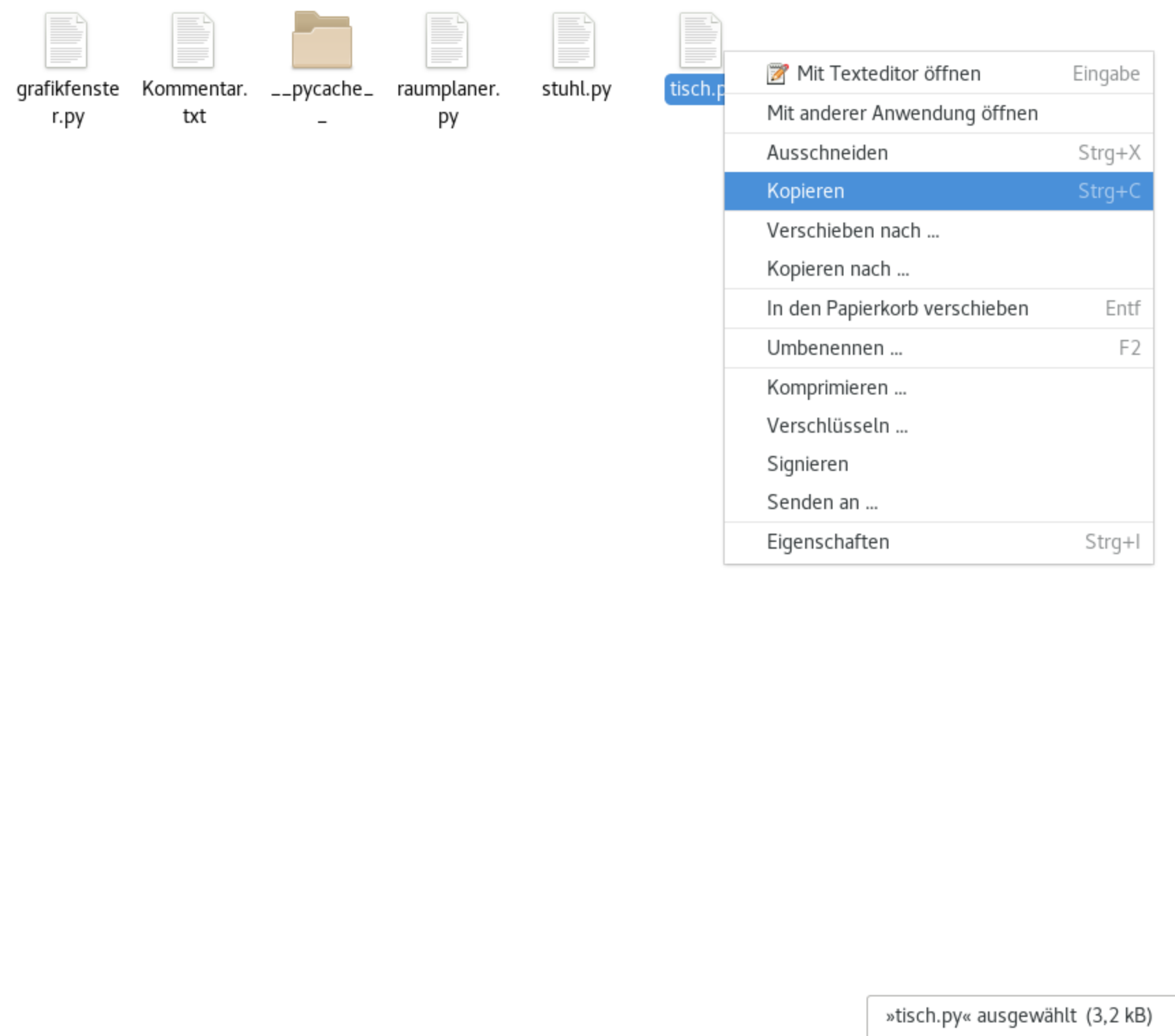
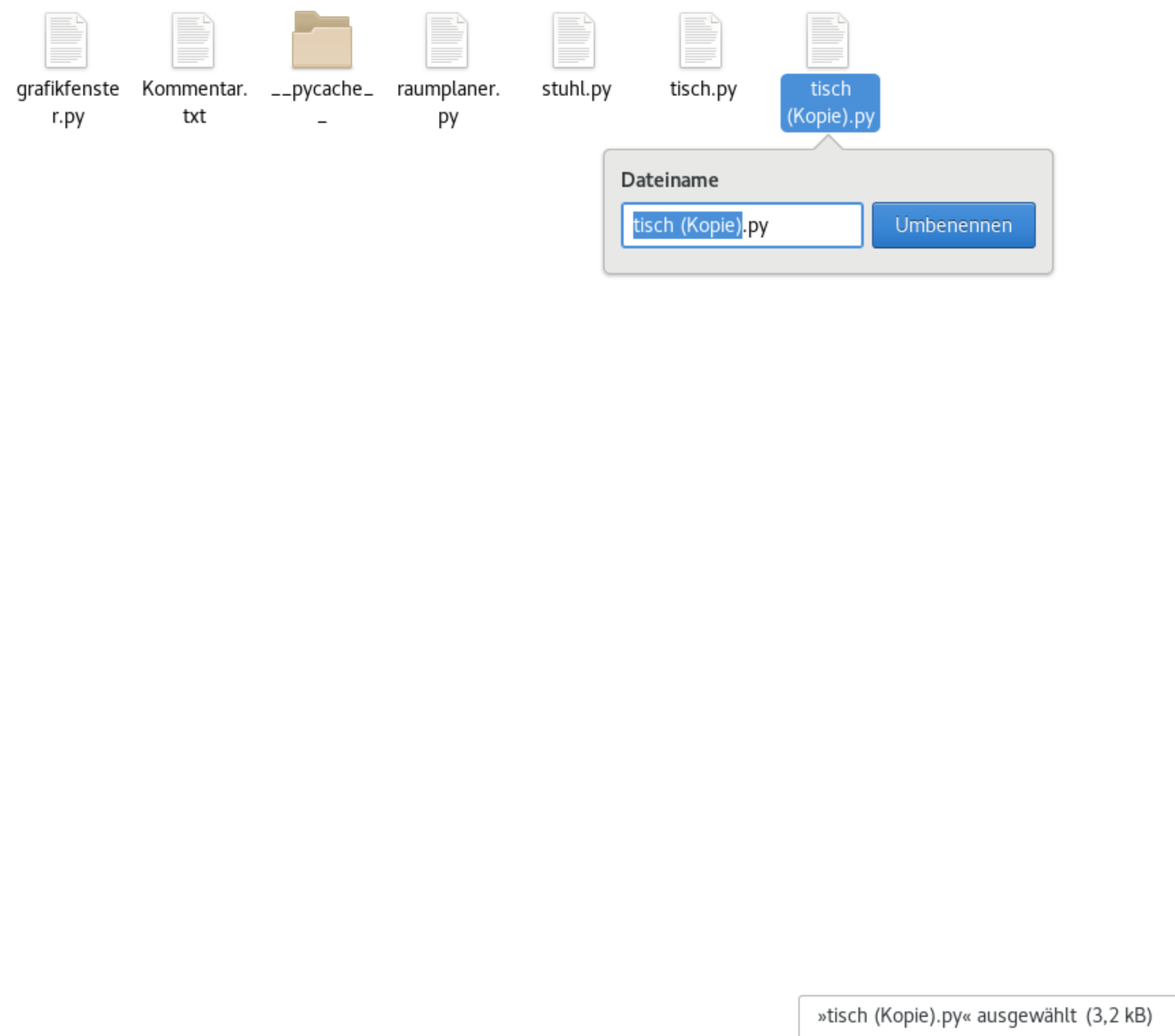

grafikfenster.py
Kommentar.txt
__pycache__
raumplaner.py
stuhl.py
tisch.py


grafikfenster.py
Kommentar.txt
__pycache__
raumplaner.py
stuhl.py
tisch.p
Mit Texteditor öffnen Eingabe
Mit anderer Anwendung öffnen
Ausschneiden Strg+X
Kopieren Strg+C
Verschieben nach ...
Kopieren nach ...
In den Papierkorb verschieben Entf
Umbenennen ... F2
Komprimieren ...
Verschlüsseln ...
Signieren
Senden an ...
Eigenschaften Strg+I
»tisch.py« ausgewählt (3,2 kB)


grafikfenster.py
Kommentar.txt
__pycache__
raumplaner.py
stuhl.py
tisch.py
tisch (Kopie).py


grafikfenster.py
Kommentar.txt
__pycache__
raumplaner.py
stuhl.py
tisch.py
tisch (Kopie).py
Dateiname
tisch (Kopie).py
Umbenennen
»tisch (Kopie).py« ausgewählt (3,2 kB)


grafikfenster.py
Kommentar.txt
moebel.py
__pycache__
raumplaner.py
stuhl.py
tisch.py

grafikfenste
r.py

Kom

moebel.py - /home/nutzer/Dokumente/LI/2021-LI-OO/Projekte/02/Moebel-erarbeiten-aus-...

File Edit Format Run Options Window Help

```
# tisch.py

from grafikfenster import *
from math import radians

### ------------------------------------------------
class Tisch():
    """Klasse Tisch
    ermoeglicht das Zeichnen und Bearbeiten eines
    Tisch-Symbols fuer den Raumplaner"""

    def __init__(self, sichtbar=False):
        """einfacher Konstruktor"""
        self.x=70
        self.y=10
        self.b=120
        self.t=60
        self.w=0
        self.f='red'
        self.s=sichtbar
        if sichtbar: self.Zeige()

    def GibFigur(self):
        """definiert und transformiert die zu zeichnende Figur"""
        gc = Zeichenflaeche.GibZeichenflaeche().GibGC()
        path = gc.CreatePath()

        path.AddRectangle(0, 0, self.b, self.t)

        gc.PushState()
        gc.Translate(self.x+self.b/2, self.y+self.t/2)
        gc.Rotate(radians(self.w))
        gc.Translate(-self.b/2, -self.t/2)
        transformation = gc.GetTransform()
        gc.PopState()
```

Ln: 1 Col: 0

grafikfenste
r.py

Kom

*moebel.py - /home/nutzer/Dokumente/LI/2021-LI-OO/Projekte/02/Moebel-erarbeiten-aus-...

File Edit Format Run Options Window Help

```
# moebel.py

from grafikfenster import *
from math import radians

### ------------------------------------------------
class Moebel():
    """Klasse Moebel
    Oberklasse der Moebel-Symbole fuer den Raumplaner"""

    def __init__(self, sichtbar=False):
        """einfacher Konstruktor"""
        self.x=70
        self.y=10
        self.b=120
        self.t=60
        self.w=0
        self.f='red'
        self.s=sichtbar
        if sichtbar: self.Zeige()

    def GibFigur(self):
        """definiert und transformiert die zu zeichnende Figur"""
        gc = Zeichenflaeche.GibZeichenflaeche().GibGC()
        path = gc.CreatePath()

        path.AddRectangle(0, 0, self.b, self.t)

        gc.PushState()
        gc.Translate(self.x+self.b/2, self.y+self.t/2)
        gc.Rotate(radians(self.w))
        gc.Translate(-self.b/2, -self.t/2)
        transformation = gc.GetTransform()
        gc.PopState()
        path.Transform(transformation)
```

Ln: 10 Col: 0

grafikfenste
r.py

Kom

*moebel.py - /home/nutzer/Dokumente/LI/2021-LI-OO/Projekte/02/Moebel-erarbeiten-aus-...

File Edit Format Run Options Window Help

```
    def Verberge(self):
        """Veraendernde Methode fuer die Sichtbarkeit mit Wert False"""
        self.s = False
        Zeichenflaeche.GibZeichenflaeche().Entferne(self)

    def Zeige(self):
        """Veraendernde Methode fuer die Sichtbarkeit mit Wert True"""
        self.s = True
        Zeichenflaeche.GibZeichenflaeche().Zeichne(self)

### ------------------------------------------------
class MoebelApp(wx.App):
    """Test-Anwendung speziell fuer Moebel"""
    def OnInit(self):
        self.fenster = GrafikFenster(None, "Raumplaner-Grafik")
        self.SetTopWindow(self.fenster)
        self.fenster.Show(True)
        self.fenster.panel.Refresh()
        self.fenster.ZeigeShellFrame()
        #self.fenster.ZeigeFillingFrame()
        self.TestAnwendung()
        return True

    def TestAnwendung(self):
        """Allein fuer die Testanwendung:"""
        global moebel
        moebel=Moebel(True)

### ------------------------------------------------
if __name__ == '__main__':
    app = MoebelApp(redirect=False)
    # Parameterwert True, wenn Ausgaben in der Standard E/A angezeigt werden sol
    app.MainLoop()
```

Ln: 84 Col: 0


*tisch.py - /home/nutzer/Dokumente/LI/2021-LI-OO/Projekte/02/Moebel-erarbeiten-aus-An... ×

File Edit Format Run Options Window Help

```
# tisch.py

from grafikfenster import *
from math import radians

from moebel import Moebel

### ------------------------------------------------
class Tisch(Moebel):
    """Klasse Tisch
    ermoeglicht das Zeichnen und Bearbeiten eines
    Tisch-Symbols fuer den Raumplaner"""

    def __init__(self, sichtbar=False):
        """einfacher Konstruktor"""
        self.x=70
        self.y=10
        self.b=120
        self.t=60
        self.w=0
        self.f='red'
        self.s=sichtbar
        if sichtbar: self.Zeige()

    def GibFigur(self):
        """definiert und transformiert die zu zeichnende Figur"""
        gc = Zeichenflaeche.GibZeichenflaeche().GibGC()
        path = gc.CreatePath()

        path.AddRectangle(0, 0, self.b, self.t)

        gc.PushState()
        gc.Translate(self.x+self.b/2, self.y+self.t/2)
        gc.Rotate(radians(self.w))
```

Ln: 13 Col: 0


*tisch.py - /home/nutzer/Dokumente/LI/2021-LI-OO/Projekte/02/Moebel-erarbeiten-aus-An... ×

File Edit Format Run Options Window Help

```
        path.Transform(transformation)
        return path

##    def GibFarbe(self):
##        """Get-Methode fuer die Farbe"""
##        return self.f
##
##    def GibSichtbar(self):
##        """Get-Methode fuer die Sichtbarkeit"""
##        return self.s
##
##    def BewegeHorizontal(self, weite):
##        """Veraendernde Methode fuer die x-Position"""
##        self.Verberge()
##        self.x += weite
##        self.Zeige()
##
##    def Bewege(self, xWeite, yWeite):
##        """Veraendernde Methode fuer die x-Position und
##        die y-Position"""
##        self.Verberge()
##        self.x += weite
##        self.Zeige()
##
##    def BewegeVertikal(self, weite):
##        """Veraendernde Methode fuer die y-Position"""
##        self.Verberge()
##        self.y += weite
##        self.Zeige()
##
##    def Drehe(self, winkel):
##        """Veraendernde Methode fuer die Orientierung [Winkel]"""
##        self.Verberge()
##        self.w += winkel
```

Ln: 39 Col: 19

»tisch.py« ausgewählt (3,3 kB)


tisch.py - /home/nutzer/Dokumente/LI/2021-LI-OO/Projekte/02/Moebel-erarbeiten-aus-Anf...

File Edit Format Run Options Window Help

```
        path.Transform(transformation)
        return path

##    def GibFarbe(self):
##        """Get-Methode fuer die Farbe"""
##        return self.f
##
##    def GibSic
##        """Get
##        return
##
##    def Bewege
##        """Ver
##        self.V
##        self.x
##        self.Z
##
##    def Bewege
##        """Ver
##        die y-
##        self.V
##        self.x
##        self.Z
##
##    def Bewege
##        """Ver
##        self.V
##        self.y
##        self.Zeige()
##
##    def Drehe(self, winkel):
##        """Veraendernde Methode fuer die Orientierung [Winkel]"""
##        self.Verberge()
##        self.w += winkel
```

Raumplaner-Grafik

Ln: 41 Col: 14

»tisch.py« ausgewählt (3,3 kB)

grafikfenst
r.py

stuhl.py - /home/nutzer/Dokumente/LI/2021-LI-OO/Projekte/02/Moebel-erarbeiten-aus-Anf...

File Edit Format Run Options Window Help

```
from grafikfenster import *
from math import radians

from moebel import Moebel

### ------------------------------------------------
class Stuhl(Moebel):
    """Klasse Stuhl
    ermoeglicht das Zeichnen und Bearbeiten eines
    Stuhl-Symbols fuer den Raumplaner"""

    def __init__(self, sichtbar=False):
        """einfacher Konstruktor"""
        self.x=20
        self.y=20
        self.b=40
        self.t=40
        self.w=270
        self.f="blue"
        self.s=sichtbar
        if sichtbar: self.Zeige()

    def GibFigur(self):
        """definiert und transformiert die zu zeichnende Figur"""
        gc = Zeichenflaeche.GibZeichenflaeche().GibGC()
        path = gc.CreatePath()

        path.MoveToPoint(0, 0)
        path.AddLineToPoint(self.b, 0)
        path.AddLineToPoint(self.b*1.1, self.t)
        path.AddLineToPoint(-self.b*0.1, self.t)
        path.AddLineToPoint(0, 0)
        path.AddLineToPoint(0, -self.t*0.1)
```

Ln: 12 Col: 40

»stuhl.py« ausgewählt (3,4 kB)

grafikfenst
r.py

stuhl.py - /home/nutzer/Dokumente/LI/2021-LI-OO/Projekte/02/Moebel-erarbeiten-aus-Anf...

File Edit Format Run Options Window Help

```
from grafikfenster import *
from math import radians

from moebel import Moebel

### ------------------------------------------------
class Stuhl(Moebel):
    """Klasse Stuhl
    ermoeglicht das Ze
    Stuhl-Symbols fuer

    def __init__(self,
        """einfacher K
        self.x=20
        self.y=20
        self.b=40
        self.t=40
        self.w=270
        self.f="blue"
        self.s=sichtba
        if sichtbar: s

    def GibFigur(self)
        """definiert u
        gc = Zeichenfl
        path = gc.Crea

        path.MoveToPoi
        path.AddLineToPoint(self.b, 0)
        path.AddLineToPoint(self.b*1.1, self.t)
        path.AddLineToPoint(-self.b*0.1, self.t)
        path.AddLineToPoint(0, 0)
```

Raumplaner-Grafik

Ln: 12 Col: 40

»stuhl.py« ausgewählt (3,4 kB)

grafikfenst
r.py

stuhl.py - /home/nutzer/Dokumente/LI/2021-LI-OO/Projekte/02/Moebel-erarbeiten-aus-Anf...

File Edit Format Run Options Window Help

```
from grafikfenster import *
from math import radians

from moebel import Moebel

### ------------------------------------------------
class Stuhl(Moebel):
    """Klasse Stuhl
    ermoeglicht das Ze
    Stuhl-Symbols fuer

    def __init__(self,
        """einfacher K
        self.x=20
        self.y=20
        self.b=40
        self.t=40
        self.w=270
        self.f="blue"
        self.s=sichtba
        if sichtbar: s

    def GibFigur(self)
        """definiert u
        gc = Zeichenfl
        path = gc.Crea

        path.MoveToPoi
        path.AddLineToPoint(self.b, 0)
        path.AddLineToPoint(self.b*1.1, self.t)
        path.AddLineToPoint(-self.b*0.1, self.t)
        path.AddLineToPoint(0, 0)
```

Raumplaner-Grafik

Ln: 12 Col: 40

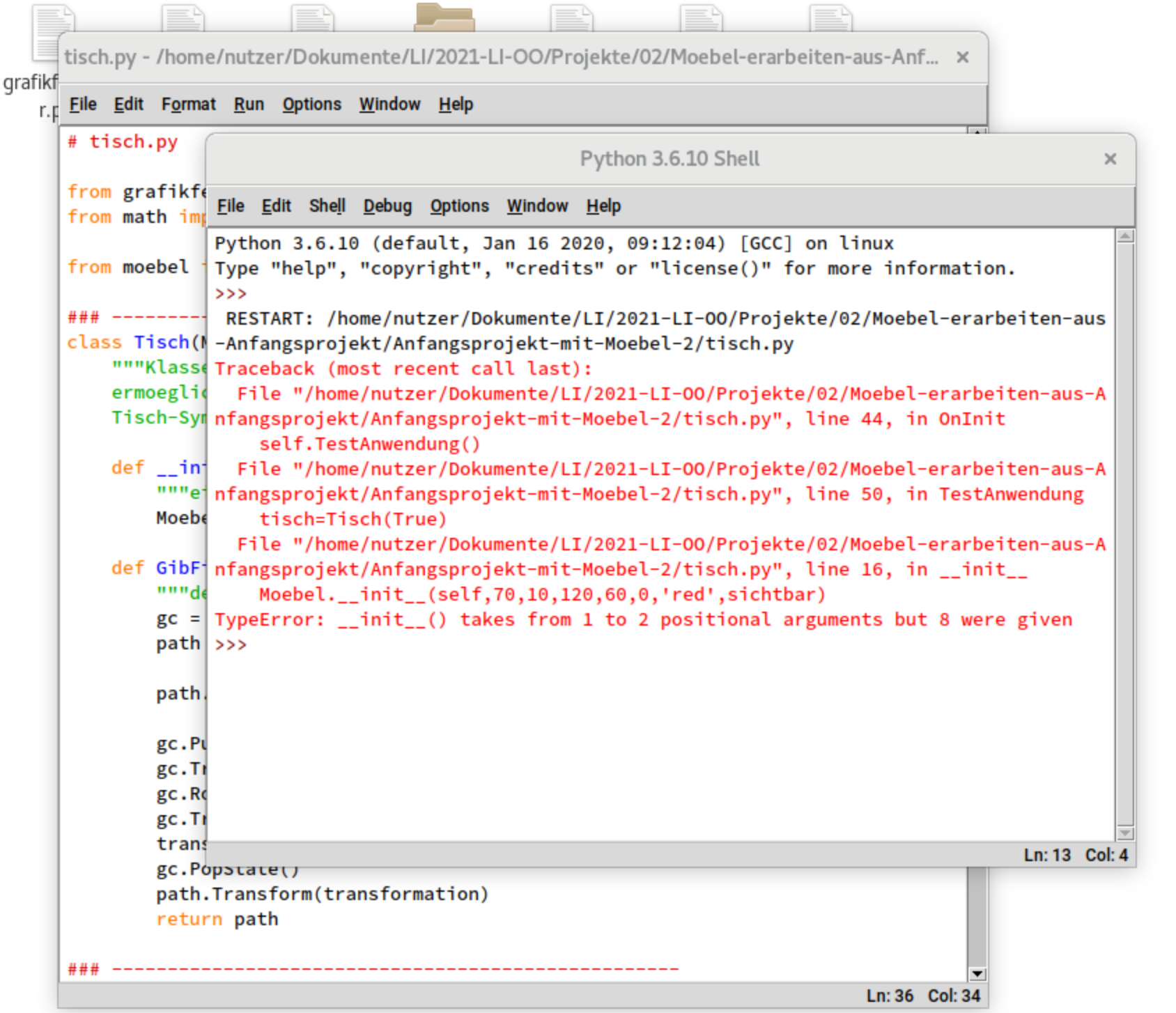
»stuhl.py« ausgewählt (3,4 kB)

tisch.py - /home/nutzer/Dokumente/LI/2021-LI-OO/Projekte/02/Moebel-erarbeiten-aus-Anf...

File Edit Format Run Options Window Help

```python
# tisch.py

from grafikfenster import *
from math import radians

from moebel import Moebel

### ------------------------------------------------
class Tisch(Moebel):
    """Klasse Tisch
    ermoeglicht das Zeichnen und Bearbeiten eines
    Tisch-Symbols fuer den Raumplaner"""

    def __init__(self, sichtbar=False):
        """einfacher Konstruktor"""
        Moebel.__init__(self,70,10,120,60,0,'red',sichtbar)

    def GibFigur(self):
        """definiert und transformiert die zu zeichnende Figur"""
        gc = Zeichenflaeche.GibZeichenflaeche().GibGC()
        path = gc.CreatePath()

        path.AddRectangle(0, 0, self.b, self.t)

        gc.PushState()
        gc.Translate(self.x+self.b/2, self.y+self.t/2)
        gc.Rotate(radians(self.w))
        gc.Translate(-self.b/2, -self.t/2)
        transformation = gc.GetTransform()
        gc.PopState()
        path.Transform(transformation)
        return path

### ------------------------------------------------
```

Ln: 36 Col: 34


tisch.py - /home/nutzer/Dokumente/LI/2021-LI-OO/Projekte/02/Moebel-erarbeiten-aus-Anf...
File Edit Format Run Options Window Help
# tisch.py
from grafikfe
from math imp
from moebel
### --------
class Tisch(M
"""Klasse
ermoeglic
Tisch-Sym
def __in
"""e
Moebe
def GibF
"""de
gc =
path
path.
gc.Pu
gc.Tr
gc.Ro
gc.Tr
trans
gc.PopState()
path.Transform(transformation)
return path
### ----------
Ln: 36 Col: 34
Python 3.6.10 Shell
File Edit Shell Debug Options Window Help
Python 3.6.10 (default, Jan 16 2020, 09:12:04) [GCC] on linux
Type "help", "copyright", "credits" or "license()" for more information.
>>>
RESTART: /home/nutzer/Dokumente/LI/2021-LI-OO/Projekte/02/Moebel-erarbeiten-aus
-Anfangsprojekt/Anfangsprojekt-mit-Moebel-2/tisch.py
Traceback (most recent call last):
File "/home/nutzer/Dokumente/LI/2021-LI-OO/Projekte/02/Moebel-erarbeiten-aus-A
nfangsprojekt/Anfangsprojekt-mit-Moebel-2/tisch.py", line 44, in OnInit
self.TestAnwendung()
File "/home/nutzer/Dokumente/LI/2021-LI-OO/Projekte/02/Moebel-erarbeiten-aus-A
nfangsprojekt/Anfangsprojekt-mit-Moebel-2/tisch.py", line 50, in TestAnwendung
tisch=Tisch(True)
File "/home/nutzer/Dokumente/LI/2021-LI-OO/Projekte/02/Moebel-erarbeiten-aus-A
nfangsprojekt/Anfangsprojekt-mit-Moebel-2/tisch.py", line 16, in __init__
Moebel.__init__(self,70,10,120,60,0,'red',sichtbar)
TypeError: __init__() takes from 1 to 2 positional arguments but 8 were given
>>>
Ln: 13 Col: 4

grafikfenster.py

moebel.py - /home/nutzer/Dokumente/LI/2021-LI-OO/Projekte/02/Moebel-erarbeiten-aus-...

File Edit Format Run Options Window Help

```
# moebel.py

from grafikfenster import *
from math import radians

### ------------------------------------------------
class Moebel():
    """Klasse Moebel
    Oberklasse der Moebel-Symbole fuer den Raumplaner"""

    def __init__(self, x, y, b, t, w, f, sichtbar):
        """einfacher Konstruktor"""
        self.x=x
        self.y=y
        self.b=b
        self.t=t
        self.w=w
        self.f=f
        self.s=sichtbar
        if sichtbar: self.Zeige()

    def GibFigur(self):
        """definiert und transformiert die zu zeichnende Figur"""
        gc = Zeichenflaeche.GibZeichenflaeche().GibGC()
        path = gc.CreatePath()

        path.AddRectangle(0, 0, self.b, self.t)

        gc.PushState()
        gc.Translate(self.x+self.b/2, self.y+self.t/2)
        gc.Rotate(radians(self.w))
        gc.Translate(-self.b/2, -self.t/2)
        transformation = gc.GetTransform()
        gc.PopState()
```

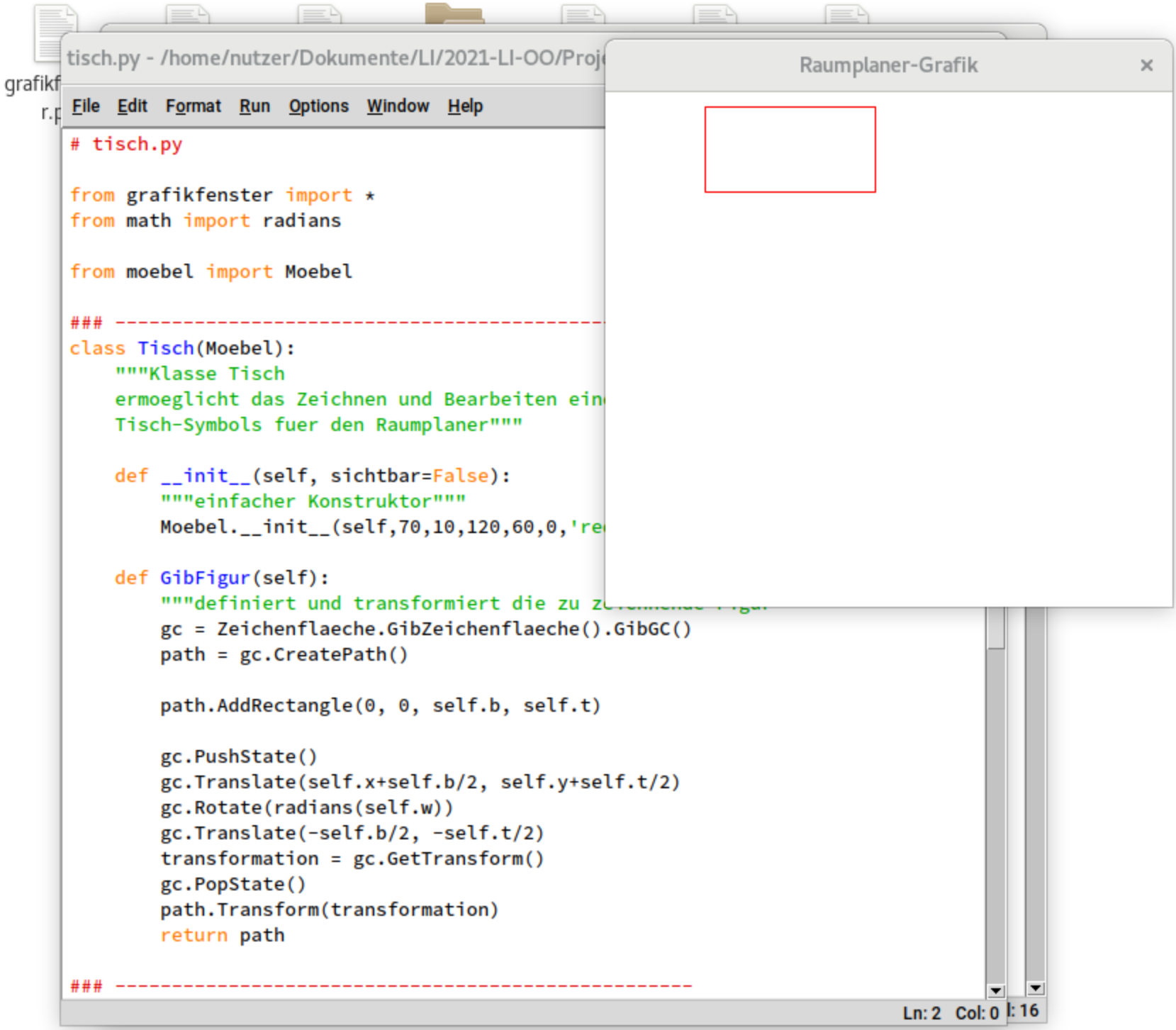
Ln: 18 Col: 16

tisch.py - /home/nutzer/Dokumente/LI/2021-LI-OO/Projekte/02/Moebel-erarbeiten-aus-Anf...

File Edit Format Run Options Window Help

```python
# tisch.py

from grafikfenster import *
from math import radians

from moebel import Moebel

### ------------------------------------------------
class Tisch(Moebel):
    """Klasse Tisch
    ermoeglicht das Zeichnen und Bearbeiten eines
    Tisch-Symbols fuer den Raumplaner"""

    def __init__(self, sichtbar=False):
        """einfacher Konstruktor"""
        Moebel.__init__(self,70,10,120,60,0,'red',sichtbar)

    def GibFigur(self):
        """definiert und transformiert die zu zeichnende Figur"""
        gc = Zeichenflaeche.GibZeichenflaeche().GibGC()
        path = gc.CreatePath()

        path.AddRectangle(0, 0, self.b, self.t)

        gc.PushState()
        gc.Translate(self.x+self.b/2, self.y+self.t/2)
        gc.Rotate(radians(self.w))
        gc.Translate(-self.b/2, -self.t/2)
        transformation = gc.GetTransform()
        gc.PopState()
        path.Transform(transformation)
        return path

### ------------------------------------------------
```

Ln: 36 Col: 34


tisch.py - /home/nutzer/Dokumente/LI/2021-LI-OO/Proj
File Edit Format Run Options Window Help
# tisch.py
from grafikfenster import *
from math import radians
from moebel import Moebel
### ----------
class Tisch(Moebel):
"""Klasse Tisch
ermoeglicht das Zeichnen und Bearbeiten ein
Tisch-Symbols fuer den Raumplaner"""
def __init__(self, sichtbar=False):
"""einfacher Konstruktor"""
Moebel.__init__(self,70,10,120,60,0,'re
def GibFigur(self):
"""definiert und transformiert die zu z
gc = Zeichenflaeche.GibZeichenflaeche().GibGC()
path = gc.CreatePath()
path.AddRectangle(0, 0, self.b, self.t)
gc.PushState()
gc.Translate(self.x+self.b/2, self.y+self.t/2)
gc.Rotate(radians(self.w))
gc.Translate(-self.b/2, -self.t/2)
transformation = gc.GetTransform()
gc.PopState()
path.Transform(transformation)
return path
### ----------
Ln: 2 Col: 0
Raumplaner-Grafik