Persistenz

Ich glaub'
ich krieg
pickle

Persistenz

- Wie sichern wir den Stand
  - nicht der Software,
  - nicht des Bildes
  - sondern des Zustands der erzeugten Objekte?

Persistenz

Eine Möglichkeit:

- Umwandlung der Daten jedes Objekts in eine Stringliste / Datenliste, die
  - in eine Datei gespeichert werden kann,
  - daraus nach einem erneuten Verwenden der Software eingelesen werden kann und
  - dabei den Zustand der Objekte wieder herstellt, wie sie beim Speichern gewesen sind.

Persistenz

Eine spezielle Python – Möglichkeit:

- Verwenden des Moduls Pickle
- Es stellt die Methoden zum
  - Speichern: pickle.dump
  - Laden: pickle.load

  bereit,
- die direkt alle Objekte speichern beziehungsweise laden.

Persistenz

… Objekte speichern beziehungsweise laden...

- Etwas präziser formuliert muss es heißen:
- "vollständige Informationen über die Objekte",
- denn es werden natürlich nicht die Objekte selbst gespeichert.

Persistenz

- Speichern

```
def Speichere(self, dateiname):
    """Objektdaten werden in Datei geschrieben"""
    with open(dateiname, 'wb') as objekt_datei:
        pickle.dump(self.__alleMoebel, objekt_datei)
        return 'OK'
```

Persistenz

- Beim Speichern kann etwas schief gehen.
- Das muss man abfangen

```
def Speichere(self, dateiname):
    """Objektdaten werden in Datei geschrieben"""
    try:
        with open(dateiname, 'wb') as objekt_datei:
            pickle.dump(self.__alleMoebel,
                        objekt_datei)
            return 'OK'
    except IOError as ioerr:
        return 'Dateifehler: ' + str(ioerr)
    except pickle.PicklingError as perr:
        return 'Pickle-Fehler: ' + str(perr)
```

Hinweise zu

```
with open(dateiname, 'wb') as objekt_datei:
```

- w kennzeichnet das Öffnen zum Schreiben, (r entsprechend Öffnen zum Lesen); das nachfolgende b kennzeichnet den Zugriff im Binärformat, also auf eine Binärdatei.
- `with` erzeugt einen eigenen Kontextbereich für die folgenden Anweisungen und gewährleistet ein ordnungsgemäßes Schließen der geöffneten Datei auch nach einem Fehler.

Persistenz

- Beim Laden entsprechend:

```
def Lade(self, dateiname):
    """Objektdaten werden aus der Datei gelesen"""
    ## Vorhandene Objekte aus Darstellung und Liste
    ## entfernen ! (fehlt hier)
    try:
        with open(dateiname, 'rb') as objekt_datei:
            self.__alleMoebel = pickle.load(objekt_datei)
            for moebel in self.__alleMoebel:
                if moebel.GibSichtbar():
                    moebel.Zeige()
            return 'OK'
    except IOError:
         return 'Datei nicht auffindbar!'
    except AttributeError as ae:
        return 'AttributeError: ' + str(ae)
    except pickle.UnpicklingError as perr:
        return 'Pickle-Fehler: ' + str(perr)
```