.set("black")
l = Label(background=cor.get())
l.pack(fill='both',expand=True)
def pinta(): l.configure(background=cor.get())
for txt,val in (("preto","black"),
                ("vermelho","red"),
                ("azul","blue"),
                ("verde","green")):
    Radiobutton(text=txt,value=val,variable=cor,
                command=pinta).pack(anchor=W)
mainloop()
```

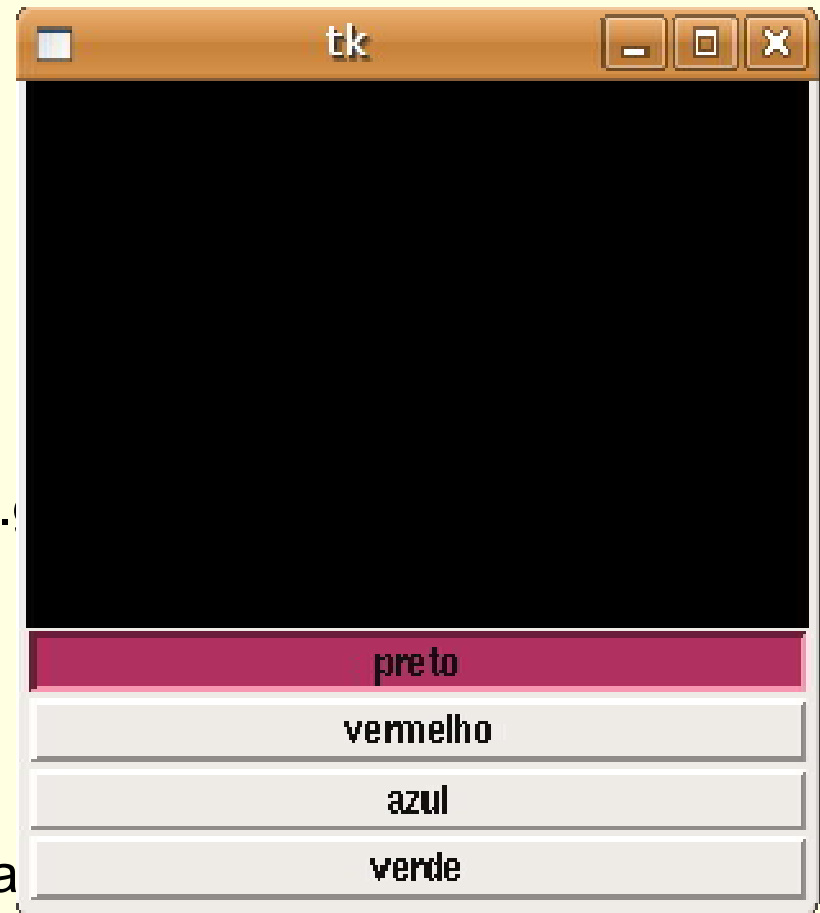


Exemplo

```
from Tkinter import *
root=Tk()
cor = StringVar(root)
cor.set("black")
l = Label(background=cor.get())
l.pack(fill='both',expand=True)
def pinta(): l.configure(background=cor.get())
for txt,val in (("preto","black"),
                ("vermelho","red"),
                ("azul","blue"),
                ("verde","green")):
    Radiobutton(text=txt,value=val,variable=cor,
                command=pinta,indicatoron=False).pack(fill='x')
mainloop()
```


Exemplo

```
from Tkinter import *
root=Tk()
cor = StringVar(root)
cor.set("black")
l = Label(background=cor.get())
l.pack(fill='both',expand=True)
def pinta(): l.configure(background=cor.get())
for txt,val in (("preto","black"),
                ("vermelho","red"),
                ("azul","blue"),
                ("verde","green")):
    Radiobutton(text=txt,value=val,variable=cor,
                command=pinta,indicatoron=False).pack(fill='x')
mainloop()
```



Entry

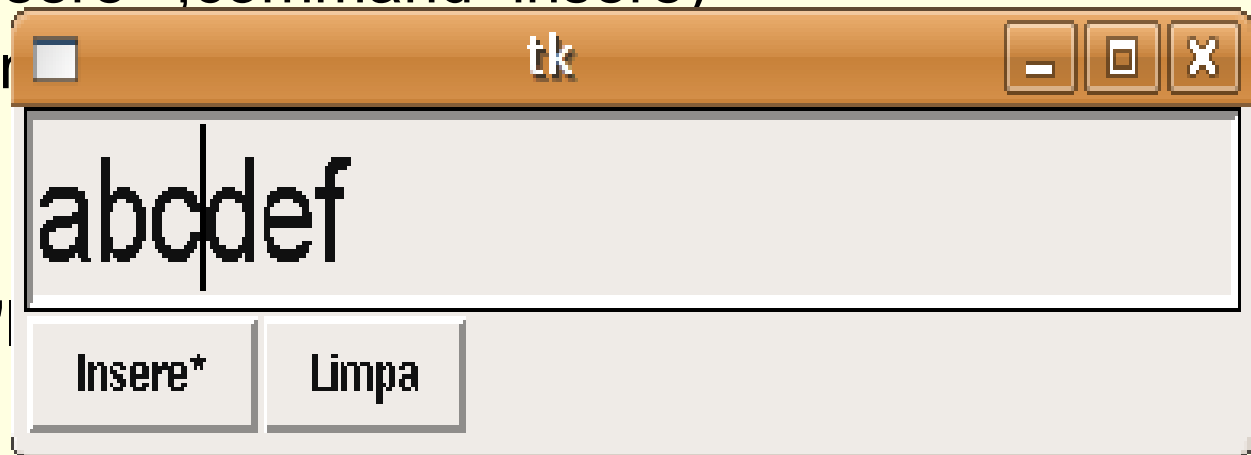
- Um Entry permite entrada/edição de uma linha de texto
- O texto associado ao Entry é normalmente armazenado numa variável indicada pela opção `textvariable`
 - Se não indicada, é usada uma variável interna cujo valor pode ser obtido usando o método `get()`
- Há diversos métodos para manipular diretamente o texto
 - Usam o conceito de índices (não confundir com os índices usado pelo Python)
 - Por exemplo, o índice `INSERT` indica a posição do texto onde o cursor de inserção se encontra, 0 a posição antes do primeiro caractere e `END` a posição ao final do texto

Exemplo

```
from Tkinter import *
def insere(): e.insert(INSERT,"*")
def limpa(): e.delete(INSERT,END)
e=Entry(font="Arial 24")
i=Button(text="Insere*",command=insere)
l=Button(text="Limpa",command=limpa)
e.pack()
for w in (i,l):
    w.pack(side='left')
mainloop()
```

Exemplo

```
from Tkinter import *
def insere(): e.insert(INSERT,"*")
def limpa(): e.delete(INSERT,END)
e=Entry(font="Arial 24")
i=Button(text="Insere*",command=insere)
l=Button(text="Limpa",command=limpa)
e.pack()
for w in (i,l):
    w.pack(side='left')
mainloop()
```



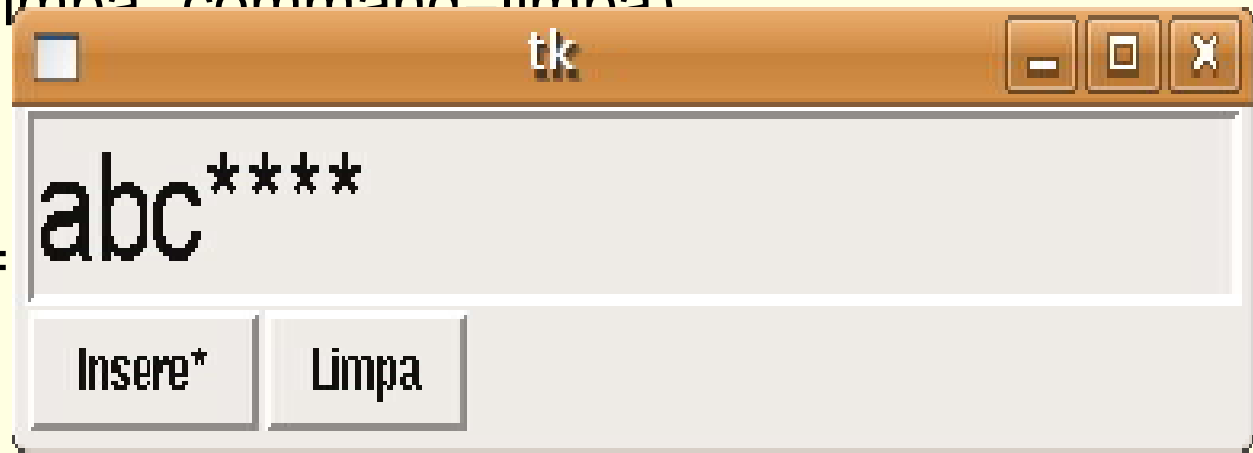
Exemplo

```
from Tkinter import *
def insere(): e.insert(
def limpa(): e.delete(
e=Entry(font="Arial 2
i=Button(text="Insere
l=Button(text="Limpa
e.pack()
for w in (i,l):
    w.pack(side='left')
mainloop()
```



Exemplo

```
from Tkinter import *
def insere(): e.insert(INSERT,"*")
def limpa(): e.delete(INSERT,END)
e=Entry(font="Arial 24")
i=Button(text="Insere*",command=insere)
l=Button(text="Limpa",command=limpa)
e.pack()
for w in (i,l):
    w.pack(side=
mainloop()
```



Canvas

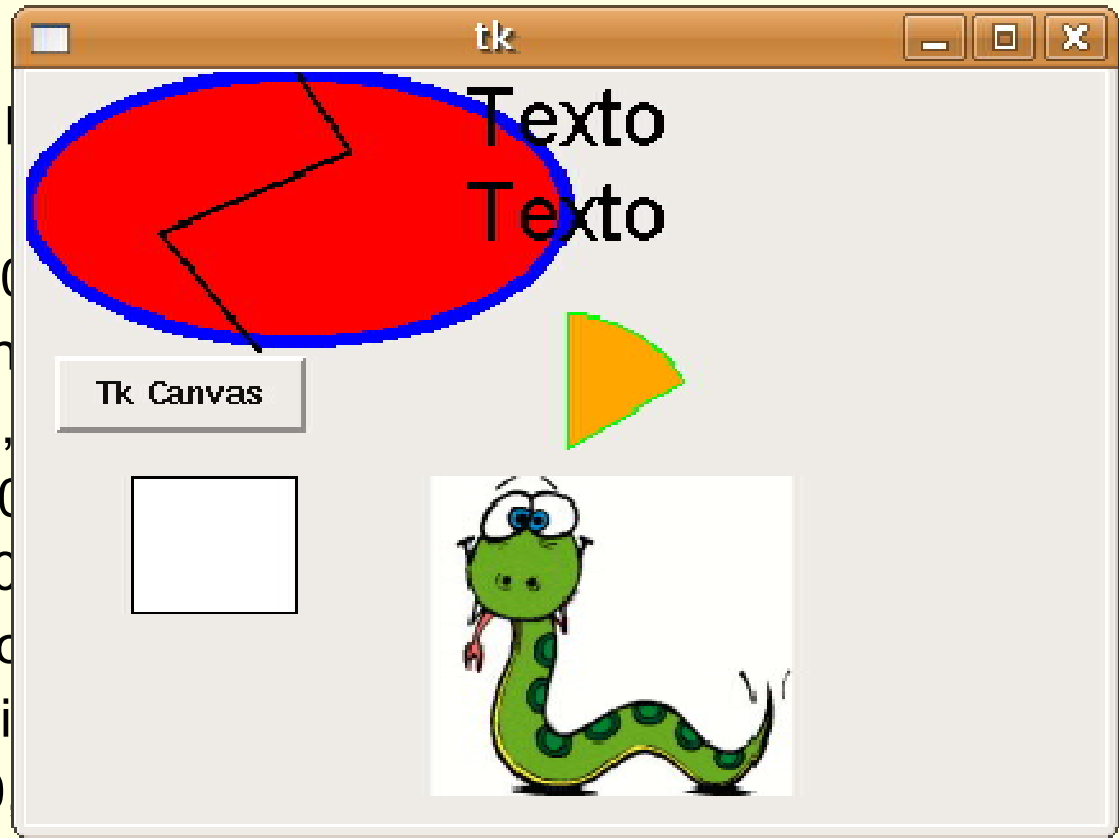
- Permite a exibição e edição de gráficos estruturados 2D
- Elementos gráficos (itens) são introduzidos usando métodos da forma `create_tipo (...)`, onde *tipo* pode ser
 - `arc` arco de círculo
 - `bitmap` imagem binária
 - `image` imagem colorida
 - `line` linha poligonal
 - `oval` círculos e elipses
 - `polygon` polígonos
 - `rectangle` retângulo
 - `text` texto
 - `window` um widget tk

Exemplo

```
from Tkinter import *
root = Tk()
root.geometry("512x512")
c = Canvas(root, width=512, height=512)
c.pack()
o = c.create_oval(1,1,200,100,outline="blue",width=5,fill="red")
widget = Button(text="Tk Canvas")
w = c.create_window(10,120,window=widget,anchor=W)
l = c.create_line(100,0,120,30,50,60,100,120,fill="black",width=2)
r = c.create_rectangle(40,150,100,200,fill="white")
img = PhotoImage(file="python.gif")
i = c.create_image (150,150,image=img,anchor=NW)
a = c.create_arc (150,90,250,190,start=30,extent=60,\
                 outline="green",fill="orange")
t = c.create_text(200,35,text="Texto\nTexto",font="Arial 22")
mainloop()
```


Exemplo

```
from Tkinter import *
root = Tk()
root.geometry("512x512")
c = Canvas(root, width=512, height=512)
c.pack()
o = c.create_oval(1,1,200,100,fill="red",outline="blue")
widget = Button(text="Tk Canvas", cursor="hand2")
w = c.create_window(10,120,window=widget)
l = c.create_line(100,0,120,30,fill="black")
r = c.create_rectangle(40,150,100,250,fill="white",outline="black")
img = PhotoImage(file="python.png")
i = c.create_image(150,150,image=img)
a = c.create_arc(150,90,250,270,
                 outline="green",fill="orange")
t = c.create_text(200,350,text="Texto\nTexto",font="Arial 22")
mainloop()
```



Coordenadas de Itens

- Todos os métodos `create_item` têm como primeiros argumentos um par de coordenadas `x,y` do item
 - Os itens `oval` e `rectangle` requerem mais um par de coordenadas para delimitar a extensão (caixa envolvente)
 - Os itens `line` e `polygon` podem ser seguidos por outros pares de coordenadas que especificam demais vértices
- As coordenadas referem-se a um sistema de coordenadas próprio que pode ser diferente do da janela
 - A área do canvas que deve ser mostrada na janela pode ser modificada pela opção `scrollarea=(xmin,ymin,xmax,ymax)`
 - Para obter as coordenadas do canvas dadas as coordenadas da janela, usam-se os métodos `canvasx(x)` e `canvasy(y)`

Identificação de Itens

- Todo item de um canvas tem um identificador numérico que é retornado pelo método `create_item`
- Pode-se também associar tags (etiquetas) a itens
 - Usa-se a opção `tags=tags` onde `tags` pode ser uma string ou uma tupla com várias strings
 - Uma mesma etiqueta pode ser associada a mais de um item
- O identificador ALL refere-se a todos os itens do canvas
- O identificador CURRENT refere-se ao item do canvas sob o cursor do mouse
 - Usado em callbacks de canvas para alterar propriedades dos itens clicados

Métodos de Canvas

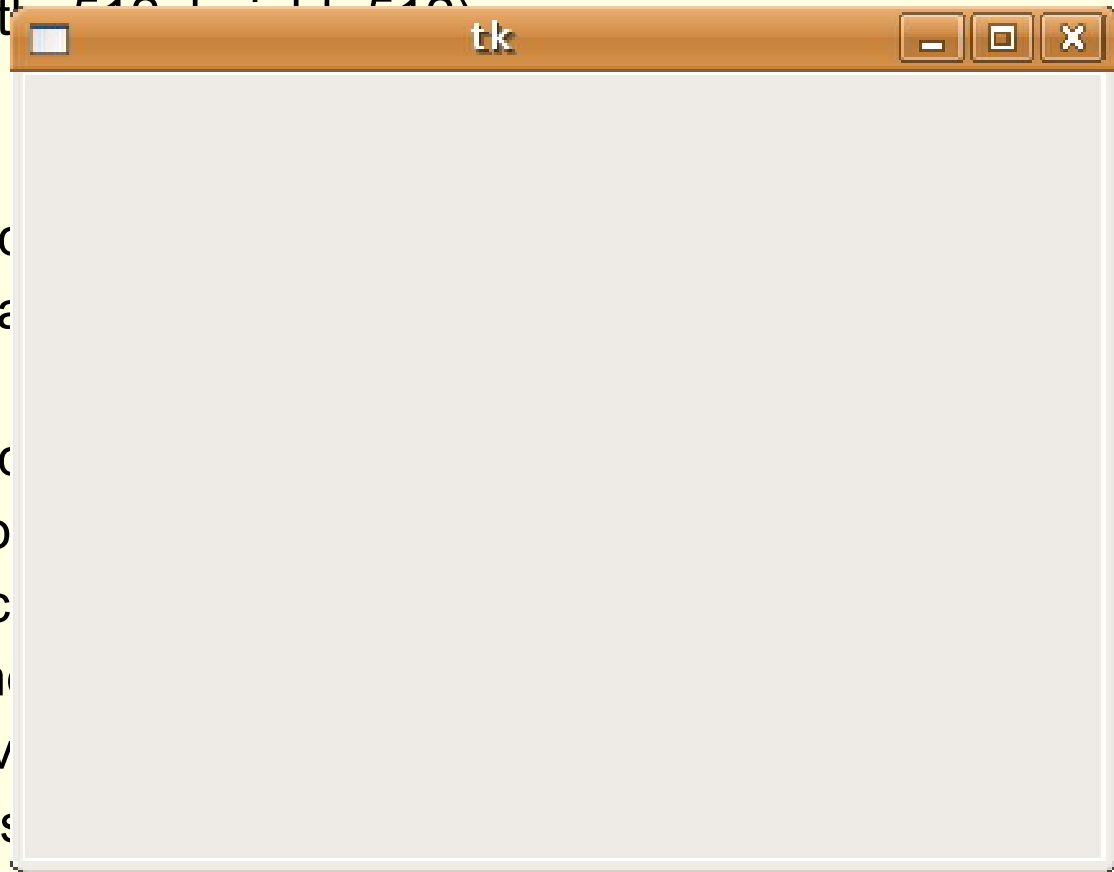
- `itemconfig (itemOuTag, ...)` altera opções do(s) item(s)
- `tag_bind(itemOuTag, padrão, callback)` associa uma *callback* a um *padrão* de eventos sobre o(s) item(s)
- `delete(itemOuTag)` remove o(s) item(s)
- `move(itemOuTag, dx,dy)` translada o(s) item(s)
- `coords(itemOuTag, x1,x2,..xN,yN)` altera as coordenadas do(s) item(s)
- `coords(item)` retorna as coordenadas do item
- `bbox(itemOuTag)` retorna uma tupla com a caixa envolvente dos itens
- `itemcget(item,opção)` retorna o valor da *opção* dada do *item*

Exemplo

```
from Tkinter import *
master = Tk()
c = Canvas(master, width=512, height=512)
c.pack()
def novalinha(e):
    x,y = c.canvasx(e.x), c.canvasy(e.y)
    c.create_line(x,y,x,y,tags="corrente")
def estendelinha(e):
    x,y = c.canvasx(e.x), c.canvasy(e.y)
    coords = c.coords("corrente") + [x,y]
    c.coords("corrente",*coords)
def fechalinha(e): c.itemconfig("corrente",tags=())
c.bind("<Button-1>", novalinha)
c.bind("<B1-Motion>", estendelinha)
c.bind("<ButtonRelease-1>", fechalinha)
c.pack()
```

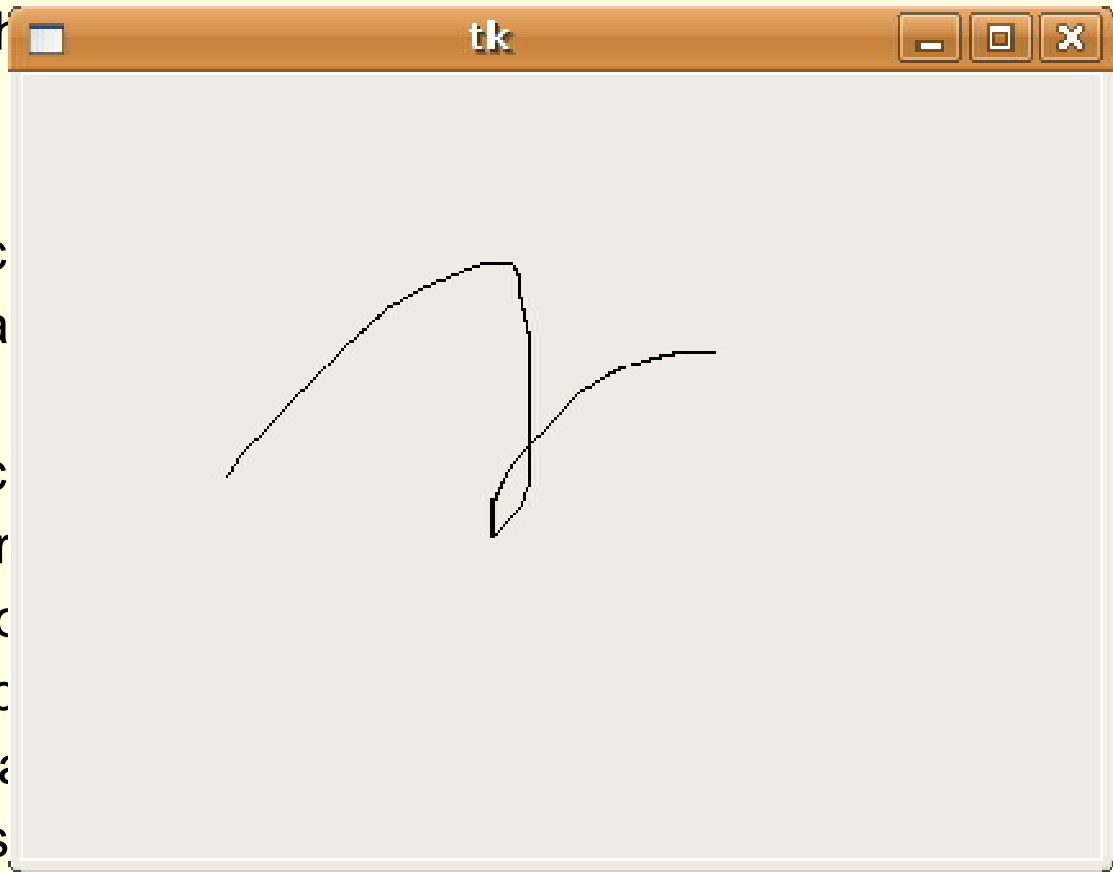
Exemplo

```
from Tkinter import *
master = Tk()
c = Canvas(master, width=512, height=512)
c.pack()
def novalinha(e):
    x,y = c.canvasx(e.x), c.canvasy(e.y)
    c.create_line(x,y,x,y,tags="nova")
def estendelinha(e):
    x,y = c.canvasx(e.x), c.canvasy(e.y)
    coords = c.coords("corrente")
    c.coords("corrente",*coords+(x,y))
def fechalinha(e): c.itemconfigure("nova", fill="red")
c.bind("<Button-1>", novalinha)
c.bind("<B1-Motion>", estendelinha)
c.bind("<ButtonRelease-1>", fechalinha)
c.pack()
```



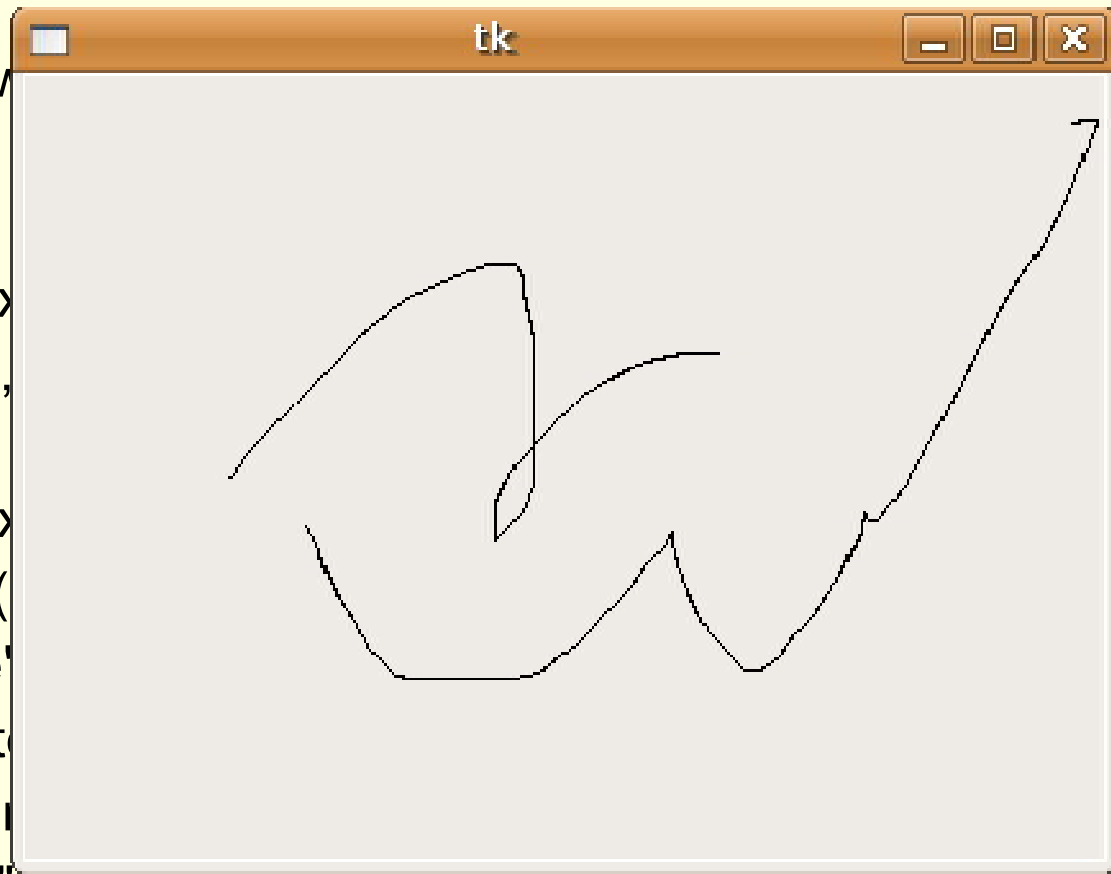
Exemplo

```
from Tkinter import *
master = Tk()
c = Canvas(master, width=400, height=400)
c.pack()
def novalinha(e):
    x,y = c.canvasx(e.x), c.canvasy(e.y)
    c.create_line(x,y,x,y,tags="nova")
def estendelinha(e):
    x,y = c.canvasx(e.x), c.canvasy(e.y)
    coords = c.coords("corrente")
    c.coords("corrente",*coords+(x,y))
def fechalinha(e): c.itemconfigure("corrente", fill="black")
c.bind("<Button-1>", novalinha)
c.bind("<B1-Motion>", estendelinha)
c.bind("<ButtonRelease-1>", fechalinha)
c.pack()
```



Exemplo

```
from Tkinter import *
master = Tk()
c = Canvas(master, width=400, height=400)
c.pack()
def novalinha(e):
    x,y = c.canvasx(e.x), c.canvasy(e.y)
    c.create_line(x,y,x,y, fill='black')
def estendelinha(e):
    x,y = c.canvasx(e.x), c.canvasy(e.y)
    coords = c.coords("corrente")
    c.coords("corrente", coords[0], coords[1], x, y)
def fechalinha(e): c.itemconfigure("corrente", fill='red')
c.bind("<Button-1>", novalinha)
c.bind("<B1-Motion>", estendelinha)
c.bind("<ButtonRelease-1>", fechalinha)
c.pack()
```



Exemplo

```
def selescionalinha(e):
    global x0,y0
    x0,y0 = c.canvasx(e.x), c.canvasy(e.y)
    c.itemconfig(CURRENT, tags="sel")
def movelinha (e):
    global x0,y0
    x1,y1 = c.canvasx(e.x), c.canvasy(e.y)
    c.move("sel",x1-x0,y1-y0)
    x0,y0=x1,y1
def deselescionalinha(e): c.itemconfig("sel", tags=())
c.bind("<Button-3>", selescionalinha)
c.bind("<B3-Motion>", movelinha)
c.bind("<ButtonRelease-3>", deselescionalinha)
c.pack()
mainloop()
```

Exemplo

```
def selescionalinha(e):
```

```
    global x0,y0
```

```
    x0,y0 = c.canvasx(e.x),
```

```
    c.itemconfig(CURRENT
```

```
def movelinha (e):
```

```
    global x0,y0
```

```
    x1,y1 = c.canvasx(e.x),
```

```
    c.move("sel",x1-x0,y1-y
```

```
    x0,y0=x1,y1
```

```
def deselescionalinha(e): c.
```

```
    c.bind("<Button-3>", selec
```

```
    c.bind("<B3-Motion>", mov
```

```
    c.bind("<ButtonRelease-3>", deselescionalinha)
```

```
    c.pack()
```

```
    mainloop()
```



Exemplo

```
def selescionalinha(e):
```

```
    global x0,y0
```

```
    x0,y0 = c.canvasx(e)
```

```
    c.itemconfig(CURR
```

```
def movelinha (e):
```

```
    global x0,y0
```

```
    x1,y1 = c.canvasx(e)
```

```
    c.move("sel",x1-x0,
```

```
    x0,y0=x1,y1
```

```
def deselescionalinha(e)
```

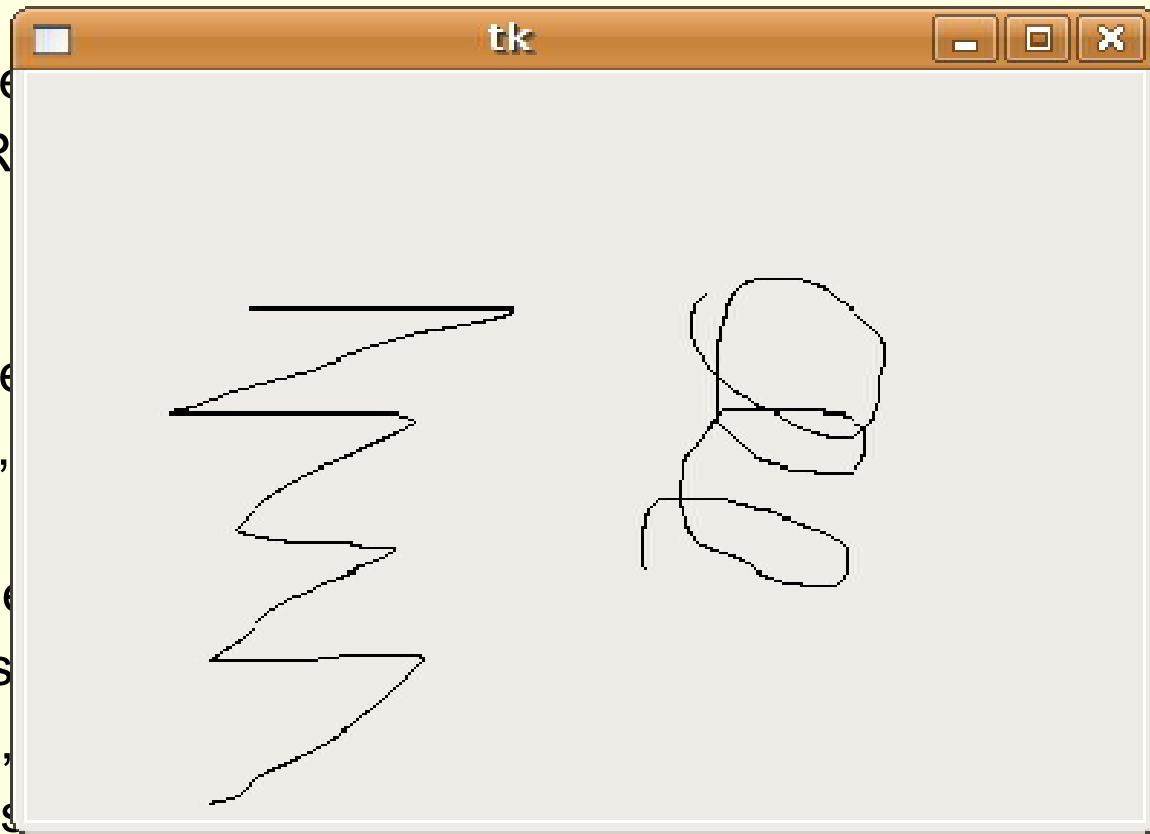
```
    c.bind("<Button-3>", s
```

```
    c.bind("<B3-Motion>",
```

```
    c.bind("<ButtonReleas
```

```
    c.pack()
```

```
    mainloop()
```



Scrollbar

- Barras de rolamento são usadas com outros widgets com área útil maior do que pode ser exibida na janela (Canvas, Text, Listbox, Entry)
- Uma barra de rolamento horizontal (vertical) funciona chamando o método xview (yview) do widget associado
 - Isto é feito configurando a opção command da barra
- Por outro lado, sempre que a visão do widget muda, a barra de rolamento precisa ser atualizada
 - Isto é feito configurando a opção xscrollcommand (ou yscrollcommand) do widget ao método set da barra

Exemplo

```
from Tkinter import *  
lb = Listbox()  
lb.pack(side=LEFT,expand=True,fill="both")  
sb = Scrollbar()  
sb.pack(side=RIGHT,fill="y")  
sb.configure(command=lb.yview)  
lb.configure(yscrollcommand=sb.set)  
for i in range(100):  
    lb.insert(END,i)
```

Exemplo

```
from Tkinter import *
lb = Listbox()
lb.pack(side=LEFT,expand=True)
sb = Scrollbar()
sb.pack(side=RIGHT,fill="y")
sb.configure(command=lb.yview)
lb.configure(yscrollcommand=sb.set)
for i in range(100):
    lb.insert(END,i)
```

