**Módulo de 15:**

# **Graphical User Interface:**

Algunos widgets de tkinter son:

* windows o root
* windows = Tk.tk()
* Frame(Marco)
* Label(Etiqueta)
* Entry (Campo de entrada)
* Button (Botón)
* Text (Texto)
* Listbox (Lista)
* Checkbutton (Casilla de verificación)
* Radiobutton (Botón de opción)
* Menu (Menú)

Descripción de los atributos comunes en Tkinter:

ubicación

* pack()
* place()
* grid()

**Graphical User Interface:**

|  |
| --- |
| En el curso solo se veran tres widgets. Lable, entry y buton. En funciones (aun no el clases) |

No podemos comparar lo que es una presentación grafica de una en consola. La(s) ventana(s) se conoce como capa gráfica, esta se agrega a la logica general.

Tengan en cuenta que la ventana GUI necesita iniciarse y al final se coloca un mainloop() que cierra el bucle de refresco de pantalla.

Dentro se configuran los widgets y su ubicación.

Si hay botones generalmente llama mediante command a una funcion (o método) que es llamado al clickear en el. Estos no admiten parámetros si no se usan lambdas.

En consola usabamos print para enviar informacion al usuario. Aquí tenemos labels o text dependiendo la cantidad de lineas del string a mostrar.

La entrada de datos en consola a hacemos con input aquí usamos mayormente entry pero tengan en cuenta que en consola se carga el dato cuando se presiona la tecla enter al final del dato introducido por el usuario.

En entry necesitamos capturar el dato cuando esto se pida mediante, por ejemplo, con una función donde con .get() recien se guarda en un objeto la informacion suministrada por el usuario.

En consola el orden de muetra en consola se debe según el flujo en que fue escrito.

Si hay un print, un input y otro print. El ultimo solo se vera en cuanto se haya ingresado el dato en el input. En consola no se superponen las saludas ya que una va siempre debajo de la otra.

En consola si tengo un label, un entry y otro label todos los widgets se mostraran dentro de la ventana normalmente al mismo tiempo.

Ademas cada widget debera ser ubicado en una posición diferente para que no se superpongan.

·

|  |
| --- |
| Tkinter es una biblioteca estándar de Python que proporciona una interfaz de programación para crear aplicaciones de escritorio multiplataforma con una interfaz gráfica. Tk es una biblioteca gráfica + interface escritas en Tcl (Tool Command Language).   * Tkinter es nativa en la instalación estándar, no se requiere una instalación adicional. * Es una biblioteca multiplataforma, lo que significa que las aplicaciones creadas con Tkinter pueden ejecutarse en diferentes sistemas operativos, como Windows, macOS y Linux. * Proporciona una amplia gama de widgets y herramientas para la creación de interfaces gráficas, como botones, etiquetas, campos de entrada, menús desplegables, etc. * Permite la personalización de la apariencia y el comportamiento de los widgets mediante propiedades y opciones. * Es altamente integrable con otras bibliotecas y módulos de Python. |

·

|  |  |
| --- | --- |
| Tkinter proporciona una amplia gama de widgets predefinidos que se pueden utilizar para construir la interfaz de usuario de una aplicación.  Algunos widgets de tkinter son: | |
| **windows o root**  (Ventana principal) | windows = Tk.tk()  Un contenedor general para agrupar y organizar otros widgets. |
| **Frame**  (Marco) | Uno o mas contenedores secundarios rectangular que se utiliza para agrupar y organizar otros widgets dentro de la ventana (windows). |
| **Label**  (Etiqueta) | Un widget que se utiliza para mostrar texto cortos o imágenes. |
| **Entry**  (Campo de entrada) | Un widget que permite al usuario ingresar texto. |
| **Button**  (Botón) | Un widget que se utiliza para realizar una acción al hacer clic en él. Permite mostrar texto cortos o imágenes |
| **Text**  (Texto) | Un widget que permite al usuario ingresar texto largos. |
| **Listbox**  (Lista) | Un widget que muestra una lista de elementos de donde se puede seleccionar uno o varios. |
| **Checkbutton**  (Casilla de verificación) | Un widget que permite al usuario seleccionar una o varias opciones. |
| **Radiobutton**  (Botón de opción) | Un widget que permite al usuario seleccionar solo una opción de un grupo de opciones mutuamente excluyentes. |
| **Menu**  (Menú) | Un widget que proporciona opciones de menú desplegable en una barra de menú.  Estos son solo algunos ejemplos de widgets disponibles en tkinter. |
| Cada widget tiene sus métodos y atributos para personalizar su apariencia y comportamiento en la interfaz de usuario.  Alguno de ellos como bg – fondo son utilizados en varios widgets diferentes | |

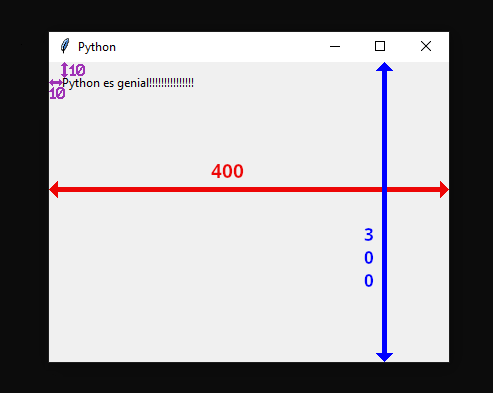
·

|  |  |
| --- | --- |
| Descripción de los atributos comunes en Tkinter: | |
| **x** | Ubicación sobre el eje x – horizontal - es un valor en píxeles con respecto al borde izquierdo del campo de texto |
| **y** | Ubicación sobre el eje y – vertical  root.  Una observacion que se debe hacer es que los valores que tome tanto el ancho como el alto de la ventana ademas de ser unicamente en pixeles deberan de ser numeros enteros positivos.  El significado del signo mas y menos indica que si se le pasa un valor positivo este sera contado desde la esquina superior izquierda de la pantalla en cambio si le pasamos un valor negativo este sera tomado en cuenta partiendo como origen desde la esquina inferior derecha; este es un detalle que hay que tener muy en cuenta cuando deseamos posicionar una ventana.  Si no le dieramos argumentos nos devolveria la medida y la posicion de la ventana.  Argumentos:  "wxh±x±y"  Donde las letras tienen este significado:  w: Ancho de la ventana en pixeles  h: Alto de la ventana en pixeles  x: Posicion en el eje X  y: Posicion en el eje Y |
| .**geometry**  ("WxH") | Damos el tamaño de la ventana  ventana.config("400x300")  o  ventana.config(width=400, height=300 |
| .**geometry**  ("WxH+  X+Y") | Damos el tamaño de la ventana y ls ubicación de estventana en la pantalla  ventana.config("400x300"+100+100)  o  ntana.config(width=400, height=300+100+100) |
| **height** | Define la altura en líneas no en pixeles del componente. |
| **width** | Indicamos el ancho del control, no en píxeles, sino en caracteres del componente |
| **font** | Permite especificar la fuente a utilizar en el texto del componente. Se puede pasar una tupla con el nombre de la fuente, el tamaño y el estilo, por ejemplo: Font("Times New Roman", 24, "bold underline"). |
| **bg** | Modifica el color de fondo. Puedes indicar el color en inglés (incluyendo modificadores, como "darkgreen") o su código RGB en hexadecimal ("#aaaaaa" para blanco).  En MacOS, no se puede modificar el color de fondo de los botones, pero se puede configurar el highlightbackground, que pinta el fondo alrededor del botón con el color especificado. |
| **fg** | Cambia el color del texto. |
| **bd** | Modifica el ancho del borde del widget. |
| **cursor** | Modifica la forma del cursor. Algunas opciones comunes son "arrow", "crosshair", "hand", "ibeam", etc. |
| **relief** | Cambia el estilo del borde del componente.  **flat:**  No se muestra ningún borde.  **solid:**  Borde solido  **sunken:**  Borde hundido, que provoca que el elemento que encierra parezca que se encuentra por debajo del nivel de la superficie de la pantalla.  **raised:**  Borde saliente, que provoca que el elemento que encierra parezca que se encuentra por encima del nivel de la superficie de la pantalla.  **groove:**  Borde hundido, que visualmente parece que se encuentra por debajo del nivel de la superficie de la pantalla.  **ridge:**  Borde saliente, que visualmente parece que se encuentra por encima del nivel de la superficie de la pantalla.  etiqueta = tk.Label(root, text="Hola Mundo!!!", relief="sunken",borderwidth=5) |
| **state** | permite deshabilitar el componente.  Puede establecerse como DISABLED, por ejemplo, para una Label en la que no se pueda escribir o un Button que no se pueda hacer clic. |
| **padding** | Define el espacio en blanco alrededor del widget en píxeles . |
| **command** | Especifica la función que se ejecutará cuando se haga clic en un botón.  No acepta parámetros.  Para poder enviarlos se requiere una fncion lambda |
| **justify** | etiqueta = tk.Label(root, text="Hola\n Mundo!!!",  justify="right")  justify="center")  justify="left") |

.

|  |  |
| --- | --- |
| Hay tres métodos principales para ubicar y organizar widgets en una ventana o marco: | |
| **pack()** | Aqui los widgets se colocan uno debajo del otro o uno al lado del otro, según la dirección especificada (vertical u horizontal).  Los widgets se ajustan automáticamente al tamaño del contenido y al tamaño de la ventana.  Es útil cuando deseas una disposición rápida y sencilla de los widgets. |
| **place()** | Puedes especificar manualmente la ubicación exacta de un widget utilizando coordenadas X e Y. (una batalla naval)  También puedes controlar el ancho y alto del widget. Esto te da un control preciso sobre la ubicación y el tamaño de los widgets en la ventana.  Sin embargo, es importante tener en cuenta que el uso excesivo de place() puede hacer que la interfaz gráfica sea menos flexible y menos adaptable a diferentes tamaños de ventana o cambios en el contenido. |
| **grid()** | Permite organizar los widgets en una estructura de filas y columnas, similar a una tabla (similar a una hoja de calculo).  Puedes especificar en qué fila y columna deseas ubicar cada widget, y también puedes controlar el ancho y alto de las celdas en la cuadrícula. grid() es especialmente útil para diseños más complejos en los que necesitas organizar los widgets en una estructura tabular. |

·

·

|  |
| --- |
| import tkinter as tk  ventana = tk.Tk()  ventana.config(width=400, height=300)  ventana.title("Python")  #---------------------------------------------------------------------  etiqueta\_nombre = tk.Label(text="Python es genial!!!!!!!!!!!!!!! ")  etiqueta\_nombre.place(x=10, y=10 , width=350, height=20)  ventana.mainloop() |

### ·

|  |
| --- |
| def salir():  exit()  ventana = tk.Tk()  ventana.config(width=400, height=300)  ventana.title("Python")  #---------------------------------------------------------------------  etiqueta\_nombre = tk.Label(text="Nombre alumno:")  etiqueta\_nombre.place(x=10, y=10)  #---------------------------------------------------------------------  caja\_nombre = tk.Entry()  caja\_nombre.place(x=110, y=10, width=150, height=20)  #---------------------------------------------------------------------  boton\_lista = tk.Button(text="salir", bg='#A8A8A8', command=salir)  boton\_lista.place(x=110, y=50, width=150, height=20)  #---------------------------------------------------------------------  text\_ = tk.Text( )  lista = f"""  Es fácil de aprender y usar, especialmente para aquellos que están familiarizados con Python.  Es una biblioteca multiplataforma, lo que significa que las aplicaciones creadas con Tkinter pueden ejecutarse en diferentes sistemas operativos, como Windows, macOS y Linux.  Proporciona una amplia gama de widgets y herramientas para la creación de interfaces gráficas, como botones, etiquetas, campos de entrada, menús desplegables, etc.  Permite la personalización de la apariencia y el comportamiento de los widgets mediante propiedades y opciones.  Es altamente integrable con otras bibliotecas y módulos de Python."""  text\_.insert(tk.END, lista)  text\_.place(x=10, y=100, width=380, height=180)  #---------------------------------------------------------------------  ventana.mainloop() |

### ·

