Systematisierung von Klassenbegriffen

Nach der Lernplattform tigerjython.ch und jython.ch

Begriffe

- Instanzvariablen
- Vererbung
- Klassenhierarchie
- Überschreiben
- Is-a-Relation

Instanzvariablen

- Die Eigenschaften oder Attribute eines Objekts werden als Instanzvariablen definiert.
- Sie besitzen für jedes Objekt der Klasse individuelle Werte.
- Der Zugriff auf Instanzvariablen innerhalb der Klasse erfolgt durch Vorstellen von *self*.
- Von ausserhalb der Klasse kann man durch Voranstellen des Instanznamens darauf zugreifen.
- Einer Klasse stehen aber auch die Variablen und Funktionen des Programm-Hauptteils zur Verfügung (zBsp.: alle Methoden der Klasse GameGrid).
- Die Methoden können auch eine Variable des Hauptteils verändern, falls sie in der Methode als *global* deklariert wird.

globale und lokale Variablen

- In vielen Programmiersprachen sind Variablen automatisch global, wenn man sie nicht explizit als lokal deklariert.
- In Python ist dies genau umgekehrt.
- Das heisst, dass jede Variable, die man innerhalb einer Funktion definiert, automatisch einen lokalen Gültigkeitsbereich hat.
- Was immer man mit dieser Variable innerhalb der Funktion macht, hat keinen Einfluss auf andere Variablen außerhalb der Funktion, auch wenn diese den gleichen Namen haben.

Instanzvariablen: Beispiel

Wir wollen «Tierobjekte» durch einen Mausklick erzeugen:

1. Die Klasse *Animal* wird definiert: *class Animal():*
2. Der Konstruktor wird innerhalb der Klasse definiert:
   def __init__(self, imgPath):
   Dieser wird immer aufgerufen, wenn ein Objekt der Klasse erzeugt wird.
3. Eine Methode wird innerhalb der Klasse definiert:
   def showMe(self, x, y):

Damit alle Methoden auf den Dateipfad des Konstruktors zugreifen können, muss dieser als Instanzvariable mit dem Präfix *self* innerhalb des Konstruktors abgespeichert werden.

Vererbung, Methoden hinzufügen

- Durch Vererbung erstellt man eine Klassenhierarchie und kann damit einer bestehenden Klasse zusätzliche Eigenschaften und Verhalten hinzufügen.
- Objekte der abgeleiteten Klasse sind automatisch auch Objekte der übergeordneten Klasse (auch **Ober-, Basis**- oder **Superklasse** genannt).
- Alle Eigenschaften und Methoden der übergeordneten Klasse können in der abgeleiteten Klasse verwendet werden.

Vererbung, Methoden hinzufügen: Beispiel

Ein Haustier ist ein Tier, dass zusätzlich noch einen Namen hat, den es mit *tell()* ausschreiben soll:

1. Man definiert daher eine Klasse *Pet*, die von *Animal* abgeleitet ist.
2. Der Tiernamen wird für jedes Haustier individuell bei seiner Erzeugung festgelegt.
   Deshalb wird er als Initialisierungswert dem Konstruktor von *Pet* übergeben, der ihn in einer Instanzvariablen abspeichert.

Klassenhierarchie, Methoden Überschreiben

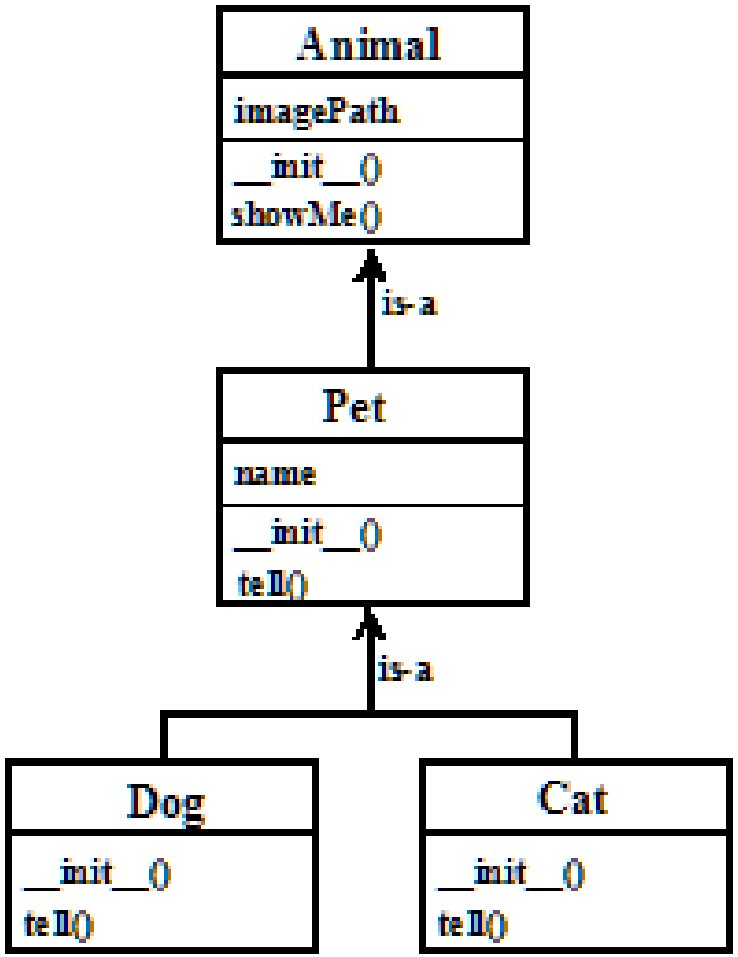
In einer abgeleiteten Klasse können Methoden der Basisklasse auch verändert werden, indem man sie mit dem gleichen Namen und der gleichen Parameterliste neu definiert.

Klassenhierarchie: Beispiel

Die Haustiere aus der Klasse *Animal* werden nun in Hunde und Katzen unterteilt, die unterschiedlich auf die Methode tell() reagieren.

1. Die Klasse *Dog* wird von der Klasse *Pet* abgeleitet und die Methode *tell()* wird entsprechend überschrieben.
2. Die Klasse *Cat* wird analog erstellt.

Klassendiagramm



Typenbezogener Methodenaufruf: Polymorphismus

Unter Polymorphismus versteht man den Aufruf von überschriebenen Methoden, wobei der Aufruf automatisch der Klassenzugehörigkeit angepasst wird.

Polymorphismus: Beispiel

Wir verwenden mit den vorher definierten Klassen eine Liste *animals:*

animals = [Dog(), Dog(), Cat()]

Beim Durchlaufen der Liste und Aufruf von *tell()* mit

```
for animal in animals
    animal.tell
```

tritt eine Schwierigkeit auf. Es gibt drei verschiedene Methoden von *tell()*, nämlich je eine in den Klassen *Pet*, *Dog* und *Cat*.
In einer polymorphen Programmiersprache wie Python findet der Computer heraus, um welche Sorte von Pets es sich handelt und ruft das entsprechende *tell()* auf.

Beispiel: Pizza-Kurier

Bei einem Pizza-Kurier können folgende Pizzen bestellt werden:

1. Margarita — Tomaten, Mozzarella, Oregano
2. Prosciutto — Tomaten, Mozzarella, Schinken
3. Vegetariana — Tomaten, Mozzarella, Gemüse
4. Napoli — Tomaten, Mozzarella, Schinken, Champignons, Oliven

Wir wollen ein Programm schreiben, welche die Bestellung entgegennimmt und die entsprechende Pizza liefert (keine echte Pizza, dafür jedoch ein hübsches Bildchen davon ☺)

Beispiel: Pizza-Kurier

- Da alle Pizzasorten Teig, Tomaten und Mozzarella enthalten, ist es sinnvoll diese Aufgabe mit Hilfe von Klassen und Vererbung zu lösen.
- Die Grundzutaten sind in der ***Basisklasse*** *Pizza* enthalten.
- Die übrigen Pizzen sind von der Klasse *Pizza* ***abgeleitet***. Sie ***erben*** alle Zutaten der Basisklasse und haben zusätzlich aber noch weitere Zutaten.
- Die Basisklasse Pizza hat Methoden *drawDoutgh(), drawTomato(), drawMozarella(), createPizza() und displayPrice()*.
- Pizza Napoli kann von der Pizza Prosciutto abgeleitet werden, da sie die gleichen Zutaten und zusätzlich noch Pilze und Oliven hat.

Literatur

- https://www.python-kurs.eu/python3_global_lokal.php
- http://www.tigerjython.ch/index.php?inhalt_links=navigation.inc.php&inhalt_mitte=gamegrid/klassen.inc.php
- http://jython.ch/index.php?inhalt_links=navigation.inc.php&inhalt_mitte=turtle/vererbung.inc.php