

CONTROLES CON TKINTER EN PYTHON

¿Qué es Tkinter?

- Biblioteca estándar de interfaces gráficas de Python
- Permite crear ventanas y elementos de interacción visual (widgets)
- Fácil de usar, ideal para principiantes

Clase Tk y algunos métodos

- `Tk()` clase para crear una ventana principal donde añadir los widgets
- `pack()` ubica los widgets en una posición que podemos cambiar a través de los correspondientes atributos.
- `mainloop()` inicia el bucle de mensajes, aquí se **monitorea la interacción del usuario a través del ratón o teclado con la aplicación**, cuando se produzca un evento recibiremos la notificación correspondiente y podremos responder a dicho evento, por ejemplo: el usuario hace clic sobre el botón cerrar, respondemos cerrando la aplicación.

```
import tkinter as tk
ventana = tk.Tk()
ventana.title("Mi ventana")
ventana.geometry("300x200")
ventana.mainloop()
```

VENTANA PRINCIPAL

```
import tkinter as tk
```

Esta línea importa la librería tkinter y le asigna el alias tk. Esta librería permite crear interfaces gráficas (GUI). Usar un alias (tk) facilita referirse a sus elementos sin tener que escribir tkinter completo cada vez.

```
ventana = tk.Tk()
```

Aquí se crea una instancia de la clase Tk, lo cual inicializa una ventana principal. La variable ventana representa esa ventana y se usará para configurarla.

```
ventana.title("Mi ventana")
```

Esta línea define el título que aparecerá en la barra superior de la ventana. En este caso, el título será "Mi ventana".

```
ventana.geometry("300x200")
```

Se define el tamaño de la ventana: 300 píxeles de ancho por 200 píxeles de alto. El formato es "ancho x alto".

```
ventana.mainloop()
```

Inicia el bucle principal de la aplicación.

Es un ciclo que mantiene abierta la ventana y espera eventos (como clics del mouse o teclas presionadas).

Sin esta línea, la ventana se cerraría inmediatamente.

Etiqueta (Label)

Muestra texto estático

```
etiqueta = tk.Label(ventana, text="Hola Mundo")  
etiqueta.pack()
```

Se crea un widget de tipo Label (etiqueta de texto) y se le asigna a la variable etiqueta.tk.

Label indica que estás usando el widget Label de la librería tkinter. ventana es la ventana donde se colocará la etiqueta. text="Hola Mundo" especifica el texto que se mostrará dentro de la etiqueta.

pack() es un método de gestión de geometría que organiza automáticamente el widget en el contenedor (en este caso, la ventana), colocándolo en la parte superior por defecto.

Botón (Button)

Ejecuta una función al hacer clic

```
def saludar():  
    print("Hola!")  
boton = tk.Button(ventana, text="Saludar", command=saludar)  
boton.pack()
```



- Se define una función llamada saludar.
- Cuando esta función se ejecuta, imprime el texto "Hola!" en la consola.
- Esta función será llamada al presionar el botón.
- Se crea un botón y se lo asigna a la variable boton.tk.Button(...)
- crea el botón.ventana: indica en qué ventana se va a colocar.text="Saludar": es el texto que se verá en el botón.command=saludar: enlaza el botón con la función saludar.
- Al hacer clic, se ejecuta esa función (¡sin paréntesis!, para que no se ejecute al cargar la ventana).
- Coloca (empaqueta) el botón dentro de la ventana, igual que hicimos con la etiqueta antes.

Campo de Entrada (Entry)

Para ingresar texto

```
entrada = tk.Entry(ventana)
entrada.pack()
texto = entrada.get()
```

- Se crea un widget de entrada (Entry) donde el usuario podrá escribir texto. ventana indica dónde se va a colocar. La variable entrada permite acceder al contenido del campo después.
- Coloca el campo de entrada en la ventana. pack() lo ubica automáticamente.
- Obtiene el texto que el usuario escribió en el campo entrada y lo guarda en la variable texto.

 Importante: Esta línea debe ejecutarse después de que el usuario haya escrito algo (por ejemplo, dentro de una función que se llama al presionar un botón).

Si se llama inmediatamente, se ejecutará antes de que el usuario pueda escribir.

Área de Texto (Text)

Texto largo con varias líneas

```
caja = tk.Text(ventana, height=5, width=30)  
caja.pack()
```

`tk.Text(...)` crea un widget de texto que permite ingresar o mostrar múltiples líneas.
`ventana`: es la ventana donde se insertará el widget.
`height=5`: define la altura del área en número de líneas.
`width=30`: define el ancho en cantidad de caracteres.

Coloca (empaqueta) el widget caja dentro de la ventana.

`"1.0"`: indica desde la línea 1, carácter 0 (es decir, el inicio).
`"end-1c"`: significa hasta el final del contenido menos 1 carácter, para evitar que se copie el salto de línea final automáticamente incluido.

Casilla de Verificación (Checkbox)

```
var = tk.IntVar()  
check = tk.Checkbutton(ventana, text="Acepto", variable=var)  
check.pack()
```

Se crea una variable de tipo entero (IntVar) que almacenará el estado del checkbox: Valor 1 si está marcado Valor 0 si está desmarcado

Crea el checkbox y lo coloca en la ventana.`text="Acepto"` es el texto visible junto a la casilla.`variable=var` vincula el estado del checkbox con la variable `var`, que puede ser consultada con `var.get()`.

Agrega el checkbox a la ventana con el método `pack()`.

Botones de Opción (Radiobutton)

```
opciones = ["Rojo", "Verde", "Azul"]  
seleccion = tk.StringVar(value=opciones[0])  
menu = tk.OptionMenu(ventana, seleccion, *opciones)  
menu.pack()
```


- Se define una lista con las opciones que estarán disponibles en el menú desplegable.
- Se crea una variable de tipo `StringVar`, que almacenará la opción seleccionada del menú.
- Se inicializa con "Rojo" (la primera opción de la lista).
- Crea el widget desplegable (`OptionMenu`) y lo coloca en la `ventana.ventana`: contenedor donde va el menú.
- `seleccion`: variable vinculada que guarda el valor seleccionado.
- `*opciones`: usa el operador
- `*` para descomponer la lista y pasar los elementos como argumentos individuales.
- Agrega el menú desplegable a la ventana.

Lista Desplegable (OptionMenu)

```
lista = tk.Listbox(ventana)
lista.insert(1, "Python")
lista.insert(2, "Java")
lista.insert(3, "C++")
lista.pack()
```

Se crea un widget de tipo Listbox y se asigna a la variable lista.

Este widget permite mostrar una lista de ítems de los que el usuario puede seleccionar uno o varios.

insert(posición, texto) agrega un ítem en una posición específica.

Aunque comienza en 1, en tkinter se recomienda empezar desde índice 0, ya que la numeración es cero-indexada.

Lo correcto sería:

```
lista.insert(0, "Python")  
lista.insert(1, "Java")  
lista.insert(2, "C++")
```

Agrega el Listbox a la ventana principal para que sea visible.

EJEMPLO INTEGRADO

A vertical white line is positioned to the right of the text, extending from the top of the word 'EJEMPLO' down to the bottom of the word 'INTEGRADO'.

```
import tkinter as tk

def mostrar():
    nombre = entrada.get()
    etiqueta.config(text=f"Hola {nombre}!")

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

entrada = tk.Entry(ventana)
entrada.pack()
```

```
import tkinter as tk
```

- **Importa** la biblioteca `tkinter`, que permite crear interfaces gráficas.
- Se le da el **alias** `tk` para que sea más corto llamar a los elementos (por ejemplo: `tk.Label`, `tk.Button`, etc.).

```
def mostrar():
```

- **Define una función** llamada `mostrar` que se ejecutará cuando el usuario haga clic en el botón.

```
nombre = entrada.get()
```

- Obtiene el **texto escrito por el usuario** en el campo de entrada `entrada` y lo guarda en la variable `nombre`.

```
etiqueta.config(text=f"Hola {nombre}!")
```

- **Actualiza el texto de la etiqueta** con el mensaje "Hola" seguido del nombre que se ingresó.
- `f"Hola {nombre}!"` es una **f-string** que inserta el valor de la variable `nombre` dentro del texto.

```
boton = tk.Button(ventana, text="Mostrar saludo", command=mostrar_saludo)
boton.pack()
```

```
etiqueta = tk.Label(ventana, text="")
etiqueta.pack()
```

```
ventana.mainloop()
```

```
ventana = tk.Tk()
```

- Crea la **ventana principal** de la aplicación gráfica.
-

```
ventana.title("Ejemplo Tkinter")
```

- Establece el **título** que aparecerá en la barra de la ventana.

```
entrada = tk.Entry(ventana)
```

- Crea un **campo de entrada de texto** dentro de la ventana principal.
- `Entry` es el widget que permite al usuario escribir texto.

```
entrada.pack()
```

- Coloca el campo de entrada dentro de la ventana utilizando el método `pack()`, que organiza automáticamente los elementos de arriba hacia abajo.


```
boton = tk.Button(ventana, text="Mostrar saludo", command=mostrar)
```

- Crea un **botón** con el texto "Mostrar saludo".
- El parámetro `command=mostrar` le dice al botón que, cuando sea presionado, ejecute la función `mostrar` definida arriba.

```
boton.pack()
```

- Coloca el botón en la ventana, también con `pack()`.

```
etiqueta = tk.Label(ventana, text="")
```

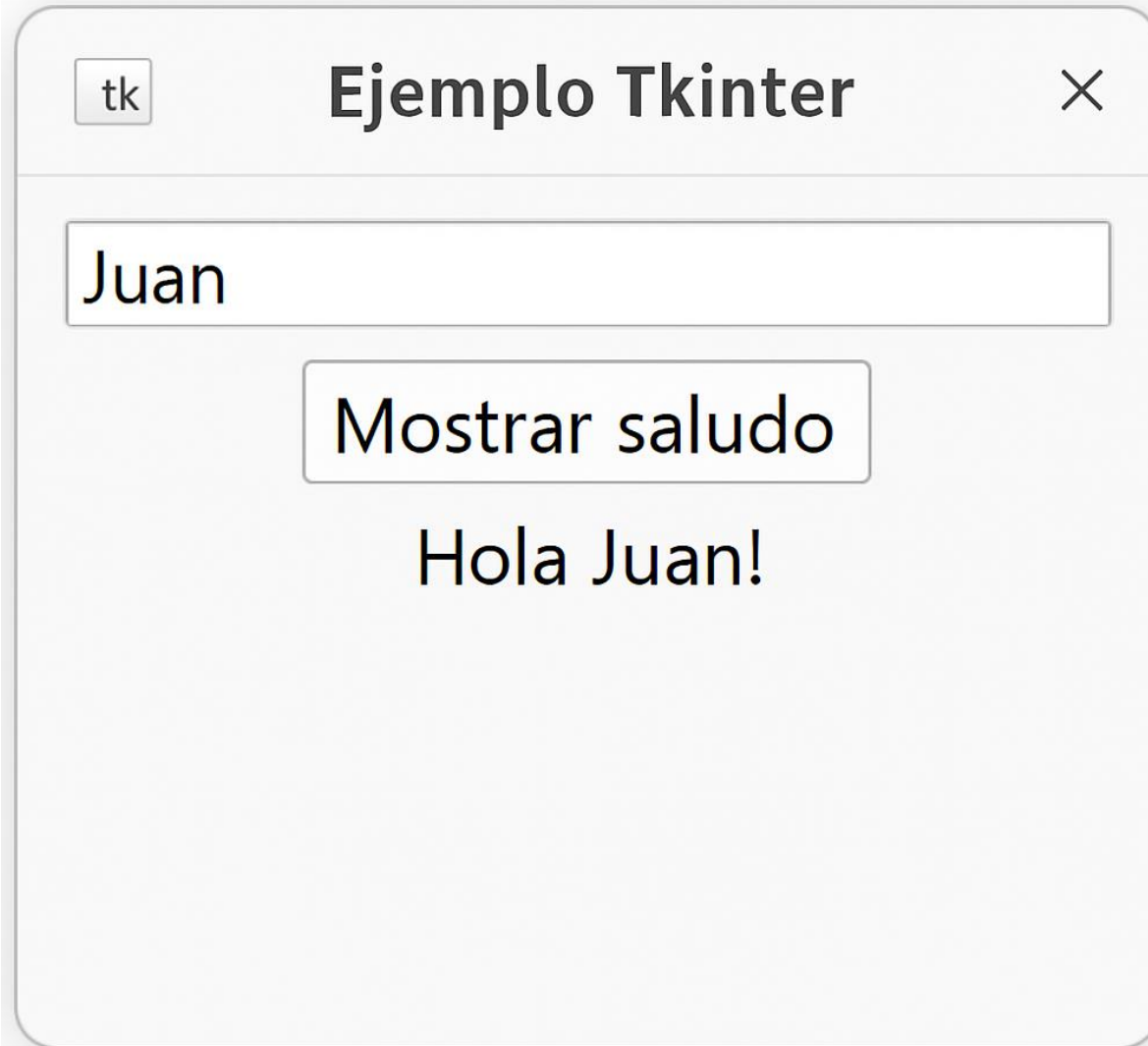
- Crea una **etiqueta vacía**. Esta se actualizará cuando el usuario presione el botón.
- `Label` es el widget para mostrar texto estático.

```
etiqueta.pack()
```

- Coloca la etiqueta en la ventana.

```
ventana.mainloop()
```

- Inicia el bucle principal de la interfaz gráfica.
- La ventana queda abierta y esperando eventos (clics, escritura, etc.) hasta que el usuario la cierre.



```
import tkinter as tk

def mostrar():
    nombre = entrada.get()
    etiqueta.config(text=f"Hola {nombre}!")

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

entrada = tk.Entry(ventana)
entrada.pack()

boton = tk.Button(ventana, text="Mostrar saludo", command=mostrar)
boton.pack()

etiqueta = tk.Label(ventana, text="")
etiqueta.pack()

ventana.mainloop()
```

MOSTRAR SALUDO PERSONALIZADO CON ENTRADA DE TEXTO

