# Kapselung mit property

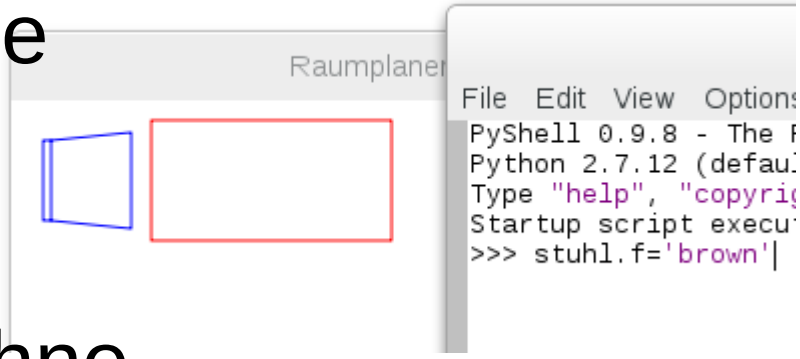
Kapselung umsetzen
mit
***property***

# Kapselung mit property

- Anlass ist (weiterhin) die bereits behandelte Aufgabe zum direkten Setzen des Attributwertes für die Farbe

```
stuhl.f="brown"
```

- Sie zeigt, dass der Wert verändert werden kann, ohne dass sich die Darstellung des Objekts Stuhl verändert.



# Kapselung mit property

- Ziel ist, dass Zugriffe auf Attribute alle Attribute nicht direkt ausgeführt werden (können), sondern nur
  - lesend über die Get-Methoden (*getter*) (*sondierende Methoden*)
  - schreibend über die Set-Methoden (setter) (*verändernde Methoden*)
- Direkte Zugriffe auf Attribute sind dann nur innerhalb der Klassendefinition möglich. (Sie können durch den Zugriff auf Get- und Set-Methoden auch dort vermieden werden.)

# Kapselung mit property

Umsetzung bei Python mit property:

- Wir bleiben bei den Attributnamen am Beginn mit zwei Unterstreichungsstrichen.
- Aufruf von

  ```
  stuhl.__b=100
  ```

  führt zu einem Fehler.

# Kapselung mit property

Umsetzung bei Python mit property:

- Wir markieren die Attribute wieder mit den beiden Unterstreichungsstrichen
- Eine Eingabe von `stuhl.__x=100` funktioniert,
- führt aber zu Unsinn, nämlich zur Definition eines neuen Attributs mit diesem Namen:
  Eingabe von `stuhl.__x`
  ergibt `100` ,
  allerdings ohne Wirkung beim eigentlichen Attributwert.

# Kapselung mit property

Umsetzung bei Python mit property x:

```
### --- property x ---
    def GibX(self):
        return self.__x

    def AendereXPosition(self,neueXPosition):
        istSichtbar=self.sichtbar
        if istSichtbar: self.Verberge()
        self.__x=neueXPosition
        if istSichtbar: self.Zeige()
    x = property(GibX,AendereXPosition)
```

# Kapselung mit property

Umsetzung bei Python mit property y:

```
### --- property y ---
    def GibY(self):
        return self.__y

    def AendereYPosition(self,neueYPosition):
        istSichtbar=self.sichtbar
        if istSichtbar: self.Verberge()
        self.__y=neueYPosition
        if istSichtbar: self.Zeige()
    y = property(GibY,AendereYPosition)
```

# Kapselung mit property

Umsetzung bei Python mit property breite:

```
### --- property breite ---
def GibBreite(self):
    return self.__b

def AendereBreite(self, neueBreite):
    if neueBreite<=0: return
    istSichtbar=self.sichtbar
    if istSichtbar: self.Verberge()
    self.__b = neueBreite
    if istSichtbar: self.Zeige()
breite=property(GibBreite,AendereBreite)
```

# Kapselung mit property

Umsetzung bei Python mit property tiefe:

```
### --- property tiefe ---
def GibTiefe(self):
    return self.__t

def AendereTiefe(self, neueTiefe):
    if neueTiefe<=0: return
    istSichtbar=self.sichtbar
    if istSichtbar: self.Verberge()
    self.__t = neueTiefe
    if istSichtbar: self.Zeige()
tiefe = property(GibTiefe,AendereTiefe)
```

# Kapselung mit property

Umsetzung bei Python mit property winkel:

```
### --- property winkel ---
def GibWinkel(self):
    return self.__w

def AendereWinkel(self, neuerWinkel):
    istSichtbar=self.sichtbar
    if istSichtbar: self.Verberge()
    self.__w = neuerWinkel
    if istSichtbar: self.Zeige()
winkel=property(GibWinkel,AendereWinkel)
```

# Kapselung mit property

Umsetzung bei Python mit property farbe:

```
### --- property farbe ---
def GibFarbe(self):
    return self.__f

def AendereFarbe(self, neueFarbe):
    istSichtbar=self.sichtbar
    if istSichtbar: self.Verberge()
    self.__f = neueFarbe
    if istSichtbar: self.Zeige()
farbe = property(GibFarbe,AendereFarbe)
```

# Kapselung mit property

Umsetzung bei Python mit property fuellfarbe:

```
### --- property fuellfarbe ---
def GibFarbe(self):
    return self.__ff

def AendereFuellfarbe(self, neueFarbe):
    istSichtbar=self.sichtbar
    if istSichtbar: self.Verberge()
    self.__ff = neueFarbe
    if istSichtbar: self.Zeige()
fuellfarbe =
    property(GibFarbe,AendereFuellfarbe)
```

# Kapselung mit property

Umsetzung bei Python mit property sichtbar:

```
### --- property sichtbar ---
def GibSichtbarkeit(self):
    return self.__s

def SetzeSichtbar(self, wie):
    if wie: self.Zeige()
    else: self.Verberge()
sichtbar=property(GibSichtbarkeit,SetzeSichtbar)
```

# Kapselung mit property

mit den bekannten Methoden:

```
def Verberge(self):
    self.__s = False
    Zeichenflaeche.GibZeichenflaeche().Entferne(self)

def Zeige(self):
    self.__s = True
    Zeichenflaeche.GibZeichenflaeche().Zeichne(self)
```

# Kapselung mit property

Erfolgreiche Aufrufe sind nun beispielsweise:

```
stuhl.x=100
stuhl.y=70
tisch.breite=140
tisch.tiefe=70
tisch.winkel=10
tisch.farbe='blue'
tisch.fuellfarbe='yellow'
schrank.sichtbar=False
schrank.sichtbar=True
```