6. Objektorientierung

Montag, 14. November 2022 14:46

4. Objektorientierung

Die objektorientierte Programmierung (OOP) ist aus der modernen SW-Entwicklung nicht mehr wegzudenken und bietet einige Vorteile:

- Nähe zur natürlichen Umgebung:
  Alle Eigenschaften ähnlicher Objekte können in einer Klasse beschrieben werden. Objekte, die im realen Leben verbunden sind, sollten auch im Programm verbunden sein.
- Wiederverwendbarkeit:
  Erstmals erzeugte und überprüfte Implementierungen einer Klasse können in vielen anderen Anwendungen wieder eingesetzt werden und müssen nicht neu implementiert werden.
- Kontrollfunktion:
  Während der Übersetzung und der Laufzeit erfolgt eine strengere Überprüfung.

OOP ganz einfach: Sprechen wir mal von Dingen (um die Materie greifbar zu machen)

- Dinge haben Eigenschaften wie z.B. Größe, Farbe und Alter
- Dinge können Aktionen machen/bzw. mit diesen Dingen gemacht werden wie z.B. knurren, schmusen und schlafen

Bisher hatten wir in Python entweder Daten (was wir bei unseren Dingen mit Eigenschaften beschrieben haben) oder wir haben irgendwelche Aufgaben erledigt mit Funktionen und Methoden (in die wir zeitweise verschiedene Daten reingekippt haben).

Was aber passiert, wenn wir Daten und Methoden miteinander verknüpfen? Dann haben wir schon objektorientierte Programmierung (OOP) bzw. den Kerngedanken begriffen.

Wir trennen uns von den unspezifischen Datenstrukturen wie Variablen, Listen und Tupeln und gehen hin zu Datenstrukturen, die ein Objekt (sprich ein Ding) beschreiben.

Schauen wir uns einmal ganz konkret eine Katze an. Die ist schwarz, dick und frisst nur Milch, falls sie nicht schläft und heißt ChiChi. Überleg einmal, welche Eigenschaften von Katzen einem einfallen und was Katzen so machen.

Eigenschaften:

Katze
+ farbe + alter + name
+ mag_kartons() + schlaef_t()

Wir bauen uns also ein allgemeines Bild von einer Katze – einen Bauplan. Wir spielen mit Python Gott und schaffen einen allgemeinen Katzen-Zusammenbau-Plan. Das ist unsere Katzen-Klasse.

Und nun können wir virtuelle Katzen in beliebiger Anzahl erschaffen – sprich ganz viele Objekte, die grundlegend Gleich nach dem Bauplan aufgebaut sind, aber sich in Ihren Eigenschaften (Farbe, Alter, Name) unterscheiden und in der Ausprägung der Methoden.

```python
class cat():
    def __init__(self, name, age, color):
        self.name = name
        self.age = age
        self.color = color

    def likes_cardboard(self):
        pass

    def sleeping(self):
        pass
```

45