

Hinweise zur Entwicklung von UI mit Python/Tkinter

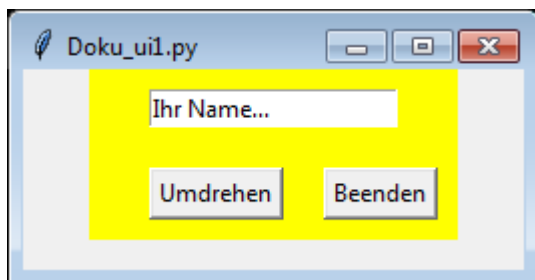
wichtige Parameter der pack-Funktion:

Parameter der pack-Methode:

Parameter	Werte	Erläuterung
anchor	"n", "ne", "e", "se", "s", "sw", "w", "nw", "center"	Anchor à la GridbagLayout
expand	True / False	wx bzw. wy à la GridbagLayout. Dieser Wert sollte nur beim Wert fill="y" oder fill="both" gesetzt werden. Label, Entry, Checkbox, Buttons brauchen kein expand.
fill	"x", "y", "both", "none"	fill-Attribut à la GridbagLayout. Wenn man das Fenster vergrößert, wird das UI-Element mitvergrößert.
padx	int	äußerer Rand in Pixel
pady	int	äußerer Rand in Pixel
side	"left", "right", "top", "bottom"	Layout à la DockPanel von WPF

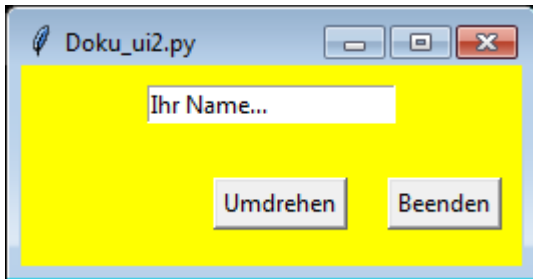
Wenn man in Konstruktor kein „expand“ und „fill“ einbaut, so wird der Layout-Manager nicht gezoomt:

```
def __init__(self, master=None):  
    tkinter.Frame.__init__(self, master)  
    self.pack()  
    self.config(background = "yellow")
```



Mit `expand=True`, `fill="both"` „zoomt“ der Layout-Manager

```
def __init__(self, master=None):  
    tkinter.Frame.__init__(self, master)  
    self.pack(expand=True, fill="both")  
    self.config(background = "yellow")
```



Dieselbe Technik funktioniert nun auch beim Entry-Element:

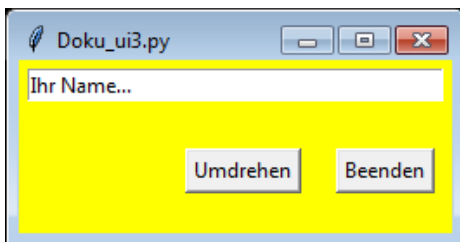
```
inputui = tkinter.Entry(self)  
inputui.pack(padx="30", pady="10")
```

Folge: kein Zoom

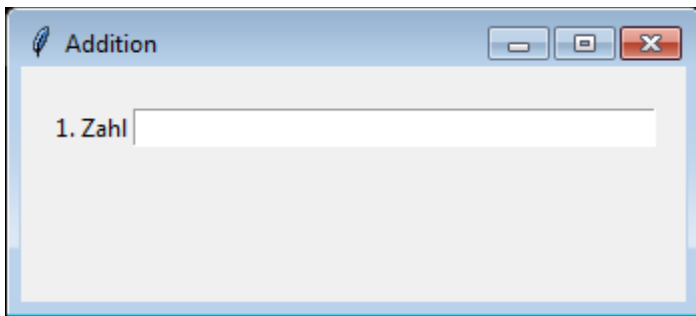
mit `fill="x"`, `padx="5"` geht die Eingabezeile über die gesamte Breite.

```
self.inputui = tkinter.Entry(self)
```

```
self.inputui.pack(fill="x", padx="5", pady="5")
```



Um ein Label mit einer Textzeile einzutragen, ist es sinnvoll diese beiden UI-Elemente mit einem Rahmen, Frame, zu versehen



Quellcode:

```
frame1 = tkinter.Frame(self)      # Rahmen à la JPanel
```

```
frame1.pack(fill="x", side="top", pady="5")
```

```
self.label1 = tkinter.Label(frame1)
```

```
self.label1["text"] = "1. Zahl"
```

```
self.label1.pack(side="left")
```

```
# , justify=tkinter.RIGHT.   Text ist rechtsbündig
```

```
self.inputui1 = tkinter.Entry(frame1, justify=tkinter.RIGHT)
```

```
self.inputui1.pack(expand=True, fill="x", side="left")
```

Bei Label, Entry, Spinbox, Checkbutton benötigt man nur **fill="x"**

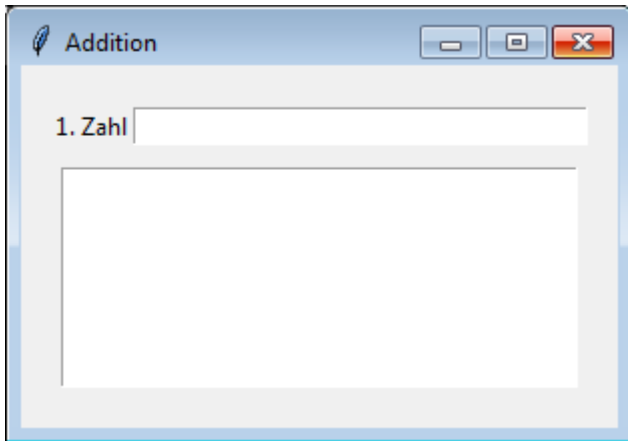
```
frame1.pack(fill="x", side="top", pady="5")
```

Bei Listbox, Text, mehrere Radiobutton sollte man **fill="both"** benutzen.

```
frame1.pack(fill="both", side="top", pady="5")
```

Wenn man nun nach der Eingabezeile einen Editor einfügt

UND danach zwei Button einfügt, so verdrängt der Editor komplett die Schalter.



Fehlerhafte Quellcode:

```
# coding=utf8

import tkinter

from tkinter import messagebox

class MyApp(tkinter.Frame):

    def __init__(self, master=None):

        tkinter.Frame.__init__(self, master)

        self.pack(expand=True, fill="both", padx="15", pady="15")

        self.setGUI()

    def setGUI(self):

        # JPanel für ui-Elemente

        frame1 = tkinter.Frame(self)

        frame1.pack(fill="x", side="top", pady="5")

        self.label1 = tkinter.Label(frame1)

        self.label1["text"] = "1. Zahl"

        self.label1.pack(side="left")
```

```

self.inputu1 = tkinter.Entry(frame1, justify=tkinter.RIGHT)
self.inputu1.pack(expand=True, fill="x",side="left")

self.editor = tkinter.Text(self)
self.editor.pack(expand=True, fill="both", padx="5",pady="5")

buttonframe = tkinter.Frame(self)
buttonframe.pack(fill="x", side="bottom" )      # fill=x,y,both

self.bnEsc = tkinter.Button(buttonframe)
self.bnEsc["text"] = "Ende"
self.bnEsc["command"] = self.quit
self.bnEsc.pack(side="right")

self.bnAction = tkinter.Button(buttonframe)
self.bnAction["text"] = "Action"
self.bnAction.pack(side="right",padx="5")


root = tkinter.Tk()
root.title("Addition")
root.geometry("250x150")
app = MyApp(root)
app.mainloop()

```

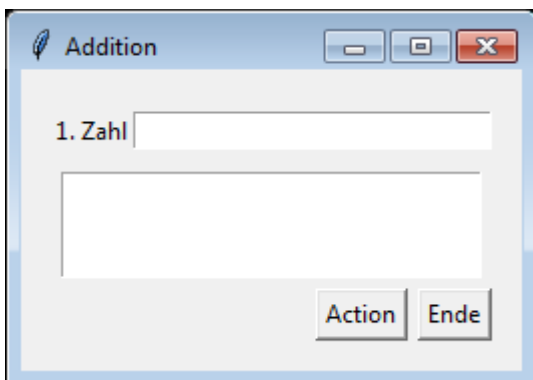
Abhilfe:

Erst alle fill-Elemente mit x oder y eintragen.

Danach erst die UI-Elemente mit fill="both" einfügen.

Das Schema entspricht den Schema eines DockPanels (WPF).

Korrekte Programmierung:



Quellcode:

```
# coding=utf8

import tkinter

from tkinter import messagebox


class MyApp(tkinter.Frame):

    def __init__(self, master=None):

        tkinter.Frame.__init__(self, master)

        self.pack(expand=True, fill="both", padx="15", pady="15")

        self.setGUI()

    def setGUI(self):

        # 1. JPanel für ui-Elemente

        frame1 = tkinter.Frame(self)
```

```
frame1.pack(fill="x", side="top", pady="5")
```

```
self.label1 = tkinter.Label(frame1)
```

```
self.label1["text"] = "1. Zahl"
```

```
self.label1.pack(side="left")
```

```
self.inputu1 = tkinter.Entry(frame1, justify=tkinter.RIGHT)
```

```
self.inputu1.pack(expand=True, fill="x",side="left")
```

2. JPanel für Schalter

```
buttonframe = tkinter.Frame(self)
```

```
buttonframe.pack(fill="x", side="bottom" )      # fill=x,y,both
```

```
self.bnEsc = tkinter.Button(buttonframe)
```

```
self.bnEsc["text"] = "Ende"
```

```
self.bnEsc["command"] = self.quit
```

```
self.bnEsc.pack(side="right")
```

```
self.bnAction = tkinter.Button(buttonframe)
```

```
self.bnAction["text"] = "Action"
```

```
self.bnAction.pack(side="right",padx="5")
```

3. FILL

```
self.editor = tkinter.Text(self)
```

```
self.editor.pack(expand=True, fill="both", padx="5",pady="5")
```

```
root = tkinter.Tk()
```

```
root.title("Addition")
```

```
root.geometry("250x150")
```

```
app = MyApp(root)
```

```
app.mainloop()
```

Variablen für den Zugriff auf ein ui-Element

```
self.inputui = tkinter.Entry(inputframe)
```

```
self. inputui.pack(expand=True,fill="x",padx="5",pady="5")
```

```
self.var_name = tkinter.StringVar()
```

```
self.var_name.set("Ihr Name...")
```

```
self. inputui ["textvariable"] = self.var_name
```

tkinter.BooleanVar()	checkbox_bsp1.py
----------------------	------------------

tkinter.StringVar()	Doku_ui1.py
---------------------	-------------

tkinter.IntVar()	SpinBox1
------------------	----------

tkinter.DoubleVar()	SpinBox2
---------------------	----------