Vererbung

Im Raumplaner-Projekt

Vererbung einsetzen

(kurze Version ohne UML)

Vererbung

Schritt 1

- Klären Sie, welche Attribute und Methoden in den Klassen *`stuhl`* und *`tisch`* (und …) identisch sind und daher nach *`moebel`* verschoben werden können.
- Im folgenden Bild sind die wesentlichen Unterschiede hervorgehoben.

stuhl.py - /home/nutzer/Dokumente/LI/2021-LI-OO/Projekte/01 Loes...

File Edit Format Run Options Window Help

```
### ------------------------------------------------
class Stuhl():
    """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)
        path.AddLineToPoint(self.__b, -self.__t*0.1)
        path.AddLineToPoint(self.__b, 0)

        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

    def GibX(self):
        """Get-Methode fuer die x-Position"""
        return self.__x

    def GibY(self):
        """Get-Methode fuer die y-Position"""
        return self.__y
```

Ln: 1 Col: 0

tisch.py - /home/nutzer/Dokumente/LI/2021-LI-OO/Projekte/01 Loe...

File Edit Format Run Options Window Help

```
# -*- coding: utf-8 -*-
# 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()
        path.Transform(transformation)
        return path

    def GibX(self):
        """Get-Methode fuer die x-Position"""
        return self.__x

    def GibY(self):
        """Get-Methode fuer die y-Position"""
        return self.__y
```

Ln: 1 Col: 0

Vererbung

Schritt 2

- Im Dateibrowser mit copy-and-paste beispielsweise die Datei *`tisch.py`* kopieren und zu *`moebel.py`* umbenennen.
- Im Text alle auftretenden *`Tisch`* durch *`Moebel`* ersetzen
- Im Text alle auftretenden *`tisch`* durch *`moebel`* ersetzen

Vererbung

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

File Edit Format Run Options Window Help

```
# -*- coding: utf-8 -*-
# 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):
        """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()
        path.Transform(transformation)
        return path

    def GibX(self):
        """Get-Methode fuer die x-Position"""
```

Ln: 1 Col: 0

Vererbung

Schritt 3

- Im Text von allen anderen Möbelklassen alles Gemeinsame löschen
- Wir bleiben hier bei Tisch und gehen "vorsichtig" vor, indem wir zunächst diese gemeinsamen Abschnitte auskommentieren

File Edit Format Run Options Window Help

```
### ------------------------------------------------
class Tisch():
    def __init__(self,
                 xPos=60,
                 yPos=10,
                 breite=120,
                 tiefe=60,
                 winkel=0,
                 farbe="black",
                 sichtbar=False):
##        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()

    def GibFigur(self):
        """Definiert die Figur <path>"""
        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

##    def GibX(self):
##        """Get-Methode fuer die x-Position"""
##        return self.__x
##
##    def GibY(self):
##        """Get-Methode fuer die y-Position"""
##        return self.__y
##
##    def GibBreite(self):
##        """Get-Methode fuer die Breite"""
```

Ln: 70 Col: 12

rbung

Vererbung

Schritt 4

- Ein Starten des Programms schlägt natürlich fehl, da ihm wichtige Teile fehlen.
  Der erste gefundene Fehler ist der fehlende Inhalt des Konstruktors.
    - Wir ...

Vererbung

Schritt 4

- ...
  - Wir importieren Moebel,
  - teilen dem Programm mit, dass es von der Klasse Moebel abgeleitet wird und
  - fügen den Aufruf des Konstruktors von Moebel als Inhalt in den Tisch-Konstruktor ein.
    Der Aufruf muss **`self`** als ersten Parameter enthalten

Vererbung

```
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()

        x,y = self.GibX(),self.GibY()
        b,t,w = self.GibBreite(),self.GibTiefe(),self.GibWinkel()

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

##    def GibX(self):
##        """Get-Methode fuer die x-Position"""
##        return self.__x
##
```

Ln: 8 Col: 0

Hilfsvariable für alle benötigten Attribute

Vererbung

Erläuterung zu den „*Hilfsvariablen für alle benötigten Attribute*“

- Diese Hilfsvariablen sind zunächst einmal nur eine Schreibvereinfachung in den folgenden Programmzeilen.
- Allerdings ist die vorher verwendete Zeile
  b,t =self.__b,self.__t
  nun nicht mehr zu verwenden, da auf die Attribute aus Moebel in Tisch nicht direkt zugegriffen werden kann (Kapselung).
- Also: Get-Methoden einsetzen!

Vererbung

Schritt 5:

- tisch.py, stuhl.py und raumplaner.py ausführen sollte erfolgreich sein.
- Die privaten Attribute sichern klare Zugriffe
- Klären Sie:
  - Welcher Klassentext wird ausgeführt?
  - Welche Methode GibFigur() wird ausgeführt? *(wenn sie auch in Moebel vorhanden ist)*
  - Wie wird auf die Attribute zugegriffen?