

Crear una ventana

Creando una ventana

```
1  # Importar librería
2  import tkinter as tk
3
4  # Inicializar ventana
5  ventana = tk.Tk()
6  ventana.title("Python - Tkinter")
7
8  # Activar ventana
9  ventana.mainloop()
```



Evitar modificar el tamaño
de una ventana

Evitar modificar el tamaño

- **resizable (se agrega antes del mainloop):**

```
# Para evitar modificar el tamaño de una ventana  
ventana.resizable(0, 0)
```

Agregar etiquetas (textos)

Agregar etiquetas

- Label (se requiere importar ttk)

```
1  # Importar librería
2  import tkinter as tk
3  from tkinter import ttk
4
5  # Inicializar ventana
6  ventana = tk.Tk()
7  ventana.title("Python - Tkinter")
8
9  # Agregar etiqueta
10 ttk.Label(ventana, text="Hola Crayola!!!").grid(column=0, row=0)
11
12 # Activar ventana
13 ventana.mainloop()
```



Agregar etiquetas (método 2)

- **Label (se requiere importar ttk)**

```
1  # Importar librería
2  import tkinter as tk
3  from tkinter import ttk
4
5  # Inicializar ventana
6  ventana = tk.Tk()
7  ventana.title("Python - Tkinter")
8
9  # Agregar etiqueta por medio de un objeto
10 etiqueta = ttk.Label(ventana, text="Hola Crayola!!!")
11 etiqueta.grid(column=0, row=0)
12
13 # Activar ventana
14 ventana.mainloop()
```



Agregar botones

Agregar botones

- Se agrega el botón:

```
# Agregar un botón
accion = ttk.Button(ventana, text="Haz Click Aquí!", command=funcion_click)
accion.grid(column=1, row=0)
```

- Se le asigna una función para el manejo de su evento:

```
def funcion_click():
    accion.configure(text="** Haz hecho Click! **")
    etiqueta.configure(foreground='red')
```

- **Nota:** Las funciones deben agregarse antes del código principal que las manda llamar.



Agregar botones

- Código completo:

```
1  # Importar librería
2  import tkinter as tk
3  from tkinter import ttk
4
5  def funcion_click():
6      accion.configure(text="** Haz hecho Click! **")
7      etiqueta.configure(foreground='red')
8
9  # Inicializar ventana
10 ventana = tk.Tk()
11 ventana.title("Python - Tkinter")
12
13 # Agregar etiqueta por medio de un objeto
14 etiqueta = ttk.Label(ventana, text="Hola Crayola!!!")
15 etiqueta.grid(column=0, row=0)
16
17 # Agregar un botón
18 accion = ttk.Button(ventana, text="Haz Click Aquí!", command=funcion_click)
19 accion.grid(column=1, row=0)
20
21 # Activar ventana
22 ventana.mainloop()
```



Agregar botones

- **Ventana antes y después de generado el evento (click):**



Cajas de texto

Agregar una caja de texto

- **Entry:**

```
# Agregar una caja de texto
nombre = tk.StringVar()
preguntar_nombre = ttk.Entry(ventana, width=20, textvariable=nombre)
preguntar_nombre.grid(column=0, row=1)
```

- **Modificamos la función del evento para que imprima el nombre que se acaba de preguntar:**

```
def funcion_click():
    accion.configure(text='Hola ' + nombre.get())
```



Agregar caja de texto

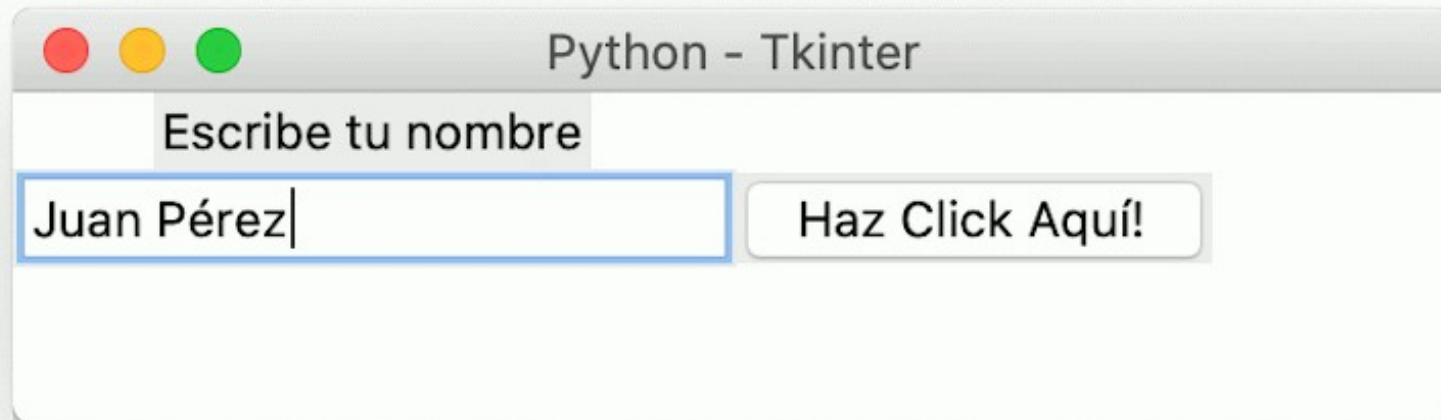
- Código completo:

```
1  # Importar librería
2  import tkinter as tk
3  from tkinter import ttk
4
5  def funcion_click():
6      accion.configure(text='Hola ' + nombre.get())
7
8  # Inicializar ventana
9  ventana = tk.Tk()
10  ventana.title("Python - Tkinter")
11
12  # Agregar etiqueta por medio de un objeto
13  etiqueta = ttk.Label(ventana, text="Escribe tu nombre")
14  etiqueta.grid(column=0, row=0)
15
16  # Agregar una caja de texto
17  nombre = tk.StringVar()
18  preguntar_nombre = ttk.Entry(ventana, width=20, textvariable=nombre)
19  preguntar_nombre.grid(column=0, row=1)
20
21  # Agregar un botón
22  accion = ttk.Button(ventana, text="Haz Click Aquí!", command=funcion_click)
23  accion.grid(column=1, row=1)
24
25  # Activar ventana
26  ventana.mainloop()
```



Agregar caja de texto

- **Ventana antes y después de generado el evento (click):**



Objeto activo

Objeto activo

- **focus** (se agrega antes del mainloop y nos define cual de los objetos de la interfaz gráfica estará activo cuando se arranque la interfaz:

```
# Definir objeto activo  
preguntar_nombre.focus()
```

Deshabilitar eventos

Deshabilitar eventos

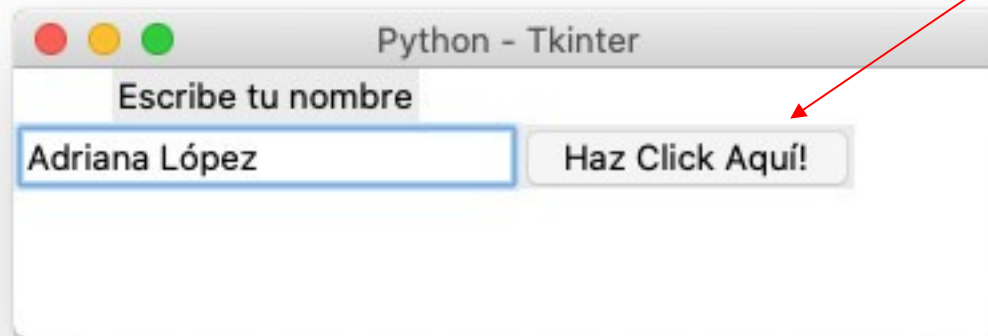
- **configure:**

```
# Deshabilitar eventos  
accion.configure(state='disabled')
```

Deshabilitar eventos

- **configure:**

No se puede
hacer click en el
evento



Listas desplegables

Listas desplegables

■ Combobox:

```
# Agregar lista desplegable
numero = tk.StringVar()
seleccionar_numero = ttk.Combobox(ventana, width=12, textvariable=numero)

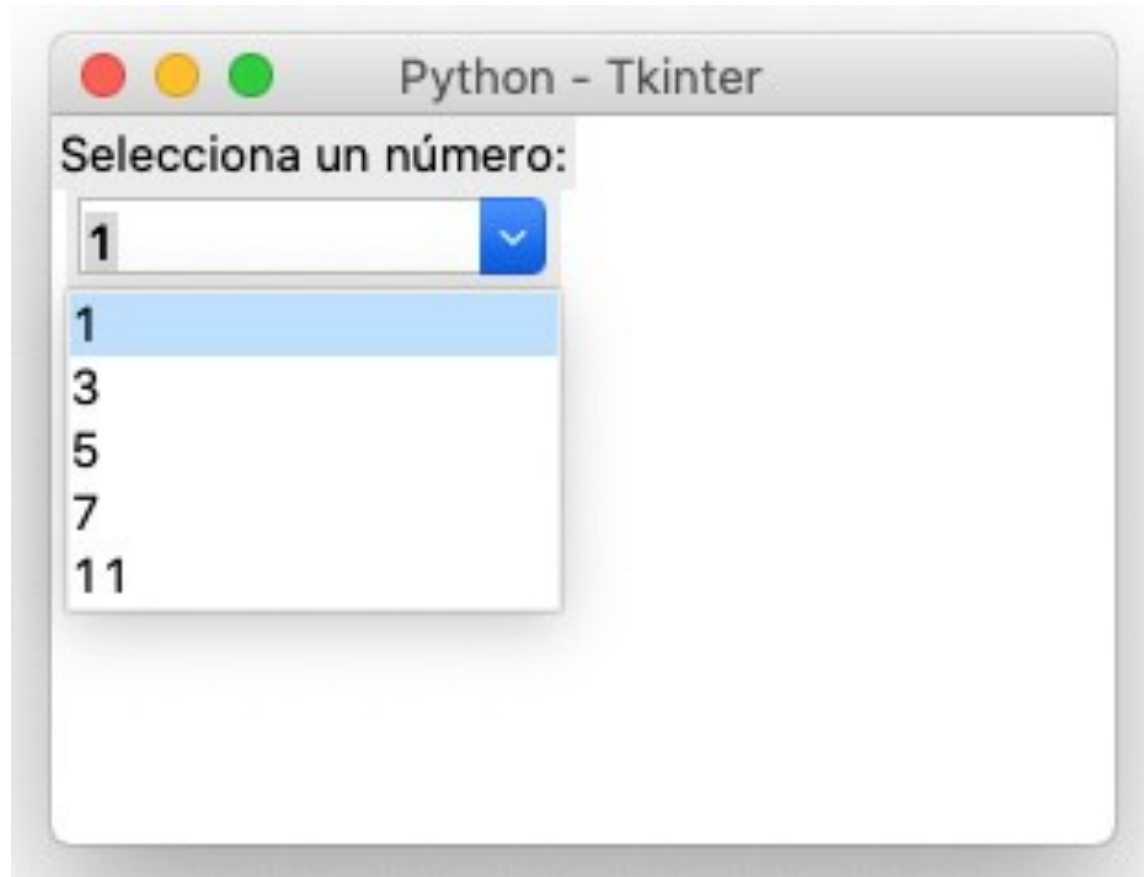
# Llenar la lista desplegable
seleccionar_numero['values'] = (1, 3, 5, 7, 11)

# Posicionar la lista desplegable
seleccionar_numero.grid(column=0, row=1)

# Elemento de la lista seleccionado por default
seleccionar_numero.current(0)
```

Listas desplegables

- **Combobox:**



Botones de Control

Botones de control

■ Checkbutton:

```
# Checkbutton de 3 opciones

# Casilla 1: Deshabilitada ("Disabled")
opcion_1 = tk.IntVar()
casilla_1 = tk.Checkbutton(ventana, text="Leer", variable=opcion_1, state='disabled')
casilla_1.select()
casilla_1.grid(column=0, row=4, sticky=tk.W)

# Casilla 2: No seleccionada ("deselect")
opcion_2 = tk.IntVar()
casilla_2 = tk.Checkbutton(ventana, text="Ver películas", variable=opcion_2)
casilla_2.deselect()
casilla_2.grid(column=1, row=4, sticky=tk.W)

# Casilla 3: Seleccionada ("select")
opcion_3 = tk.IntVar()
casilla_3 = tk.Checkbutton(ventana, text="Redes Sociales", variable=opcion_3)
casilla_3.select()
casilla_3.grid(column=2, row=4, sticky=tk.W)
```

Botones de control

- **Checkbox:**



Botones para opciones

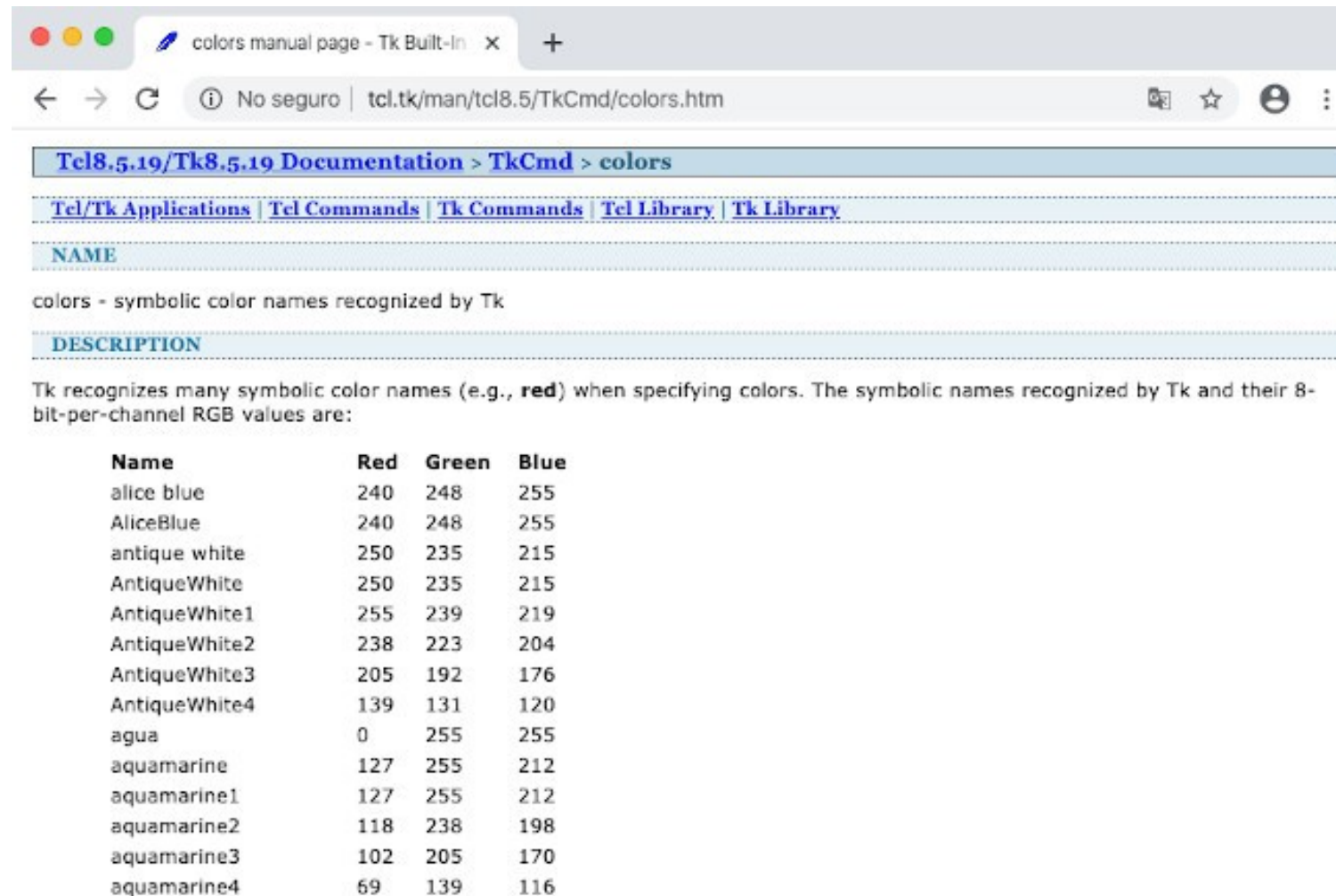
Botones para opciones

- Radiobutton (variables para colores):

```
# Definir colores  
COLOR1 = "Blue"  
COLOR2 = "Gold"  
COLOR3 = "Red"
```

Colores

- **Lista de colores reconocidos por TK:**



Tcl8.5.19/Tk8.5.19 Documentation > TkCmd > colors

[Tcl/Tk Applications](#) | [Tcl Commands](#) | [Tk Commands](#) | [Tcl Library](#) | [Tk Library](#)

NAME

colors - symbolic color names recognized by Tk

DESCRIPTION

Tk recognizes many symbolic color names (e.g., **red**) when specifying colors. The symbolic names recognized by Tk and their 8-bit-per-channel RGB values are:

Name	Red	Green	Blue
alice blue	240	248	255
AliceBlue	240	248	255
antique white	250	235	215
AntiqueWhite	250	235	215
AntiqueWhite1	255	239	219
AntiqueWhite2	238	223	204
AntiqueWhite3	205	192	176
AntiqueWhite4	139	131	120
agua	0	255	255
aquamarine	127	255	212
aquamarine1	127	255	212
aquamarine2	118	238	198
aquamarine3	102	205	170
aquamarine4	69	139	116

Botones para opciones

- **Radiobutton:**

```
# Crear 3 Radiobuttons
```

```
# Radiobutton 1
```

```
opcion = tk.IntVar()
```

```
radio1 = tk.Radiobutton(ventana, text=COLOR1, variable=opcion, value=1, command=funcion_radio)
```

```
radio1.grid(column=0, row=5, sticky=tk.W)
```

```
# Radiobutton 2
```

```
radio2 = tk.Radiobutton(ventana, text=COLOR2, variable=opcion, value=2, command=funcion_radio)
```

```
radio2.grid(column=1, row=5, sticky=tk.W)
```

```
# Radiobutton 3
```

```
radio3 = tk.Radiobutton(ventana, text=COLOR3, variable=opcion, value=3, command=funcion_radio)
```

```
radio3.grid(column=2, row=5, sticky=tk.W)
```

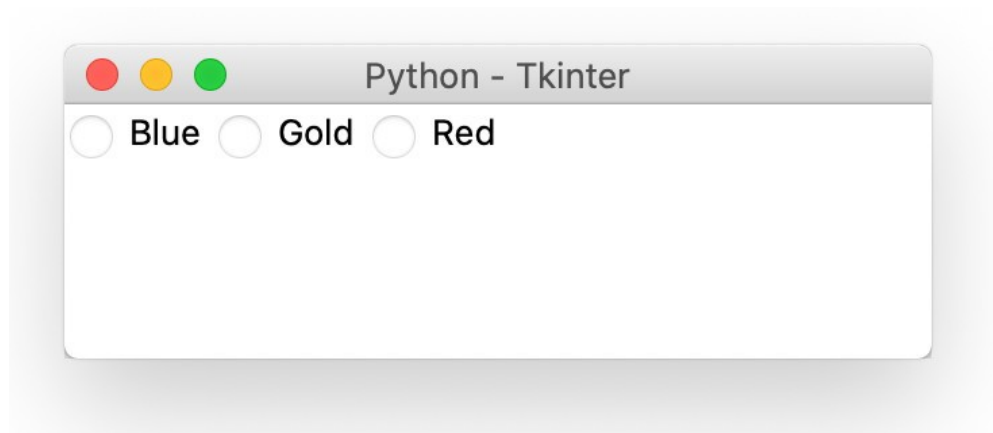
Botones para opciones

- **Radiobutton:**

```
# Función para manejo de los botones
def funcion_radio():
    selector=opcion.get()
    if selector == 1: ventana.configure(background=COLOR1)
    elif selector == 2: ventana.configure(background=COLOR2)
    elif selector == 3: ventana.configure(background=COLOR3)
```

Botones para opciones

- **Radiobutton:**



- **Rabiobutton (rojo seleccionado):**



Cajas de texto
(de varias líneas)

Cajas de texto (varias líneas)

- **Importar librería:**

```
# Importar la librería de las cajas de texto
from tkinter import scrolledtext
```

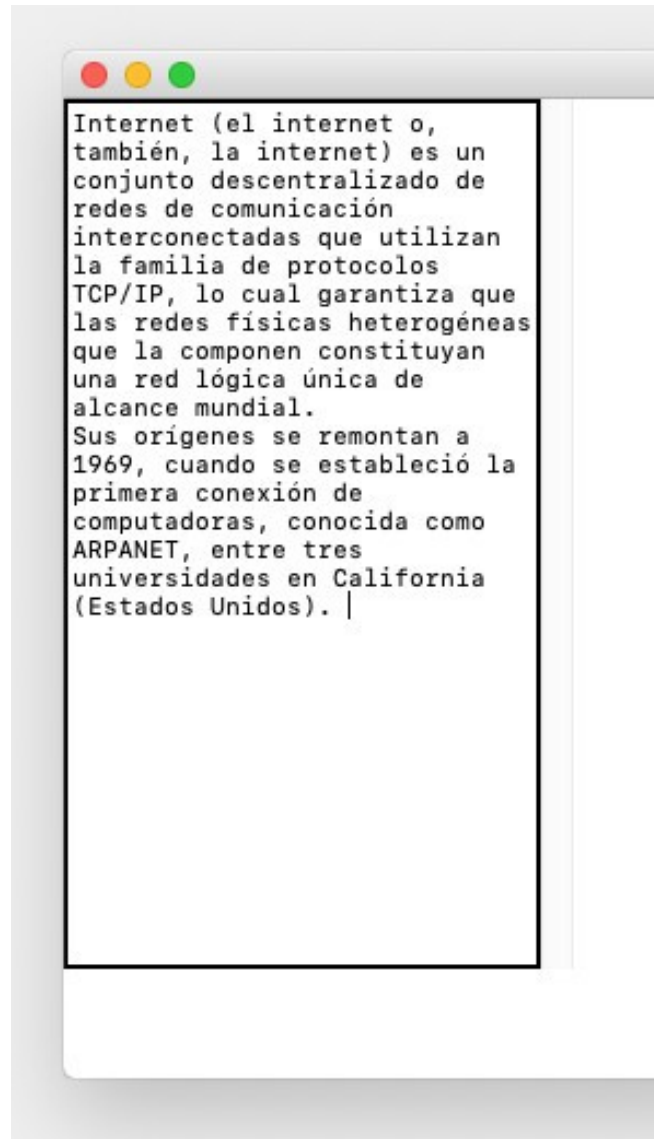
- **Insertar caja de texto:**

```
# Configurar la caja de texto
scrol_ancho = 30
scrol_alto = 3

# Agregar la caja de texto
caja = scrolledtext.ScrolledText(ventana, width=scrol_ancho, height=scrol_alto, wrap=tk.WORD)
caja.grid(column=0, columnspan=3)
```

Cajas de texto (varias líneas)

- **scrolledtext:**



Caja de etiquetas

Cajas de etiquetas

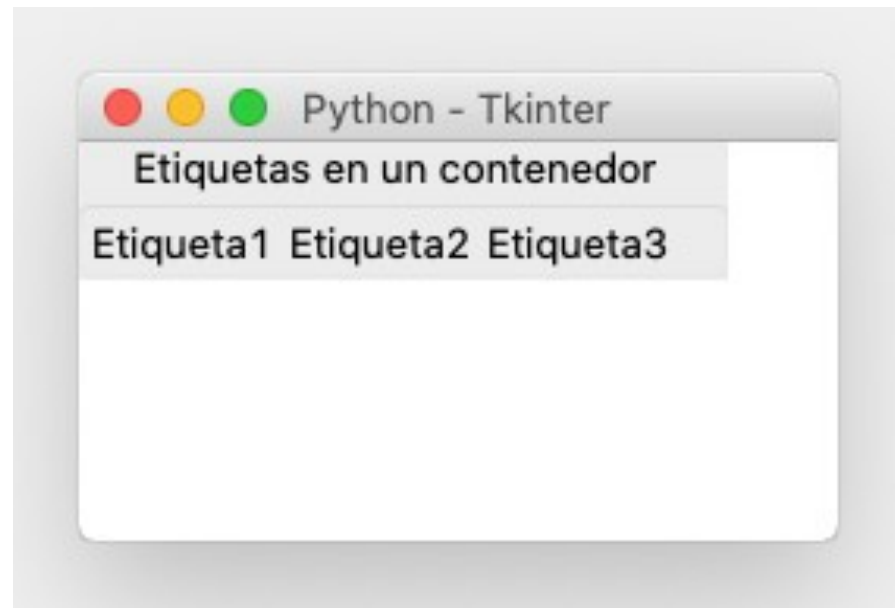
- **LabelFrame** - Permite ubicar varios componentes dentro de un contenedor:

```
# Crear un contenedor para almacenar componentes
contenedor = ttk.LabelFrame(ventana, text=' Etiquetas en un contenedor ')
contenedor.grid(column=0, row=7)

# Etiquetas dentro del contenedor
ttk.Label(contenedor, text="Etiqueta1").grid(column=0, row=0)
ttk.Label(contenedor, text="Etiqueta2").grid(column=1, row=0)
ttk.Label(contenedor, text="Etiqueta3").grid(column=2, row=0)
```

Cajas de etiquetas

- **LabelFrame** - Permite ubicar varios componentes dentro de un contenedor:



Cajas de etiquetas

- Para colocar la posición de inicio del contenedor:

```
contenedor.grid(column=0, row=7, padx=20, pady=40)
```



Barra de Menús

Barra de Menús

- **Importar biblioteca:**

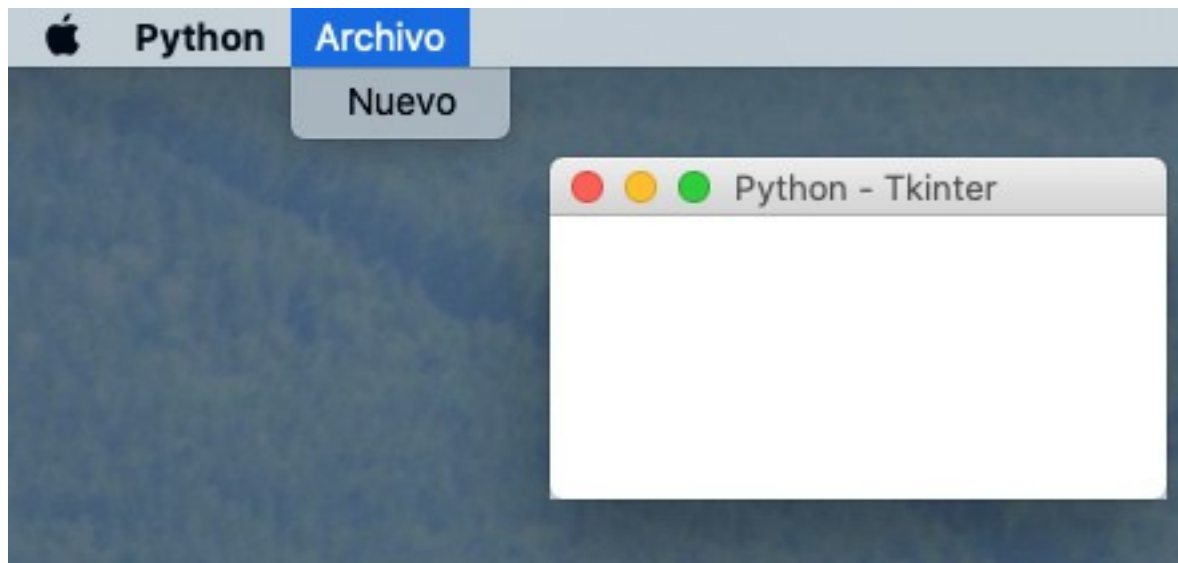
```
# Importar la biblioteca para hacer menús  
from tkinter import Menu
```

- **Crear Menú:**

```
# Crear la barra del menú  
barra_menu = Menu(ventana)  
ventana.config(menu=barra_menu)  
  
# Agregar opciones al menú  
opciones_menu = Menu(barra_menu)  
opciones_menu.add_command(label="Nuevo")  
barra_menu.add_cascade(label="Archivo", menu=opciones_menu)
```

Barra de Menús

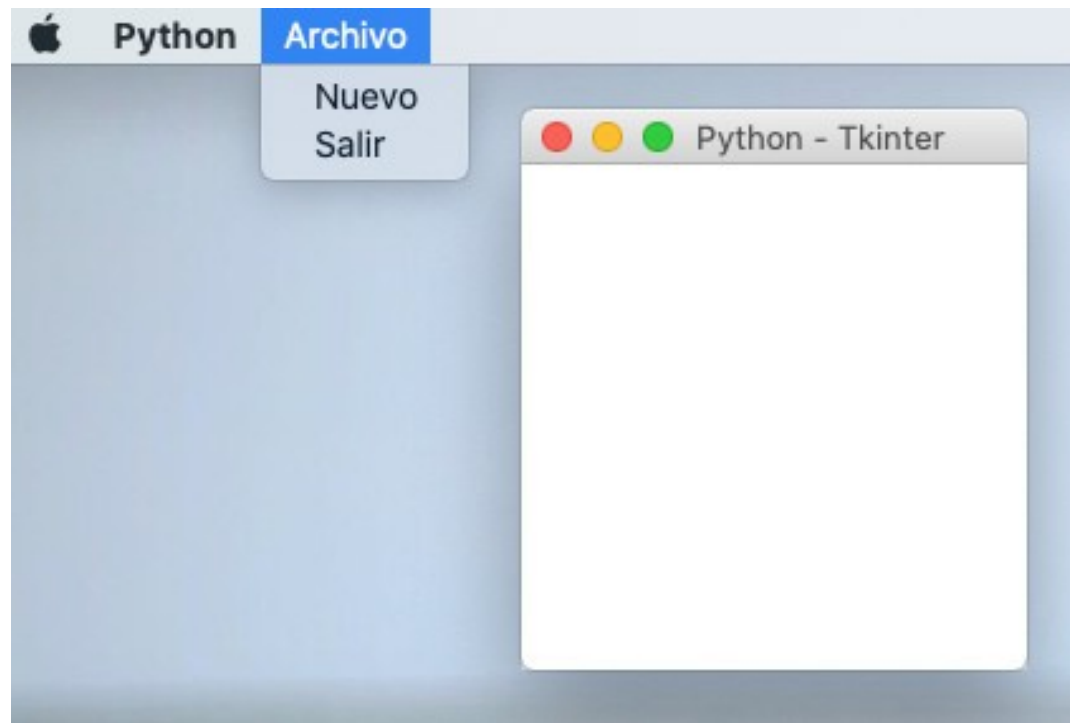
- **Pantalla con menús:**



Barra de Menús

- **Menú con una opción y submenús:**

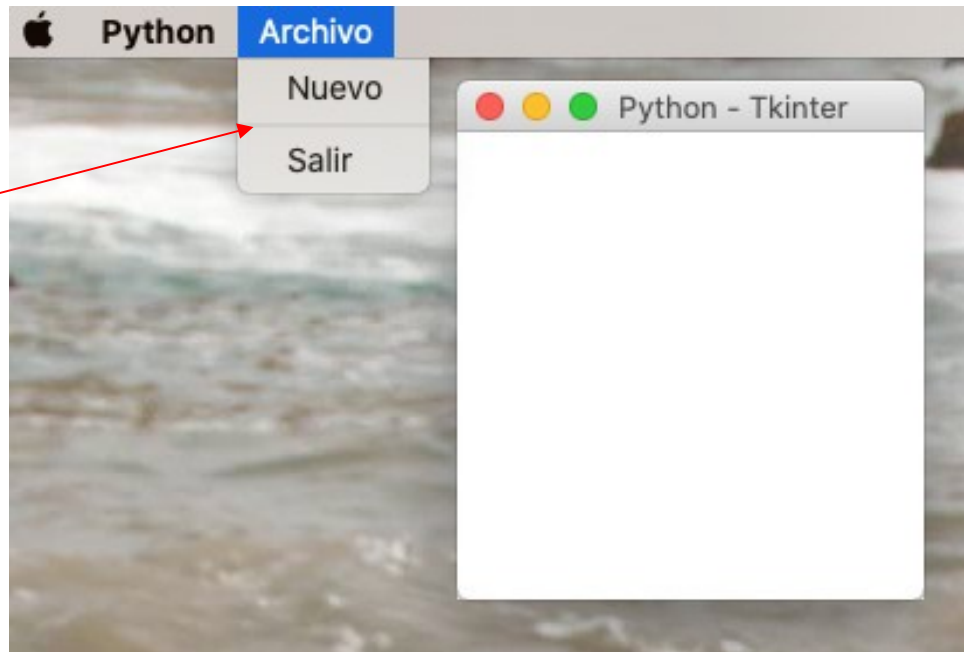
```
# Agregar opciones al menú
opciones_menu = Menu(barra_menu)
opciones_menu.add_command(label="Nuevo")
opciones_menu.add_command(label="Salir")
barra_menu.add_cascade(label="Archivo", menu=opciones_menu)
```



Barra de Menús

- **Menú con 2 opciones y línea divisoria:**

```
# Agregar opciones al menú
opciones_menu = Menu(barra_menu)
opciones_menu.add_command(label="Nuevo")
opciones_menu.add_separator()
opciones_menu.add_command(label="Salir")
barra_menu.add_cascade(label="Archivo", menu=opciones_menu)
```



Barra de Menús

- **Menú con 2 opciones:**

```
# Agregar 2 opciones al menú

# Opción 1: Menú Archivo
menu_archivo = Menu(barra_menu)
menu_archivo.add_command(label="Nuevo")
menu_archivo.add_separator()
menu_archivo.add_command(label="Salir")
barra_menu.add_cascade(label="Archivo", menu=menu_archivo)

# Opción 2: Menú Ayuda
menu_ayuda = Menu(barra_menu, tearoff=0)
menu_ayuda.add_command(label="Acerca de")
barra_menu.add_cascade(label="Ayuda", menu=menu_ayuda)
```



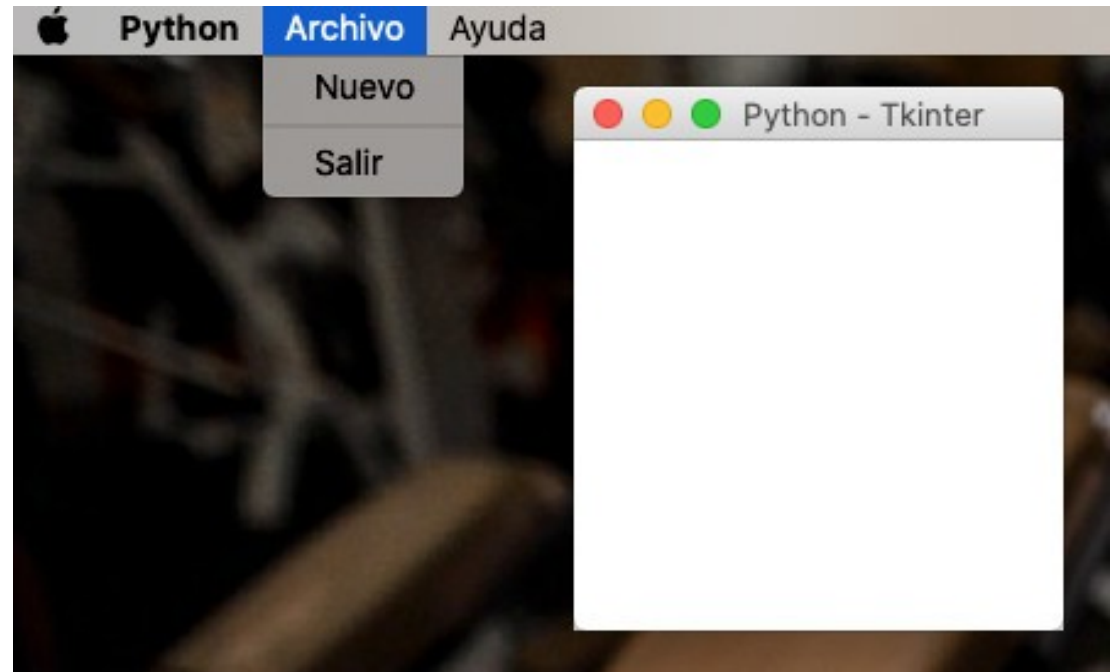
Barra de Menús (salir)

- Evento salir (se modifica el menú para el manejo del evento):

```
menu_archivo.add_command(label="Salir", command=funcion_salir)
```

- Se agrega la función para el manejo del evento

```
def funcion_salir():  
    ventana.quit()  
    ventana.destroy()  
    exit()
```



Pestañas

Pestañas

- **Ventana con pestaña:**

```
# Agregar pestaña (tab)
tabControl = ttk.Notebook(ventana)
tab1 = ttk.Frame(tabControl)
tabControl.add(tab1, text='Tab 1')
tabControl.pack(expand=1, fill="both")
```

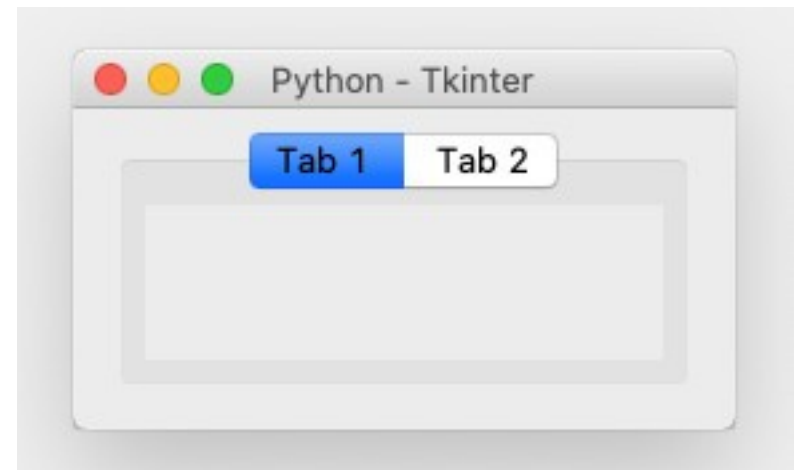


Pestañas

- **Ventana con dos pestañas:**

```
# Agregar pestaña (tab)
tabControl = ttk.Notebook(ventana)
tab1 = ttk.Frame(tabControl)
tabControl.add(tab1, text='Tab 1')
tabControl.pack(expand=1, fill="both")

# Agregar segunda pestaña
tab2 = ttk.Frame(tabControl)
tabControl.add(tab2, text='Tab 2')
```



Cajas de Mensajes

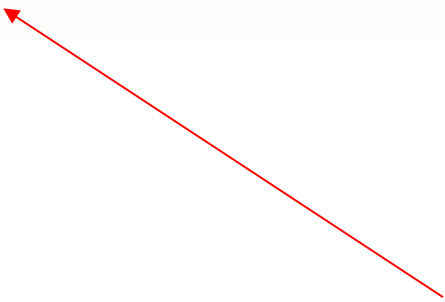
Cajas de mensajes

- **Importar librería para los mensajes (messagebox):**

```
# Importar librería
import tkinter as tk
from tkinter import ttk

# Importar librería para los menús
from tkinter import Menu

# Importar librería para las cajas de mensajes
from tkinter import messagebox as mBox
```



Cajas de mensajes

- **Crear la barra de menús:**

```
# Crear la barra del menú
barra_menu = Menu(ventana)
ventana.config(menu=barra_menu)

# Agregar un menú
menu_ayuda = Menu(barra_menu, tearoff=0)
menu_ayuda.add_command(label="Acerca de", command=funcion_caja_mensaje)
barra_menu.add_cascade(label="Ayuda", menu=menu_ayuda)
```

Cajas de mensajes

- **Función para el manejo del evento:**

```
def funcion_caja_mensaje():  
    mBox.showinfo('Mensaje de Python en una caja de mensajes',  
        'Esta interface fué creada con tkinter\nFebrero 2020.')
```



Cajas de mensajes

```
# Importar librería
import tkinter as tk
from tkinter import ttk

# Importar librería para los menús
from tkinter import Menu

# Importar librería para las cajas de mensajes
from tkinter import messagebox as mBox

def funcion_caja_mensaje():
    mBox.showinfo('Mensaje de Python en una caja de mensajes',
        'Esta interface fué creada con tkinter\nFebrero 2020.')

# Inicializar ventana
ventana = tk.Tk()
ventana.title("Python - Tkinter")

# Crear la barra del menú
barra_menu = Menu(ventana)
ventana.config(menu=barra_menu)

# Agregar un menú
menu_ayuda = Menu(barra_menu, tearoff=0)
menu_ayuda.add_command(label="Acerca de", command=funcion_caja_mensaje)
barra_menu.add_cascade(label="Ayuda", menu=menu_ayuda)

# Activar ventana
ventana.mainloop()
```



Cajas de mensajes

- **Función para el manejo del evento:**

