

## Überarbeitung der Datei grafikfenster.py

Auf Vorschlag eines Kollegen habe ich in `grafikfenster.py` die Möglichkeit eingebaut, in die Zeichenfläche ein Hintergrundbild einzubauen. Die Vorlage stammt aus dem Demoprogramm zu wx-Python (*Projektbeispiel zu GraphicsContext*), allerdings ist dort das Bild „on top“ eingefügt.

Ergänzend habe ich die Möglichkeit eingebaut, für den *dark-modus* auf schwarze Hintergrundfarbe zu wechseln.

Die Änderungen in `grafikfenster.py`:<sup>1</sup>

Im Konstruktor sind nach

```
self.__image=None
```

die Zeilen

```
self.__bgImage=None  
self.__bgColor='WHITE'
```

als Attribute für die beiden Werte mit ihren Standardwerten eingefügt.

Es folgen die zugehörigen Set- und Get-Methoden:

```
def SetBGImage(self, name):  
    '''setzt ein Hintergrundbild'''  
    self.__bgImage=name  
  
def SetBGColor(self, colorName):  
    '''setzt die Hintergrundfarbe'''  
    self.__bgColor=colorName  
  
def GetBGColor(self):  
    '''gibt die Hintergrundfarbe'''  
    return self.__bgColor
```

Wesentlich sind die Änderungen in der Draw-Methode:

Das Zeichnen des Hintergrundbilds muss die erste Teilaktion sein. Dazu muss zunächst der aktuelle Zustand des GraphicsContext – Objekts gesichert werden:

```
gc.PushState() # save current translation/scale/other state
```

Da nicht immer ein Hintergrundbild verwendet werden soll, muss das Zeichnen bedingt ausgeführt werden:

```
if self.__bgImage!=None:  
    # Draw a bitmap with an alpha channel ...  
    bmp = wx.Bitmap(opj(self.__bgImage))  
    bsz = bmp.GetSize()  
    gc.DrawBitmap(bmp,  
        0, 0, ## xPos,yPos  
        bsz.width/2, bsz.height/2) # breite, hoehe  
    gc.PopState()
```

---

<sup>1</sup> Eine Ergänzung stammt aus der Datei Main.py des Demoprogramms, die am Kopf von Grafikfenster als Funktion (`opj`) eingebaut ist, die für das Verständnis aber nicht relevant ist.

Die Halbierung von Größe und Breite ist nur ein Beispiel. Das kann man ggf anpassen.

### **Und sonst?**

Die Alternative ist das Zeichnen eines einfarbigen Hintergrunds, allerdings abhängig von der zu verwendenden Hintergrundfarbe 'WHITE' (*Standard*) oder 'BLACK'.

```
else:
    back = gc.CreatePath()
    back.AddRectangle(0, 0,
                     self.GetClientSize().width,
                     self.GetClientSize().height)
    gc.SetBrush(wx.Brush(self.__bgColor))
    gc.FillPath(back)           # nur Fuellung
    gc.PopState()              # restore saved state
```

Das war's schon in grafikfenster.py.

### **Moebel anpassen**

In der Klasse Moebel sollten die Methoden GibFarbe() und GibFuellFarbe() an den ggf gewählten Dunkelmodus angepasst werden, damit bei schwarzer Randfarbe statt dessen beispielsweise weiss verwendet wird.

### **Hinweis zum Bildexport**

Der Export des dargestellten Bildes aus dem Grafikfenster gelingt nicht, wenn der Aufruf der Methode durch (*beispielsweise*)

```
Zeichenflaeche.GibZeichenflaeche().BildExport(<dateiname>)
```

in der selben Methode erfolgt wie das Zeichnen der Objekte. Der Grund ist, dass wxPython die Darstellung von Bildern dadurch „*optimiert*“, dass sie erst nach Ende der Methode mit den Zeichenanweisungen ausgeführt wird<sup>1</sup>. Der oben angegebene Aufruf muss also in einer anderen Methode erfolgen.

---

<sup>1</sup> Das hat auch Folgen bei der Darstellung von Bewegungen oder anderen Änderungen im Bild durch schrittweise Ausführung: Immer erst die Zeichenmethode beenden, damit der Zustand dargestellt wird, bevor die Anweisungen für den nächsten Schritt kommen.