



Componentes (widgets) & Eventos del Ratón

PYTHON

JAIME ALBERTO GUZMAN LUNA
Universidad Nacional de Colombia

1

1



Componentes: los Menús (1)

- Tkinter ofrece la clase **Menu**, la cual hereda de **Widget** y permite crear menús en la ventana.
- Una ventana puede tener una barra de menú donde se sitúan los menús desplegables.
- Cada menú tiene elementos que representan las opciones o ítems que el usuario puede seleccionar.

2

2

Componentes: los Menús (2)

- Para declarar un menú se hace lo siguiente:

```
import tkinter as tk

def evento():
    pass

ventana = tk.Tk()
ventana.title("Tkinter APP")

menuBar = tk.Menu(ventana)
ventana.config(menu=menuBar)

menu1 = tk.Menu(menuBar)
menuBar.add_cascade(label="Menu 1",
                    menu=menu1, command=evento)

menu1.add_command(label="Item 1", command=evento)
menu1.add_separator()
menu1.add_command(label="Item 2", command=evento)

ventana.mainloop()
```

ejemplo0.py

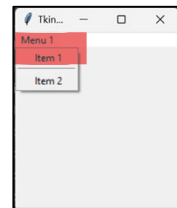
Se crea la barra de menús instanciando un objeto de la clase `Menu`, mandándole la ventana en el constructor y con el método `config()` de la clase `Tk`, se asigna la barra de menú a la ventana usando el argumento `menu`.

Se crea el menú desplegable instanciando otro objeto de la clase `Menu`, pero esta vez en el constructor se le manda la barra de menú. Y con el método `add_cascade()` se agrega a la barra.

Se agregan los ítems al menú con método `add_command()`. **Nota:** el parámetro `command` recibe una función que se ejecuta al seleccionar el ítem.

En medio de los ítems se incluye un separador con el método `add_separator()`

Al ejecutar el código sale un separador de una línea punteada en la parte superior de "Item 1"



3

3

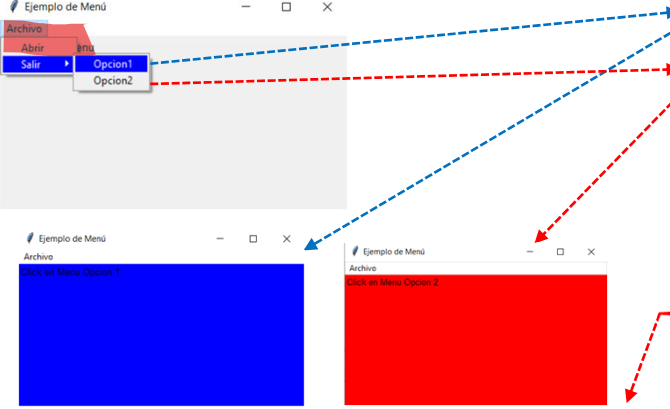
Componentes: los Menús (3)

- Otros parámetros de la clase `Menu`
 - **bg:** color de los ítems.
 - **fg:** color de texto de los ítems.
 - **activebackground:** color de fondo cuando el cursor del mouse se posiciona en un ítem.
 - **activeforeground:** color de texto cuando el cursor del mouse se posiciona en un ítem.
 - **font:** especifica la fuente de texto de los ítems.

4

Componentes: los Menús (4)

También sale el separador



```
import tkinter as tk

def opcion1():
    ventana.config(bg="blue")
    texto.config(text="Click en Menu Opcion 1",bg="blue")

def opcion2():
    ventana.config(bg="red")
    texto.config(text="Click en Menu Opcion 2",bg="red")

ventana = tk.Tk()
ventana.title("Ejemplo de Menú")
ventana.geometry("400x200")

menuBar = tk.Menu(ventana)
ventana.config(menu=menuBar)

menu1 = tk.Menu(menuBar,activebackground="blue",activeforeground="white")
menuBar.add_cascade(label="Archivo", menu=menu1)
menu1.add_command(label="Abrir")

menu2 = tk.Menu(menuBar,activebackground="blue",activeforeground="white")
menu2.add_command(label="Opcion1",command=opcion1)
menu2.add_command(label="Opcion2",command=opcion2)
menu1.add_cascade(label="Salir", menu=menu2)

texto = tk.Label(ventana,text="Presione un Menu",font=("Arial",10))
texto.pack(anchor="w")

ventana.mainloop()
```

Un ítem puede desplegar otro menú. En este ejemplo se instancia un objeto llamado menu2 y se le agregan unos ítems. Y luego menu2 se agrega al menú Archivo como una opción llamada salir.

ejemplo1.py

5

Componentes: Label (1)

ttk.Label es una clase en ttk que nos permite crear etiquetas con elementos de texto

- Label es un clase de tkinter que nos permite crear etiquetas con elementos de texto.
- Su estructura es la siguiente:

```
miLabel = tk.Label(master,text,...)      miLabel = ttk.Label(master, text, ...)
```

- El parámetro *master* recibe el nombre del contenedor del Label, el cual puede ser la ventana o un frame y el parámetro *text* recibe la cadena de texto que tendrá el objeto.

- Los tres puntos denotan parámetros opcionales tales como: *bg* (color del fondo del label), *fg* (color del texto del label), *font* (una tupla con la fuente y el tamaño de letra). Más atributos se pueden revisar en:

https://www.tutorialspoint.com/python/tk_label.htm

<https://anzelg.github.io/rin2/book2/2405/docs/tkinter/ttk-Label.html>

Con la clase **PhotoImage** se puede asociar una imagen a una etiqueta con el atributo *image* de **Label**. **ttk.Label**

Especificar que estos parámetros se editan mediante Style, y lo cambiado aquí se envía como parámetro al Label

6

6

Componentes: Label (2)

```
import tkinter as tk

ventana = tk.Tk()
ventana.title("Ejemplo de Etiqueta")
ventana.geometry("400x200")

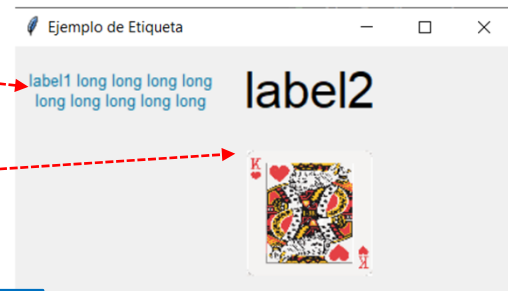
label1 = tk.Label(ventana, text="label1 long long long
long\nlong long long long long",
font=("Arial", 10), fg="#0076a3")
label1.grid(row=0, column=0, padx=10, pady=10, sticky="n")

label2 =
tk.Label(ventana, text="label2", font=("Arial", 30), fg="black")
imagen = tk.PhotoImage(file="imagenes/carta.png")
label2.config(image=imagen, compound="bottom")
label2.grid(row=0, column=1, padx=10, pady=10, sticky="n")

ventana.mainloop()
```

ejemplo2.py

Se ha especificado un valor `compound="bottom"`, para lo cual la imagen se mostrará debajo del texto. Si no se usara este atributo, la imagen taparía el texto. Los valores de `compound` pueden ser: LEFT, RIGHT, BOTTOM, or TOP



Alternativa: `label2 = tk.Label(ventana, text="label2", font=("Arial", 30), fg="black", image=imagen, compound="bottom")`

7

7 `label2 = ttk.Label(ventana, text="label2", font=("Arial", 30), foreground="black", compound="bottom")`

Componentes: Button (1)

Button es una clase de ttk que permite crear botones en una ventana.

- **Button es un clase de tkinter que nos permite crear botones en una ventana.**
 - Su estructura es la siguiente:
`miBoton = tk.Button(master,...)` `miBoton = ttk.Button(master, ...)`
 - El parámetro *master* recibe el nombre del contenedor del botón, el cual puede ser la ventana o un frame y el parámetro *text* recibe la cadena de texto que tendrá el objeto.
 - Los tres puntos denotan parámetros opcionales tales como: *bg* (color del fondo del botón), *fg* (color del texto del botón), *font* (una tupla con la fuente y el tamaño de letra del botón), *height* (el alto en píxeles), *width* (el ancho en píxeles). Más atributos se pueden revisar en:
https://www.tutorialspoint.com/python/tk_button.htm <https://anzelg.github.io/rin2/book2/2405/docs/tkinter/ttk-Button.html>
 - Con la clase `PhotoImage` se puede asociar una imagen a un botón con el atributo *image* de `Button`: `tk.Button`
- Nota:** esto hace que se reemplace el texto por la imagen. Para que ambos estén se debe agregar el parámetro *compound*, el cuál pone la imagen en una posición específica (top, bottom, left, right) en relación al texto.
- Con el parámetro *command* se puede asociar un evento al momento de dar click al botón.

8

8

Los tres puntos denotan parámetros opcionales tales como *style* (permite aplicar un estilo personalizado al botón), *image* (se utiliza para asociar una imagen al botón), *width* (ancho del área de texto del botón)

Componentes: Button (2)

```
import tkinter as tk

def botonTexto():
    print("Click en botón BotonTexto")

def botonIcono():
    print("Click en botón Icono")

def botonTextoIcono():
    print("Click en botón TextoIcono")

ventana = tk.Tk()
ventana.title("Ejemplo de Botones")
ventana.geometry("500x200")

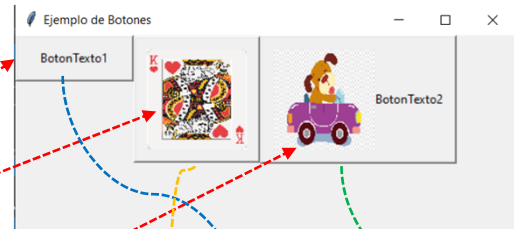
boton1 = tk.Button(ventana, text="BotonTexto1", command=botonTexto)
boton1.grid(row=0, column=0, padx=20, pady=10, sticky="n")

imagen1 = tk.PhotoImage(file="imagenes\carta.png")
boton2 = tk.Button(ventana, command=botonIcono, image=imagen1)
boton2.grid(row=0, column=1, padx=10, pady=10)

boton3 =
tk.Button(ventana, text="BotonTexto2", compound="left", command=botonTextoIcono)
imagen2 = tk.PhotoImage(file="imagenes\perrito.png")
boton3.config(image=imagen2)
boton3.grid(row=0, column=2, padx=10, pady=10)

ventana.mainloop()
```

ejemplo3.py



Click en botón BotonTexto
Click en botón Icono
Click en botón TextoIcono

9

9

Componentes: Entry (1)

- **Entry** es un clase de **ttk** **tkinter** que nos permite crear un espacio para capturar entrada por teclado en una ventana o simplemente mostrar texto
- Su estructura es la siguiente:


```
miEntry = tk.Entry(master, textvariable, ...) miEntry = ttk.Entry(master, textvariable, ...)
```
- El parámetro **master** recibe el nombre del contenedor de la entrada, el cual puede ser la ventana o un frame. Los tres puntos denotan parámetros opcionales tales como: **bg** (color del fondo de la entrada), **fg** (color del texto que se muestra o se ingresa), **state** (determina si se puede ingresar texto o no. Si es 'disabled' no lo permite).
- El parámetro **textvariable** permite definir un texto por defecto en la entrada. Este parámetro recibe un objeto de la clase **StringVar**, el cual a su vez recibe un contenedor (el mismo del objeto **Entry**) y un texto.
- **Entry** define algunos métodos tales como:
 - **get()**: Retorna el texto que se haya ingresado.
 - **delete (first, last=None)**: Borra los caracteres desde una posición 'first' hasta una posición 'last' (si se omite este último solo se borra el que está en 'first'). También, 'last' puede recibir un string "end" que indica que el borrado es hasta el último carácter.
 - **insert(index, s)**: Inserta un string 's' antes del carácter ubicado en el 'index' dado.
- Más métodos y atributos se pueden revisar en: https://www.tutorialspoint.com/python/tk_entry.htm

<https://anzelgj.github.io/rin2/book2/2405/docs/tkinter/ttk-Entry.html>

10

Especificar
que se
debe usar
Style

10

Componentes: Entry (2)

```
import tkinter as tk

def eliminar():
    entrada2.delete(0, last=tk.END)

ventana = tk.Tk()
ventana.title("Ejemplo de Entry")
ventana.geometry("400x100")

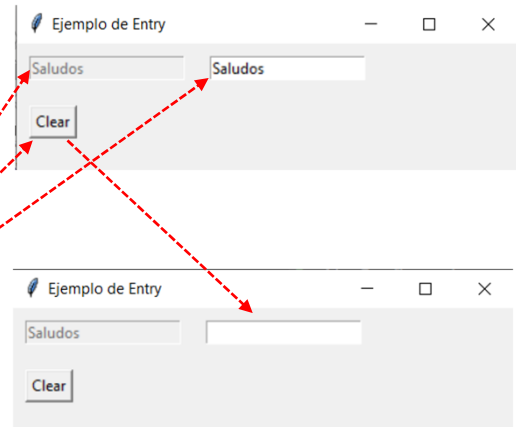
entrada1 =
tk.Entry(ventana, state="disabled", textvariable=tk.StringVar(ventana, value="Saludos"))
entrada1.grid(row=0, column=0, padx=10, pady=10, sticky="n")

entrada2 =
tk.Entry(ventana, textvariable=tk.StringVar(ventana, value="Saludos"))
entrada2.grid(row=0, column=1, padx=10, pady=10, sticky="n")

boton = tk.Button(ventana, text="Clear", command=eliminar)
boton.grid(row=1, column=0, padx=10, pady=10, sticky="w")

ventana.mainloop()
```

ejemplo4.py



11

11

Componentes: Combobox (1)

- **Combobox** es un clase de del módulo **ttk** que está en tkinter que permite insertar un listado de elementos para escoger uno de ellos.
- Su estructura es la siguiente:


```
from tkinter import ttk
miCombobox = ttk.Combobox(master, values, textvariable,...)
```
- Se debe importar un módulo interno de Tkinter llamado **ttk**.
- El parámetro **master** recibe el nombre del contenedor del listado, el cual puede ser la ventana o un frame.
- El parámetro **values** recibe una lista de strings que denotan las opciones.
- El parámetro **textVariable** permite definir un texto por defecto en la lista. Este parámetro recibe un objeto de la clase **StringVar**, el cual a su vez recibe un contenedor (el mismo del objeto Combobox) y un texto.
- Existe un **evento** especial para Combobox, el cuál se ejecuta cuando se selecciona una opción de un listado. Se denota de esta manera: **<<ComboboxSelected>>**.

12

12

Componentes: Combobox (2)

```
import tkinter as tk
from tkinter import ttk

ventana = tk.Tk()
ventana.title("Ejemplo de Combobox")
ventana.geometry("400x200")

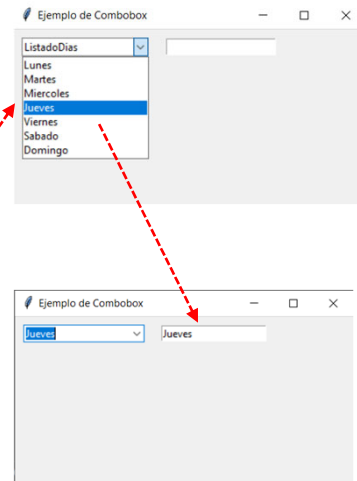
def changed(event):
    entrada.delete(0, "end")
    entrada.insert(0, combo.get())

valorDefecto = tk.StringVar(value='ListadoDias')
combo = ttk.Combobox(ventana, values=["Lunes", "Martes", "Miercoles",
    "Jueves", "Viernes", "Sabado", "Domingo"], textvariable=valorDefecto)
combo.bind("<<ComboboxSelected>>", changed)
combo.grid(row=0, column=0, padx=10, pady=10, sticky="w")

entrada = tk.Entry(ventana)
entrada.grid(row=0, column=1, padx=10, pady=10, sticky="w")

ventana.mainloop()
```

ejemplo5.py



13

13

Componentes: Diálogos y Alertas (1)

- Para los diálogos y alertas, tkinter implementa directamente el módulo `messagebox`, el cual internamente implementa métodos que pueden mostrar información de diálogos y alertas en forma de ventana.
- La estructura es la siguiente:
`messagebox.nombreFuncion(titulo, mensaje, ...)`
- En `nombreFuncion` se pone el nombre de una de las diferentes funciones que se implementa:
 - showinfo**: muestra información con un **signo de admiración**. La ventana que aparece solo tiene un botón de Aceptar.
 - showwarning**: muestra información con un **signo de alerta**. La ventana que aparece solo tiene un botón de Aceptar.
 - showerror**: muestra información respecto a un **error**. La ventana que aparece solo tiene un botón de Aceptar.
 - askokcancel**: muestra **información** en la que se debe **Aceptar** o **Cancelar**.
 - askyesno**: muestra una **pregunta** donde se debe oprimir **Sí** o **No**.
 - askretrycancel**: muestra **información** en la que se debe **Devolver** o **Cancelar**.
- Cuando se oprime el botón de **cancelar**, el método retorna **False**; de lo contrario retorna **True**. Estos retornos se pueden asignar a una variable y dependiendo de lo que devuelva se puede ejecutar un evento u otro `messagebox`.

14

14

Componentes: Diálogos y Alertas (2)

```
import tkinter as tk
from tkinter import messagebox

ventana = tk.Tk()
ventana.title("Creación alertas")
ventana.geometry("400x200")

def confirmacion():
    opcion = messagebox.askokcancel("Dialogo de confirmación","Ventana de dialogo de confirmación")
    if opcion:
        messagebox.showinfo("Dialogo de confirmación","Has pulsado Aceptar")
    else:
        messagebox.showerror("Error","Ventana de dialogo de error")

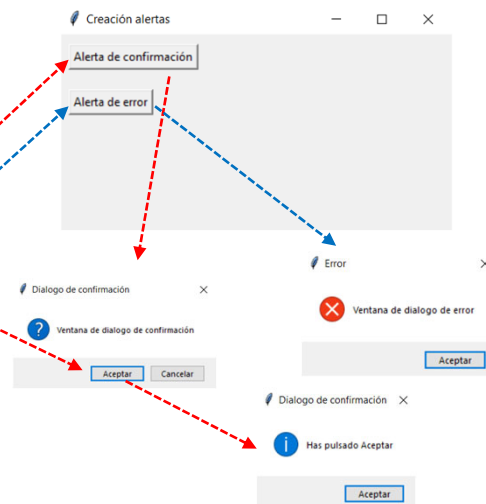
def error():
    messagebox.showerror("Error","Ventana de dialogo de error")

boton = tk.Button(ventana,text="Alerta de confirmación",command=confirmacion)
boton.pack(anchor="w",padx=10,pady=10)

boton2 = tk.Button(ventana,text="Alerta de error",command=error)
boton2.pack(anchor="w",padx=10,pady=10)

ventana.mainloop()
```

ejemplo6.py



15

15

Componentes: Text (1)

- **Text** es un clase de tkinter que nos permite crear entradas de texto multilínea.
- Su estructura es la siguiente:


```
miText = tk.Text(master,...)
```
- El parámetro *master* recibe el nombre del contenedor de la entrada, el cual puede ser la ventana o un frame.
- Los tres puntos denotan parámetros opcionales tales como: *bg* (color del fondo de la entrada), *fg* (color del texto que se muestra o se ingresa), *state* (determina si se puede ingresar texto o no. Si es 'disabled' no lo permite).
- Text define algunos métodos tales como:
 - **get(first,last)**: Retorna el texto que se haya ingresado desde cierto rango de líneas dado en decimales, los índices empiezan en 1.0 (ejemplo: si first=1.0 y last=3.0 retornará la primera y la segunda línea (no inclusivo)). Si en last se pone 'end' irá hasta la última línea (inclusive).
 - **delete (first, last)**: Borra los caracteres dentro de un rango de líneas especificado (aplica las mismas reglas que con el get).
 - **insert(index, s)**: Inserta un string 's' en una línea dada en decimal.

16

16

Componentes: Text (2)

```
import tkinter as tk

ventana = tk.Tk()
ventana.title("Ejemplo Text")

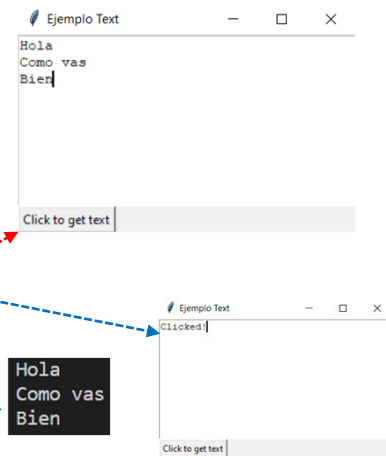
def obtenerText():
    print(texto.get(1.0, 'end'))
    texto.delete(1.0, 'end')
    texto.insert(1.0, "Clicked!")

texto = tk.Text(ventana,width=40,height=10)
texto.pack()

boton = tk.Button(ventana,text="Click to get
text",command=obtenerText)
boton.pack(side="bottom",anchor="w")

ventana.mainloop()
```

ejemplo7.py



17

17

Eventos del ratón (1)

- Recordando los eventos de ratón en Tkinter que se pueden usar con el método bind():

Evento	Acción del usuario	Tipo	Detalles
<Button>	Dar click a un widget.	Mouse	En la etiqueta definir -1, -2, -3 para click derecho, mitad o izquierdo respectivamente. Ejm: <Button-1>
<Motion>	Mover el mouse mientras da click a un widget.	Mouse	En la etiqueta definir inicialmente B1, B2 y B3 para click derecho, mitad o izquierdo. Ejm: <B1-Motion>
<ButtonRelease>	Se deja de dar click a un widget.	Mouse	En la etiqueta definir -1, -2, -3 para click derecho, mitad o izquierdo respectivamente. Ejm: <ButtonRelease-1>
<Enter>	El puntero del mouse entra en el widget (no click)	Mouse	
<Leaves>	El puntero del mouse sale del widget.	Mouse	

18

18

Eventos del ratón (2)

- Los objetos tipo **Event** que se administran en el método bind cuentan con los siguientes atributos:
 - type**: entero que señala el tipo del evento. Aquí se puede ver a cuál tipo pertenece cada entero: <https://anzeljq.github.io/rin2/book2/2405/docs/tkinter/event-types.html>
 - x**: coordenada en X del cursor del ratón relativa al widget.
 - y**: coordenada en Y del cursor del ratón relativa al widget.
 - x_root**: coordenada en X del cursor del ratón relativa a la ventana.
 - y_root**: coordenada en Y del cursor del ratón relativa a la ventana.
 - num**: tipo de click del ratón (1 si es izquierdo, 2 si es el del medio y 3 si es el derecho).
 - widget**: objeto tipo widget donde se ejecutó el evento.

19

19

Eventos del ratón (3)

```
import tkinter as tk

ventana = tk.Tk()
ventana.title("Mouse Events")
ventana.geometry("400x200")

def informacionEvento(event):
    print("Tipo de evento:", event.type, "\n", "X : Y - ", event.x, ":", event.y, "\n",
          "XVentana : YVentana - ", event.x_root, ":", event.y_root, "\n", "num:", event.num,
          "\nWidget:", event.widget, "\n")

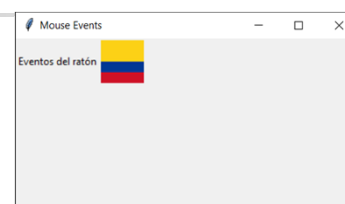
etiqueta = tk.Label(ventana, text="Eventos del ratón")
etiqueta.grid(row=0, column=0)

bandera = tk.Label(ventana)
imagen = tk.PhotoImage(file="imagenes/Colombia.png")
bandera.config(image=imagen)
bandera.bind("<Button-1>", informacionEvento)
bandera.bind("<Enter>", informacionEvento)
bandera.bind("<Leave>", informacionEvento)
bandera.bind("<B1-Motion>", informacionEvento)
bandera.bind("<ButtonRelease-1>", informacionEvento)

bandera.grid(row=0, column=1)

ventana.mainloop()
```

ejemplo8.py



```
Tipo de evento: 4
X : Y - 31 : 26
XVentana : YVentana - 188 : 109
num: 1
Widget: .Label2

Tipo de evento: 7
X : Y - 53 : 47
XVentana : YVentana - 210 : 130
num: ??
Widget: .Label2

Tipo de evento: 8
X : Y - 55 : 59
XVentana : YVentana - 212 : 142
num: ??
Widget: .Label2

Tipo de evento: 6
X : Y - 26 : 28
XVentana : YVentana - 183 : 111
num: ??
Widget: .Label2

Tipo de evento: 5
X : Y - 27 : 28
XVentana : YVentana - 184 : 111
num: 1
Widget: .Label2
```

20

20



Lecturas

- Atributos de la clase Event:

<https://anzelg.github.io/rin2/book2/2405/docs/tkinter/event-types.html>