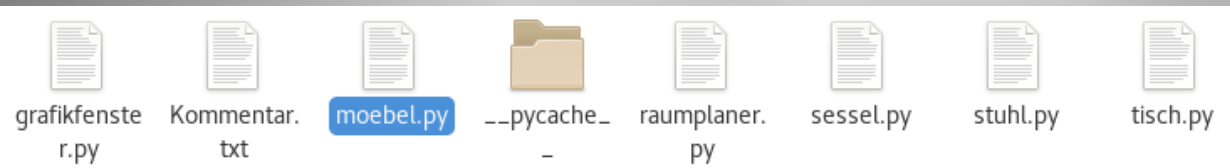
Vererbung Projekt

Bilder zur
einfachen Lösung

Vererbung Projekt

- Klasse Moebel

  Konstruktor



moebel.py - /home/nutzer/Dokumente/LI/2021-LI-OO/Projekte/02/Raumplaner-Vererbung ohn...

File Edit Format Run Options Window Help

```
# -*- coding: utf-8 -*-
# moebel.py
# Version ohne Kapselung und nicht abstrakt

from grafikfenster import *

### ------------------------------------------------
class Moebel():
    """Klasse Moebel
    Oberklasse für das Zeichnen und Bearbeiten von
    Moebel-Symbolen fuer den Raumplaner"""

    def __init__(self,
                 xPos,
                 yPos,
                 breite,
                 tiefe,
                 winkel,
                 farbe,
                 sichtbar):
        """Konstruktor mit vordefinierten Parametern
        ist bei Moebel nicht sinnvoll"""
        self.x=xPos
        self.y=yPos
        self.b=breite
        self.t=tiefe
        self.w=winkel
        self.f=farbe
        self.s=sichtbar
        if sichtbar: self.Zeige()

## Austesten: Methode GibFigur auskommentieren!
    def GibFigur(self):
        """Methode gibt leeren Path zurueck"""
        gc = Zeichenflaeche.GibZeichenflaeche().GibGC()
        path = gc.CreatePath()
        return path

    def GibSichtbar(self):
        """Get-Methode fuer die Sichtbarkeit"""
```

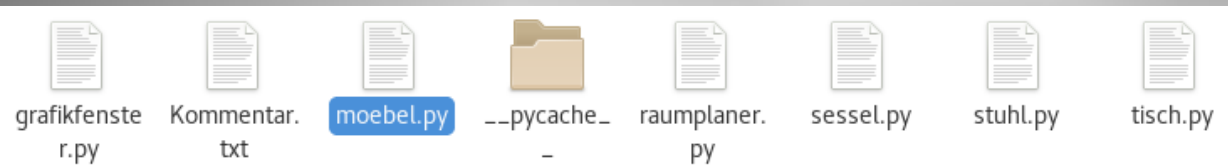
Ln: 1 Col: 0

»moebel.py« ausgewählt (3,2 kB)

Vererbung Projekt

- Klasse Moebel

  Methoden



moebel.py - /home/nutzer/Dokumente/LI/2021-LI-OO/Projekte/02/Raumplaner-Vererbung ohn...

File Edit Format Run Options Window Help

```
    def GibFarbe(self):
        """Get-Methode fuer die Farbe"""
        return self.f


    def BewegeHorizontal(self, weite):
        """Veraendernde Methode fuer die x-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
        self.Zeige()

    def AendereFarbe(self, neueFarbe):
        """Veraendernde Methode fuer die Farbe"""
        sichtbar=self.GibSichtbar()
        if sichtbar: self.Verberge()
        self.f = neueFarbe
        if sichtbar: self.Zeige()

    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)
```

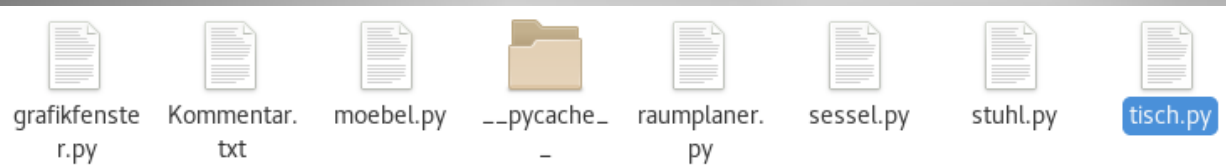
Ln: 1 Col: 0

»moebel.py« ausgewählt (3,2 kB)

Vererbung Projekt

- Klasse Tisch

  [Konstruktor von Moebel aufrufen!]



tisch.py - /home/nutzer/Dokumente/LI/2021-LI-OO/Projekte/02/Raumplaner-Vererbung ohne ...

File Edit Format Run Options Window Help

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

    def __init__(self,
                 xPos=60,
                 yPos=10,
                 breite=120,
                 tiefe=60,
                 winkel=0,
                 farbe='red',
                 sichtbar=False):
        """Konstruktor mit vordefinierten Parametern
        Konstruktor der Oberklasse aufrufen!"""
        Moebel.__init__(self, xPos, yPos, breite, tiefe, winkel, farbe, sichtbar)
        if sichtbar: self.Zeige()

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

        # Hilfsvariable zur Vereinfachung
        b,t = self.b, self.t
        path.AddRectangle(0, 0, b, t)

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

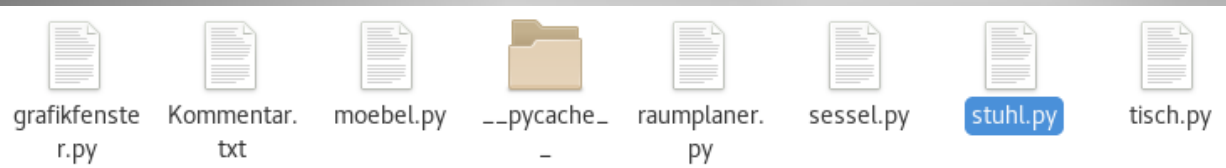
Ln: 1 Col: 0

»tisch.py« ausgewählt (2,2 kB)

Vererbung Projekt

- Klasse Stuhl

  [Konstruktor von Moebel aufrufen!]



stuhl.py - /home/nutzer/Dokumente/LI/2021-LI-OO/Projekte/02/Raumplaner-Vererbung ohne ...

File Edit Format Run Options Window Help

```
    Stuhl-Symbols fuer den Raumplaner"""

    def __init__(self,
                 xPos=20,
                 yPos=20,
                 breite=40,
                 tiefe=40,
                 winkel=0,
                 farbe="blue",
                 sichtbar=False):
        """Konstruktor mit vordefinierten Parametern
        Konstruktor der Oberklasse aufrufen!"""
        Moebel.__init__(self, xPos, yPos, breite, tiefe, winkel, farbe, sichtbar)
        if sichtbar: self.Zeige()

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

        # Hilfsvariable zur Vereinfachung
        b,t = self.b, self.t
        path.MoveToPoint(0, 0)
        path.AddLineToPoint(b, 0)
        path.AddLineToPoint(b*1.1, t)
        path.AddLineToPoint(-b*0.1, t)
        path.AddLineToPoint(0, 0)
        path.AddLineToPoint(0, -t*0.1)
        path.AddLineToPoint(b, -t*0.1)
        path.AddLineToPoint(b, 0)

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

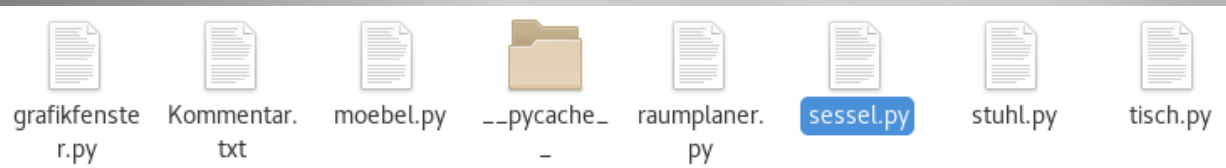
Ln: 1 Col: 0

»stuhl.py« ausgewählt (2,5 kB)

Vererbung Projekt

- Klasse Sessel

  [Konstruktor von Moebel aufrufen!]



sessel.py - /home/nutzer/Dokumente/LI/2021-LI-OO/Projekte/02/Raumplaner-Vererbung ohne...

File Edit Format Run Options Window Help

```
    """Klasse Sessel
    ermoeglicht das Zeichnen und Bearbeiten eines
    Sessel-Symbols fuer den Raumplaner"""

    def __init__(self,
                 xPos=50,
                 yPos=150,
                 breite=60,
                 tiefe=60,
                 winkel=0,
                 farbe="gray",
                 sichtbar=False):
        """Konstruktor mit vordefinierten Parametern"""
        Moebel.__init__(self,xPos,yPos,breite,tiefe,winkel,farbe,sichtbar)

    def GibFigur(self):
        """Definiert den Pfad und transformiert ihn."""
        path = Zeichenflaeche.GibZeichenflaeche().GibGC().CreatePath()
        # Umrandung
        path.MoveToPoint(0, 0)
        path.AddLineToPoint(self.b, 0)
        path.AddLineToPoint(self.b, self.t)
        path.AddLineToPoint(0, self.t)
        path.AddLineToPoint(0, 0)
        # linke Lehne
        path.MoveToPoint(self.b/6, 0)
        path.AddLineToPoint(self.b/6, self.t)
        # rechte Lehne
        path.MoveToPoint(5*self.b/6, 0)
        path.AddLineToPoint(5*self.b/6, self.t)
        # Rueck-Lehne
        path.MoveToPoint(self.b/6, self.t/6)
        path.AddLineToPoint(5*self.b/6, self.t/6)

        gc = Zeichenflaeche.GibZeichenflaeche().GibGC()
        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()
```

Ln: 1 Col: 0

»sessel.py« ausgewählt (2,7 kB)