6. Benutzeroberflächen

In diesem und im nächsten Kapitel werden wir uns intensiv mit **Benutzeroberflächen** (GUI = Graphical User Interface) beschäftigen.

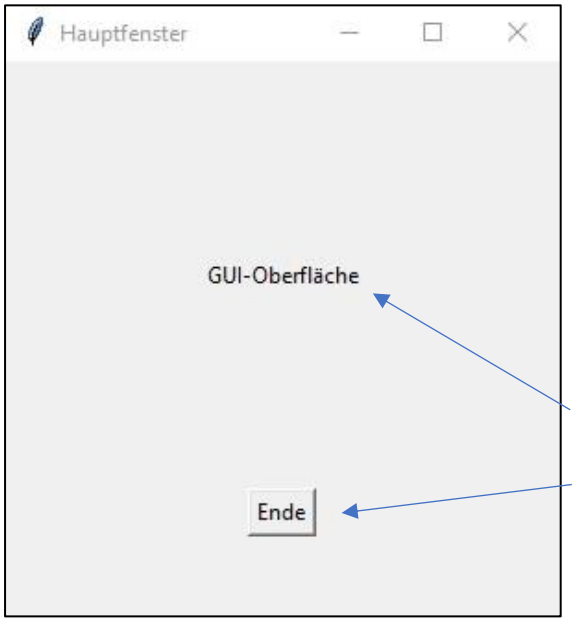
Das Modul **tkinter** stellt eine Schnittstelle zu der Bibliothek **Tk** dar. Es bietet verschiedene Klassen zur Erzeugung von Elementen auf einer Benutzeroberfläche. Der Programmablauf wird durch **Ereignisse** (Maus- und Tastatureingabe des Benutzers) bestimmt. Es wird ereignisbasiert programmiert.

Die Aktionen des Benutzers auf der Benutzeroberfläche werden über eine Endlosschleife abgefangen. Im Modul **tkinter** heißt die spezielle Schleife **mainloop()**. Dabei werden

a.) das Hauptfenster und
b.) die Steuerelemente (engl. Widgets)

erzeugt.

1	import tkinter	← import des Moduls tkinter
2	def ende():	
3	main.destroy()	
4	main = tkinter.Tk()	Hauptfenster
5	main.title("Hauptfenster")	← Titelleiste des Fensters
6	main.geometry("300x300")	← Größe des Fensters in px
7	b = tkinter.Button(main, text = "Ende", command = ende	Schaltfläche Ende (=Steuerelement)
8	b.pack()	← Geometriemanager
9	main.mainloop()	Endlosschleife



← **Titelleiste** (Kopfzeile)
mit den Symbolen – ☐ ✕
für das Verkleinern, Vergrößern und Schließen des Fensters

← **Steuerelemente**
(z.B. Schaltflächen, Felder, Bilder, Texte)
Die Steuerelemente werden auf der GUI-Oberfläche des Hauptfensters platziert.

Text	Bezeichnung, Hinweis, Information, Überschrift
Schaltfläche	klick mit der linken Maus-Taste

Abb.24: Erstes GUI-Fenster

Der **User** ist aktiv per **Maus oder Tastatur → Die Programmierung basiert auf den Ereignissen.**

Erweitern wir das obige Python-Beispielprogramm, um eine Grafik „logo.png“ (**tkinter.Photoimage**) und eine kurze Beschriftung „Das erste GUI-Programm“ (**tkinter.Label**).

logo.png (115x115)

Positionieren Sie die Steuerelemente mittig ausgerichtet.

55_GUI_v1.py - C:\Users\PAZ20\AppData\Local\Programs\Python\Python39\55_GUI_v1.py (3....

File Edit Format Run Options Window Help

```
import tkinter

def ende():
    main.destroy()

# Hauptfenster in der Größe 300 x 300 px
main = tkinter.Tk()
main.geometry("300x300")

# Beschriftung oder Überschrift
txt = tkinter.Label(main, text ="Das erste GUI-Programm")
txt.pack()

# logo.png (115px) als Bild
lab = tkinter.Label(main)
im = tkinter.PhotoImage(file="logo.png")
lab["image"] = im
lab.pack()

# ENDE-Schaltfläche (Button)
btn = tkinter.Button(main, text ="Ende", command = ende)
btn["anchor"]="center"
btn.pack()

# Endlosschleife = Hauptprogramm
main.mainloop()
```

GUI-Programm (300x300px)

Abb. 25: Python-Programm: GUI mit Beschriftung, Bild und ENDE-Schaltfläche

Die **Anordnung der Steuerelemente** auf dem Hauptfenster erfolgt mit einem der Geometrie-manager, die das Layout einer GUI-Anwendung bestimmen.

pack() - ist ein einfacher Geometriemanager

grid() - an einem Raster angeordnet: Zeilen (engl. row) und Spalten (column)

place() - an einem Raster angeordnet: absolute bzw. relative Koordinaten in Pixel

Im Python-Kurs verwenden wir nur den **pack()**-Geometriemanager.

Übersicht der möglichen Steuerelemente (engl. Widgets)

→ für die **Eingabe**

Entry	einzeiliges Textfeld
Text*	mehrzeiliges Textfeld
Listbox*	eine Liste von Optionen
Spinbox	eine Liste von Optionen wählbar
Checkbutton	Optionen als Kontrollkästchen (Mehrfachauswahl)
Radiobutton	Optionen als Optionsfelder (nur eine Option wählbar)

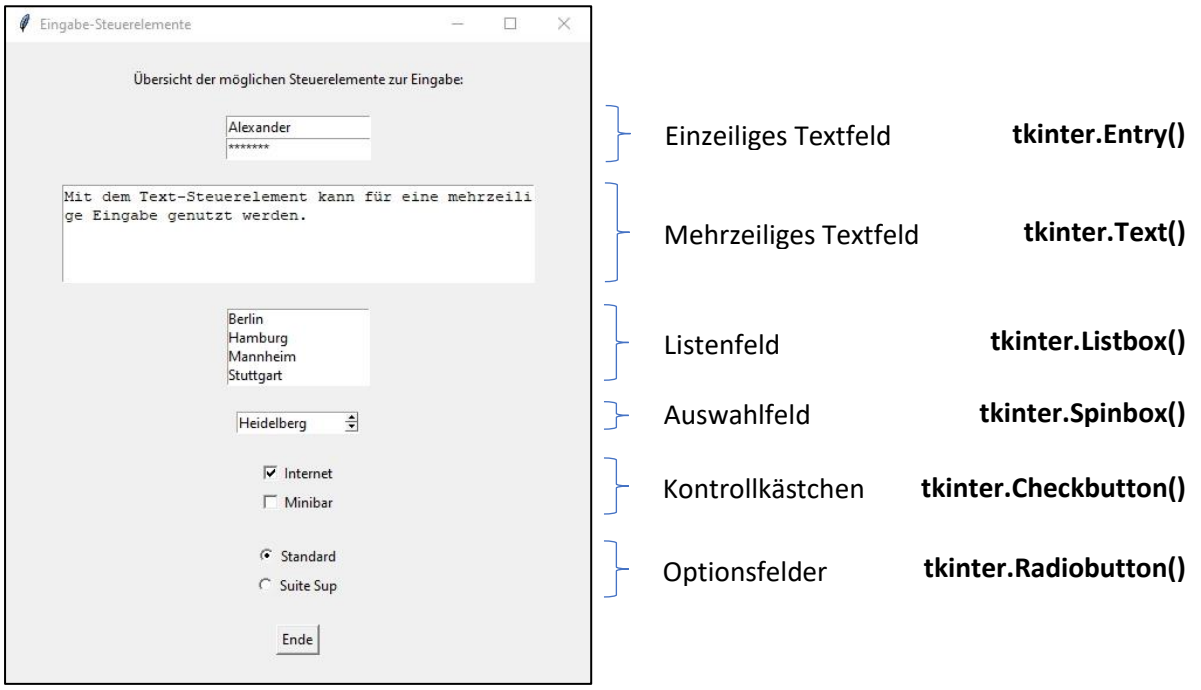


Abb. 26: tkinter: Steuerelemente für die Eingabe

* Das mehrzeilige Textfeld (**tkinter.scrolledtext.ScrolledText**) und auch das Listenfeld können mit einer **Bildlaufleiste** (**tkinter.Scrollbar**) angezeigt werden und ermöglicht das vertikale Scrollen innerhalb des Steuerelementes.

→ für die **Anordnung der Steuerelemente**

Frame	zum Organisieren der Steuerelemente mit Rahmen
PanedWindow	horizontal oder vertikal Bereiche anordnen
LabelFrame	Abstandshalter

Mit Frame (**tkinter.Frame**) und mit Abstandhaltern (**tkinter.LabelFrame**) lassen sich die Steuerelemente zu einer Gruppe zusammenfassen und abgrenzen. Eine einfache Art der horizontalen oder vertikalen Anordnung der Steuerelemente ermöglicht das **PanedWindow()**. Diese werden per **add()-Funktion** einem PanedWindow angehängt.

→ für die **Ausgabe**

Label**	einzeilige Text-Beschriftung (auch Bilder)
Message	Text in mehreren Zeilen anzeigen

**(auch Bilder)

57_GUI_SE_Ausgabe.py - C:\Users\PAZ20\AppData\Local\Programs\Python\Python39\57_GUI...

File Edit Format Run Options Window Help

```
import tkinter

def ende():
    main.destroy()

# Hauptfenster in der Größe 300 x 300 px
main = tkinter.Tk()
main.title("Ausgabe-Steuerelemente")
main.geometry("300x300")

# Beschriftung oder Überschrift
txt = tkinter.Label(main, text ="einzeilige Ausgabe")
txt.pack()

# für längere Texte
txtlang = tkinter.Message(main, text="Längerer Text über mehrere Zeilen. \
Die Zeilen sind immer gleich lang.")
txtlang.pack()

# lovherbst.png (150x100px) als Bild
lab = tkinter.Label(main)
im = tkinter.PhotoImage(file="lovherbst.png")
lab["image"] = im
lab.pack()

# ENDE-Schaltfläche (Button)
btn = tkinter.Button(main, text ="Ende", command = ende)
btn["anchor"]="center"
btn.pack()

# Endlosschleife = Hauptprogramm
main.mainloop()
```

Abb. 27: tkinter: Steuerelemente für die Ausgabe

GUI-Programm (Ausgabe)

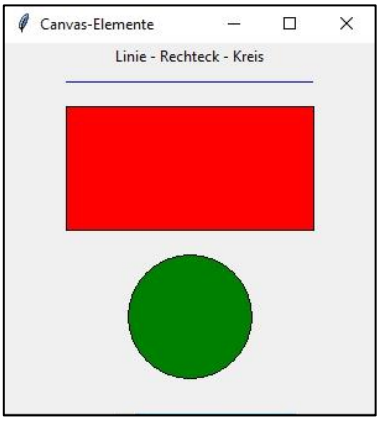
→ für die grafische **Anzeige**

Canvas	Formen wie Linien, Ovale, Polygone und Rechtecke zeichnen

58_GUI_SE_Canvas.py - C:/Users/PAZ20/AppData/Local/Programs/Python/Python39/58_GUI_...

File Edit Format Run Options Window Help

```
import tkinter

# Hauptfenster in der Größe 300 x 300 px
main = tkinter.Tk()
main.title("Canvas-Elemente")
main.geometry("300x300")

# Beschriftung oder Überschrift
txt = tkinter.Label(main, text ="Linie - Rechteck - Kreis")
txt.pack()

# Canvas
canvas = tkinter.Canvas(main)
canvas.pack()

# einfache Linie der Länge 200 (x1, y1, x2, y2, options=...)
canvas.create_line(50, 10, 250, 10, fill="blue")

# Rechteck 200 x 100 (b x h)
canvas.create_rectangle(50,30,250,130, fill="red")

# Kreis Punkt 150/200, Radius 50
canvas.create_oval(150-50,200-50,150+50,200+50, fill="green")

# Endlosschleife = Hauptprogramm
main.mainloop()
```



GUI-Programm (Zeichenelemente)

Abb. 28: tkinter: Canvas-Zeichenelemente