

Bruch - Gui

Vieles noch mit trial and error, aber es funktioniert:

Bruch mit Gui

```
# Bruch-einfachere-Addition.py
class Bruch:
    def __init__(self, zaehler, nenner):
        self.zaehler=zaehler
        self.nenner=nenner

    def zeige(self):
        print "(" , self.zaehler, "/" , self.nenner, ")"

    def gib(self):
        return self.zaehler, self.nenner

    def gibString(self):
        return str(self.zaehler)+ " / " + str(self.nenner)

    def multipliziere(self, b):
        temp=Bruch(self.zaehler*b.zaehler, self.nenner*b.nenner)
        temp.kuerze()
        return temp

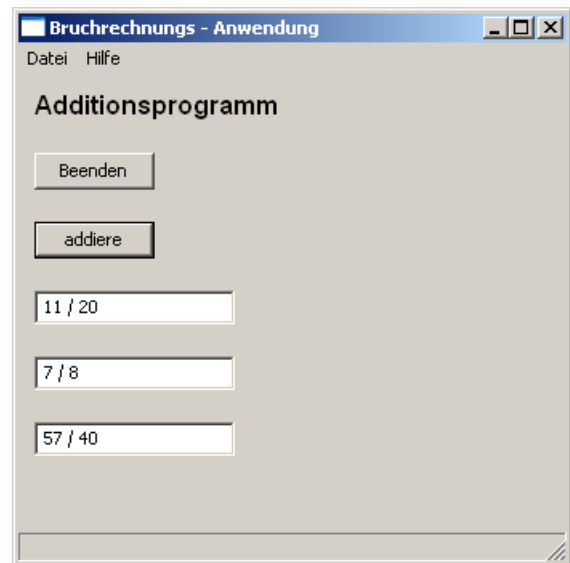
    def dividiere(self, b):
        if b.zaehler!=0:
            temp=Bruch(self.zaehler*b.nenner, self.nenner*b.zaehler)
            temp.kuerze()
        else: temp=self
        return temp

    def kuerze(self):
        # zaehler kann negativ sein!
        teiler=ggT(abs(self.zaehler),self.nenner)
        self.zaehler=self.zaehler/teiler
        self.nenner=self.nenner/teiler

# gleichnamig ist nur ein Problem ohne Langzahlarithmetik
def addiere(self, b):
    temp=Bruch(self.zaehler*b.nenner+b.zaehler*self.nenner,self.nenner*b.nenner)
    temp.kuerze()
    return temp

def subtrahiere(self, b):
    temp=Bruch(self.zaehler*b.nenner-b.zaehler*self.nenner,self.nenner*b.nenner)
    temp.kuerze()
    return temp

# Klassenunabhaengige Funktionen
def ggT(a, b):
    if a<b: a,b=b,a
```



```
rest=a%b
while rest>0:
    a=b
    b=rest
    rest=a%b
return b
```

```
# -----GUI-Abschnitt-----
import wx
```

```
class MyFrame(wx.Frame):
    """
    This is MyFrame. It just shows a few controls on a wxPanel,
    and has a simple menu.
    """
    def __init__(self, parent, title):
        wx.Frame.__init__(self, parent, -1, title,
                           pos=(150, 150), size=(350, 350))

        # Create the menubar
        menuBar = wx.MenuBar()

        # and a menu
        menu = wx.Menu()
        menu2 =wx.Menu()

        # add an item to the menu, using \tKeyName automatically
        # creates an accelerator, the third param is some help text
        # that will show up in the statusbar
        menu.Append(wx.ID_EXIT, "B&eenden\tAlt-X", "Schliessen der Anwendung")

        # bind the menu event to an event handler
        self.Bind(wx.EVT_MENU, self.OnTimeToClose, id=wx.ID_EXIT)

        # and put the menu on the menubar
        menuBar.Append(menu, "&Datei")
        menuBar.Append(menu2, "&Hilfe")
        self.SetMenuBar(menuBar)

        self.CreateStatusBar()

        # Now create the Panel to put the other controls on.
        panel = wx.Panel(self)

        # and a few controls
        text = wx.StaticText(panel, -1, "Additionsprogramm")
        text.SetFont(wx.Font(12, wx.SWISS, wx.NORMAL, wx.BOLD))
        text.SetSize(text.GetBestSize())

        btn = wx.Button(panel, -1, "Beenden")
        btn.SetToolTipString("Button beendet die Anwendung")
```

```
plusbtn = wx.Button(panel, -1, "addiere")
plusbtn.SetToolTipString("addiert die beiden Brueche")

self.textfeld1 = wx.TextCtrl(panel, -1, "11 / 20", size=(125, -1))
self.textfeld2 = wx.TextCtrl(panel, -1, "7 / 8", size=(125, -1))
self.textfeld3 = wx.TextCtrl(panel, -1, " ", size=(125, -1))

# bind the button events to handlers
self.Bind(wx.EVT_BUTTON, self.OnTimeToClose, btn)
self.Bind(wx.EVT_BUTTON, self.OnPlusButton, plusbtn)

# Use a sizer to layout the controls, stacked vertically and with
# a 10 pixel border around each
sizer = wx.BoxSizer(wx.VERTICAL)
sizer.Add(text, 0, wx.ALL, 10)
sizer.Add(btn, 0, wx.ALL, 10)
sizer.Add(plusbtn, 0, wx.ALL, 10)
sizer.Add(self.textfeld1, 0, wx.ALL, 10)
sizer.Add(self.textfeld2, 0, wx.ALL, 10)
sizer.Add(self.textfeld3, 0, wx.ALL, 10)
panel.SetSizer(sizer)
panel.Layout()

def OnTimeToClose(self, evt):
    """Event handler for the button click."""
    print "ENDE!"
    self.Close()

def OnPlusButton(self, evt):
    """Event handler for the button click."""
    z, n=self.holeZahlen(self.textfeld1.GetValue())
    b1=Bruch(z, n)
    z, n=self.holeZahlen(self.textfeld2.GetValue())
    b2=Bruch(z, n)
    b=b1.addiere(b2)
    self.textfeld3.SetValue(b.gibString())

# Hilfsmethode zum Holen von Zaehler und Nenner
def holeZahlen(self, text):
    text_teile=text.split()
    return int(text_teile[0]), int(text_teile[2])

class MyApp(wx.App):
    def OnInit(self):
        frame = MyFrame(None, "Bruchrechnungs - Anwendung")
        self.SetTopWindow(frame)

        print "Ausgaben gehen in das stdout window."
```

```
frame.Show(True)  
return True
```

```
app = MyApp(redirect=True)  
app.MainLoop()
```

Vieles ist noch zu bearbeiten, aber für den Anfang reicht es so.