

Alexander Jenke, Theodor Straube

# Übung6: Klassen

Programmierkurs Python // Mittwoch, 04. Dezember 2019

# ToC

- OOP
- Klassen
  - Definition
  - Instanziierung
  - Vererbung
- Spezielle Funktionen
- Variablen-Sichtbarkeit

# OOP – Objektorientierte Programmierung

- Die Programmarchitektur wird in Anlehnung an die tatsächliche Umgebung der Anwendung entworfen.
  - Aufteilen des Problems in beteiligte Akteure und deren Aktionen
  - Einzelne Aufgaben werden mit den benötigten Informationen gekapselt
- Mehrere Akteure können in unterschiedlichen Zuständen sein
  - Einzelne Instanzen sind nach dem Akteur-Bauplan aufgebaut aber selbstständig
- Objekte können untereinander Interagieren und Nachrichten austauschen
- Akteure werden Klassen
- Instanzen werden Objekte
- Aktionen werden Funktionen
- Informationen werden Variablen
- Nachrichten werden Funktionsaufrufe und Zugriffe auf Variablen anderer Objekte

# Klassen

## Definition

- Syntax: *class* gefolgt von *Klassenname*
- Alles was zur Klasse gehört wird darunter um eine Stufe eingerückt
- Klassen können enthalten:
  - Klassen
  - Funktionen
  - Variablen
- Funktionen in einer Klasse haben als erstes Argument ``self``, welches das Objekt der Klasse enthält auf welchem die Funktion ausgeführt wird
- *pass* ist ein Platzhalter um Compiler-Fehler zu umgehen

```
class Computer:  
    pass
```

# Klassen

## Instanziierung

- Instanziierung mit *Klassenname()*
- In den Klammern werden die zur Initialisierung benötigten Parameter übergeben, so wie in der `__init__` Funktion gefordert.
- die `__init__`-Funktion initialisiert neue Instanzen
  - `self` ist das neue Objekt welches initialisiert werden soll
  - `*args` sind die hierfür benötigten Argumente, so wie von normalen Funktionen gewohnt

```
my_computer = Computer()
```

```
def __init__(self, *args):  
    pass
```

# Klassen

## Vererbung

- Klassen können von anderen Klassen erben und übernehmen damit alle objekt-unabhängigen Eigenschaften, wie
  - Funktionen
  - feste Variablen
- Funktionen der Oberklasse können überschrieben werden
- In der `__init__`-Funktion wird die `__init__`-Funktion der Oberklasse mit `super().__init__()` aufgerufen

```
class Laptop(Computer):  
    pass
```

# Spezielle Funktionen

- `__init__` wird aufgerufen, wenn eine neue Instanz initialisiert wird
- `__del__` wird aufgerufen, wenn eine Instanz gelöscht wird
- `__str__` wird aufgerufen, wenn `print(Instanz)` aufgerufen wird. Gibt lesbaren String zurück
- `__repr__` gibt einen String zurück der möglichst viele Informationen über die Instanz enthält
- `__getitem__` gibt das Element am geforderten Index zurück, wenn die mehrere Elemente verwaltet werden
- `__len__` gibt die Menge der verwalteten Elemente zurück
- `__call__` wird aufgerufen, wenn die Instanz als Funktion aufgerufen wird
- `__getattr__` wird aufgerufen, wenn das Attribut nicht an den üblichen Orten gefunden werden kann.
- `__add__` / `__sub__` / `__mul__` / `__div__` implementiert die + - \* / Operationen

# Variablen-Sichtbarkeit

- Auf Variablen außerhalb des Scopes kann zugegriffen werden, wenn diese mit dem Keyword `global` zugänglich gemacht wurden
- Variablen deren Namen mit einem Unterstrich beginnen sind *protected*  
d.h. sie können von außerhalb der Klasse nur gelesen, aber nicht beschrieben werden
- Variablen und Funktionen deren Namen mit zwei Unterstrichen beginnen sind *private*  
d.h. sie können von außerhalb der Klasse weder gelesen noch geschrieben werden



# nächste Woche

- Mehr Zeit für Aufgaben
- Zeit für Fragen
- Lösungen