Entry

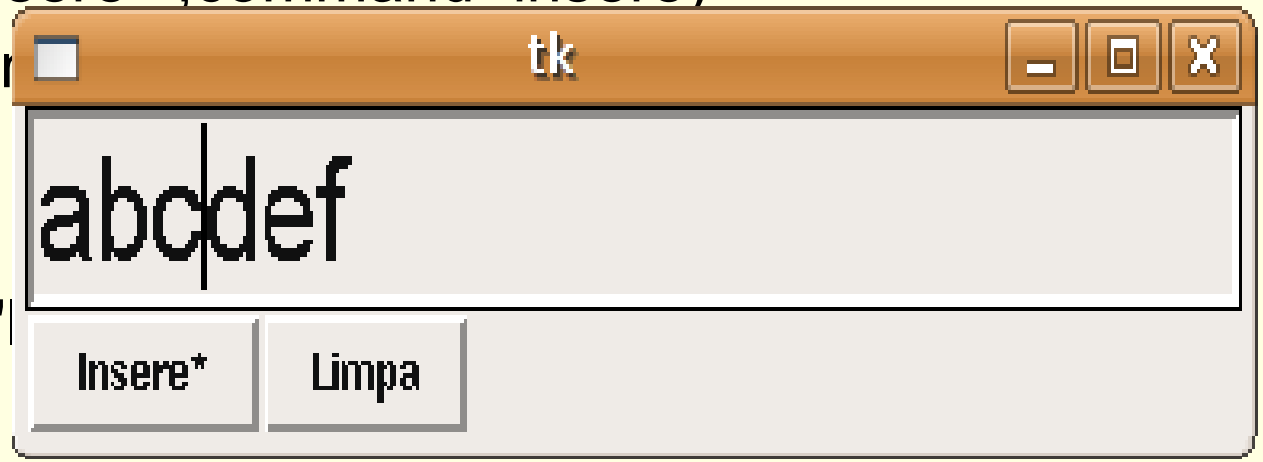
- Um Entry permite entrada/edição de uma linha de texto
- O texto associado ao Entry é normalmente armazenado numa variável indicada pela opção textvariable
  - Se não indicada, é usada uma variável interna cujo valor pode ser obtido usando o método get()
- Há diversos métodos para manipular diretamente 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="Li
e.pack()
for w in (i,l):
    w.pack(side='
mainloop()
```

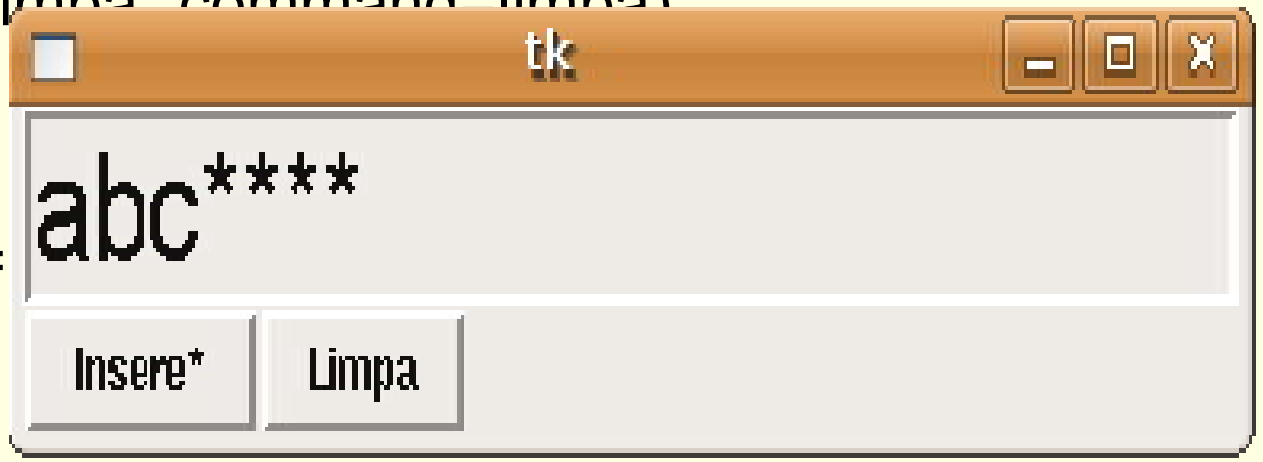


Exemplo

```
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

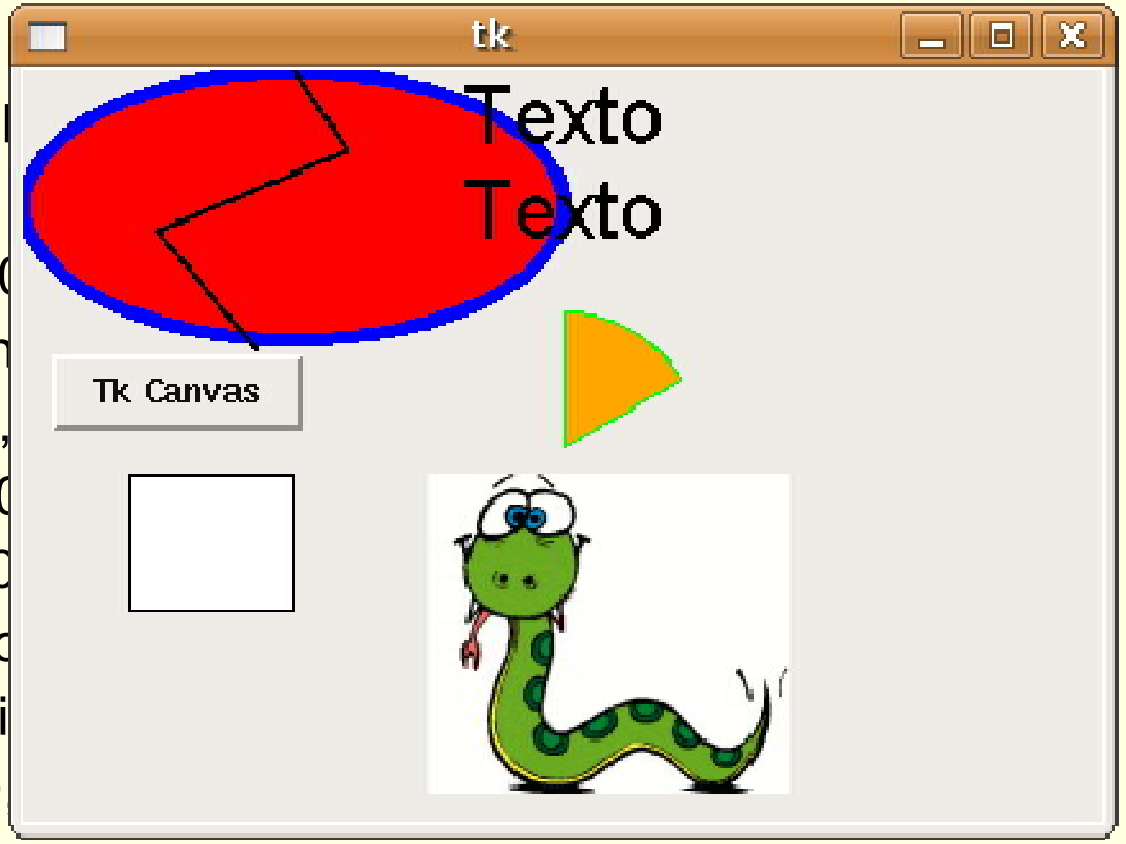
- Permite a exibição e edição de gráficos estruturados 2D
- Elementos gráficos (itens) são introduzidos usando métodos da forma create_*tipo* (...), onde *tipo* pode ser
  - arc arco de círculo
  - bitmap imagem binária
  - image imagem colorida
  - line linha poligonal
  - oval círculos e elipses
  - 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, h
c.pack()
o = c.create_oval(1,1,200,100
widget = Button(text="Tk Can
w = c.create_window(10,120,
l = c.create_line(100,0,120,30
r = c.create_rectangle(40,150
img = PhotoImage(file="pytho
i = c.create_image (150,150,i
a = c.create_arc (150,90,250
            outline="green",fill="orange")
t = c.create_text(200,35,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

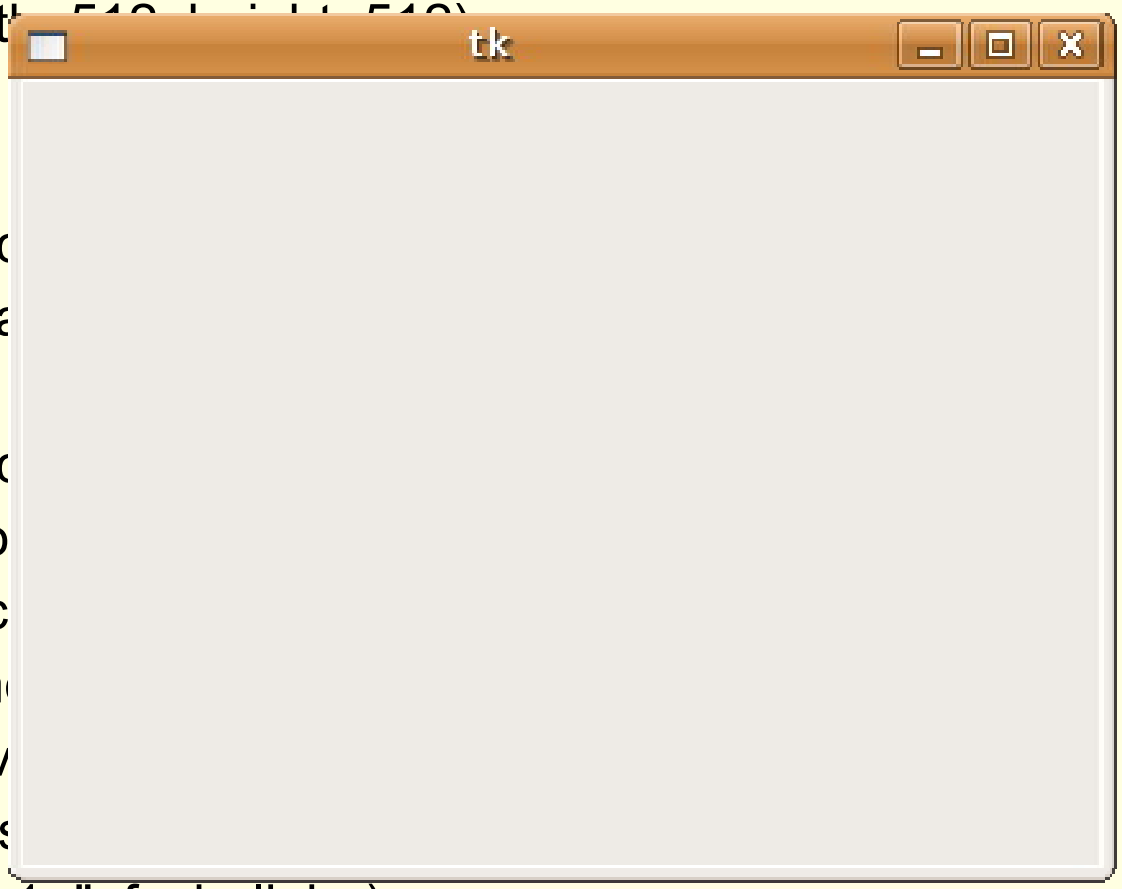
- itemconfig (*itemOuTag*, ...) altera opções do(s) item(s)
- tag_bind(*itemOuTag*, *padrão*, *callback*) associa uma *callback* a um *padrão* de eventos sobre o(s) item(s)
- delete(*itemOuTag*) remove o(s) item(s)
- move(*itemOuTag*, dx,dy) translada o(s) item(s)
- coords(*itemOuTag*, *x1*,*x2*,..*xN*,*yN*) 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()
```

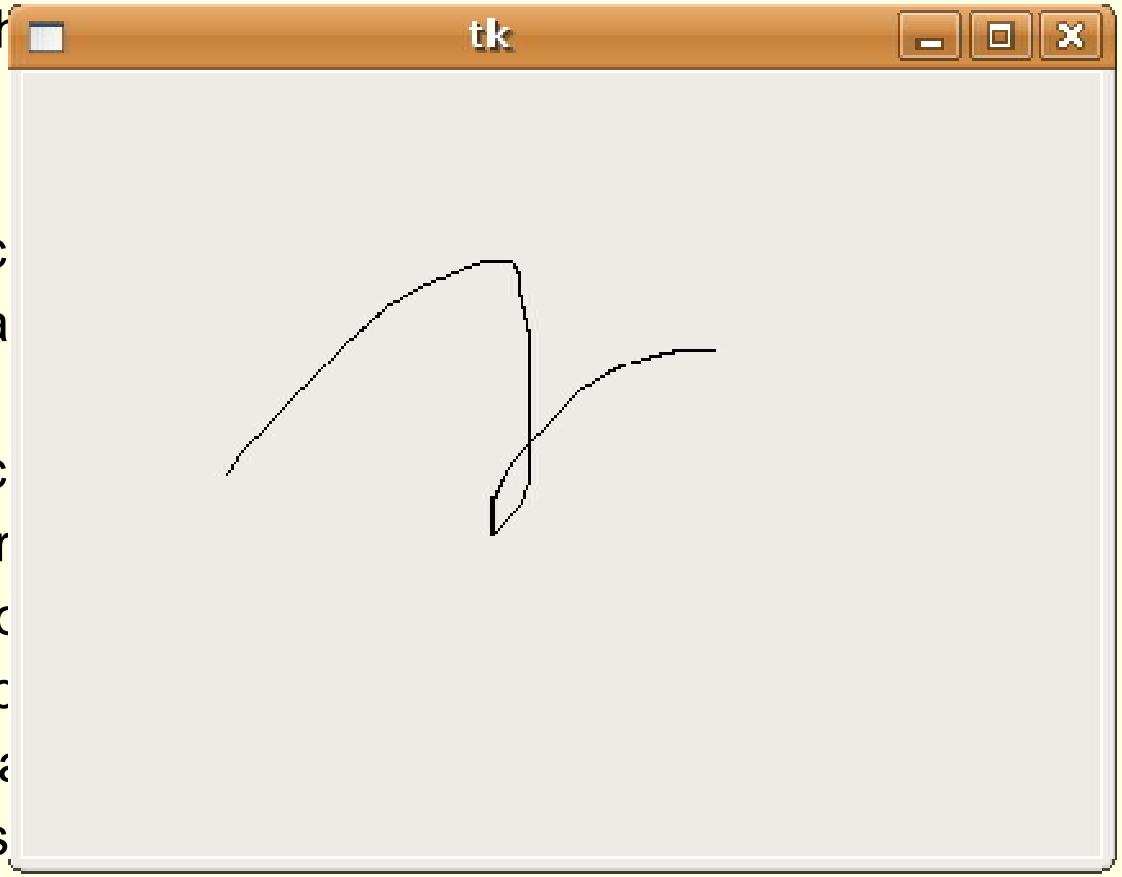
Exemplo

```
from Tkinter import *
master = Tk()
c = Canvas(master, widt
c.pack()
def novalinha(e):
    x,y = c.canvasx(e.x), c
    c.create_line(x,y,x,y,ta
def estendelinha(e):
    x,y = c.canvasx(e.x), c
    coords = c.coords("co
    c.coords("corrente",*c
def fechalinha(e): c.item
c.bind("<Button-1>", nov
c.bind("<B1-Motion>", es
c.bind("<ButtonRelease-1>", fechalinha)
c.pack()
```



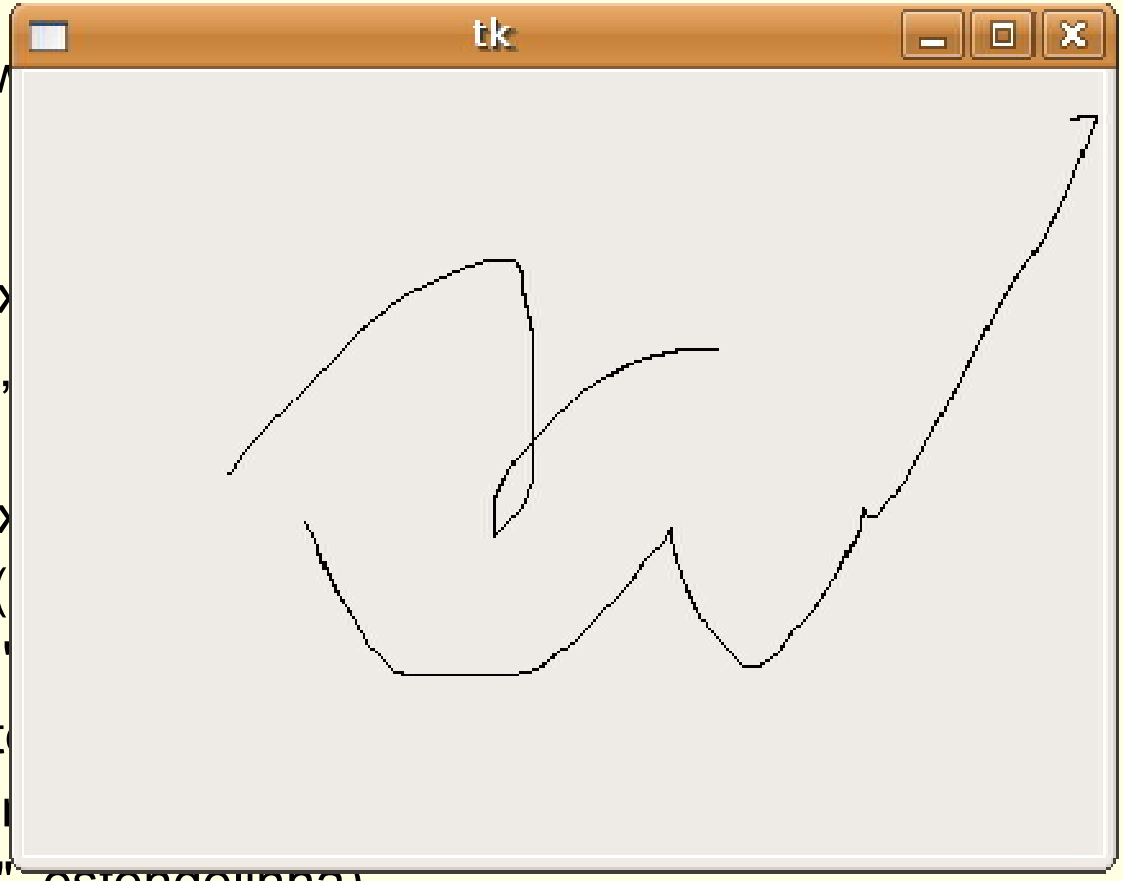
Exemplo

```
from Tkinter import *
master = Tk()
c = Canvas(master, width
c.pack()
def novalinha(e):
    x,y = c.canvasx(e.x), c
    c.create_line(x,y,x,y,ta
def estendelinha(e):
    x,y = c.canvasx(e.x), c
    coords = c.coords("cor
    c.coords("corrente",*cc
def fechalinha(e): c.itemc
c.bind("<Button-1>", nova
c.bind("<B1-Motion>", es
c.bind("<ButtonRelease-1>", fechalinha)
c.pack()
```



Exemplo

```
from Tkinter import *
master = Tk()
c = Canvas(master, w
c.pack()
def novalinha(e):
    x,y = c.canvasx(e.x
    c.create_line(x,y,x,
def estendelinha(e):
    x,y = c.canvasx(e.x
    coords = c.coords(
    c.coords("corrente"
def fechalinha(e): c.it
c.bind("<Button-1>", n
c.bind("<B1-Motion>", estendelinha)
c.bind("<ButtonRelease-1>", fechalinha)
c.pack()
```

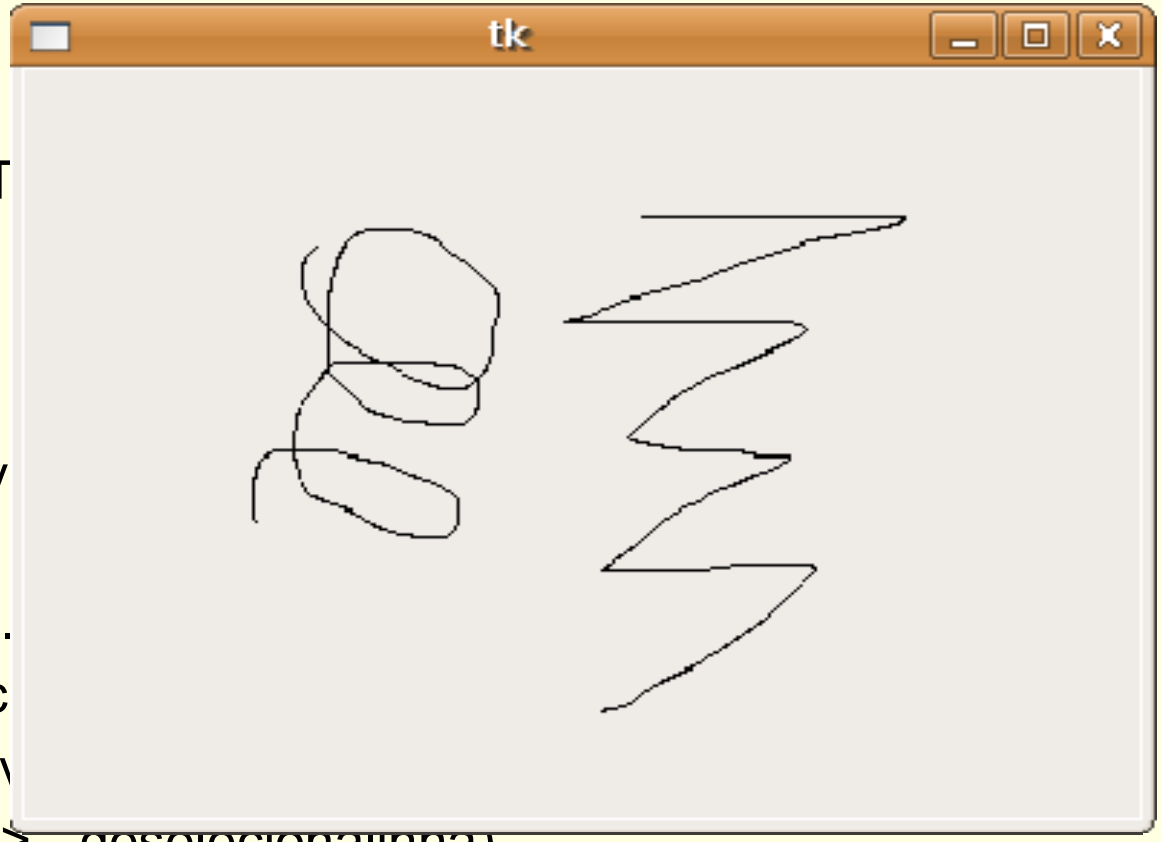


Exemplo

```
def selecionalinha(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 deselecionalinha(e): c.itemconfig("sel", tags=())
c.bind("<Button-3>", selecionalinha)
c.bind("<B3-Motion>", movelinha)
c.bind("<ButtonRelease-3>", deselecionalinha)
c.pack()
mainloop()
```

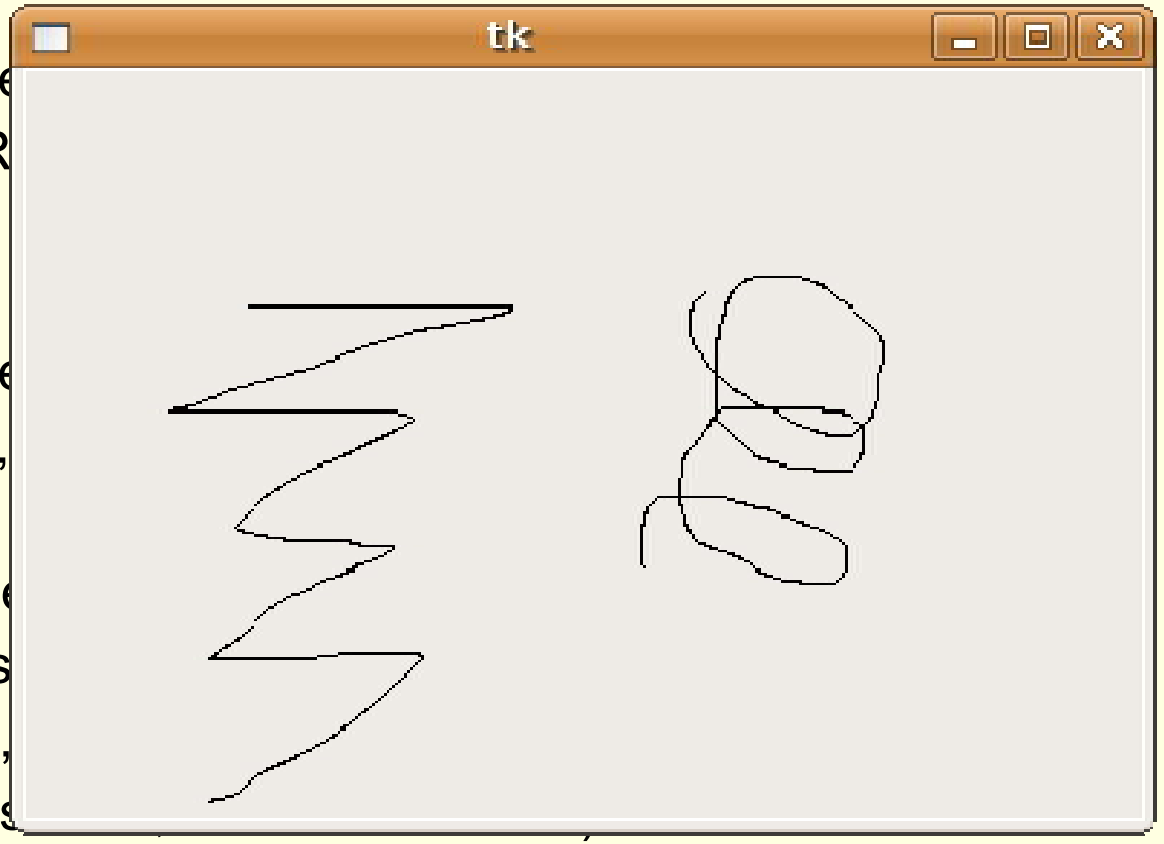
Exemplo

```
def selecionalinha(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 deselecionalinha(e): c.
c.bind("<Button-3>", selec
c.bind("<B3-Motion>", mov
c.bind("<ButtonRelease-3>", deselecionalinha)
c.pack()
mainloop()
```



Exemplo

```
def selecionalinha(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 deselecionalinha(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=Tru
sb = Scrollbar()
sb.pack(side=RIGHT,fill="y”)
sb.configure(command=lb.yvie
lb.configure(yscrollcommand=s
for i in range(100):
    lb.insert(END,i)
```