self als Parameter

Erläuterung
zum Auftreten von
self als Parameter
bei Python

self als Parameter

Warum gibt es ein Problem?

- **`self`** kennzeichnet bei Python
  das Objekt selbst ["ich"; bei Java **`this`**]
- Die Verwendung von self ist zwingend

  [Die Bezeichnung self darf geändert werden;
  auch this wäre beispielsweise zulässig;
  bitte nicht machen!]
- self steht bei Definition und Aufruf an verschiedenen Stellen

self als Parameter

self bei [Instanz-] Methoden

- In der ***Definition*** der Methode muss self zwingend an der ersten Stelle stehen.
- Beim ***Aufruf*** dieser Methode ist dafür kein Parameter zu übergeben.
- Sie muss aber durch ein vorgestelltes self als Instanzmethode gekennzeichnet sein.

self als Parameter

- Beispiel

```
def BewegeHorizontal(self, weite):
    """Veraendernde Methode
    für die x-Position"""
    self.Verberge()
    self.__x += weite
    self.Zeige()
```

Aufruf der Methode aber mit nur einem Parameter:

```
self.BewegeHorizontal(50)
```

self als Parameter

self bei [Instanz-] Methoden

- In der Definition der Methode muss self zwingend an der ersten Stelle stehen.
-

```
def BewegeHorizontal(self, weite):
```

- ...

self als Parameter

self bei [Instanz-] Methoden

```
self.BewegeHorizontal(50)
```

- Beim Aufruf dieser Methode ist dafür kein Parameter zu übergeben.
- . . .

self als Parameter

self bei [Instanz-] Methoden

- . . .
-
```
self.BewegeHorizontal(50)
```
- Sie muss aber durch ein vorgestelltes self als Instanzmethode gekennzeichnet sein

self als Parameter

self im Konstruktor

- Im Konstruktor **`__init__(self, …)`** muss self zwingend an der ersten Stelle stehen.
- Beim Aufruf dieses Konstruktors ist dafür kein Parameter zu übergeben.
- Beim Aufruf des Konstruktors der Oberklasse dagegen muss self als erster Parameter übergeben werden.

self als Parameter

- Beispiel

```
class Schrank(Moebel):
    def __init__(self, xPos, yPos):
        Moebel.__init__(self, xPos, yPos)
```

Der Aufruf des Konstruktors von Schrank erfolgt aber mit nur zwei Parametern:

```
schrank = Schrank(100,50)
```

self als Parameter

self im Konstruktor

- Im Konstruktor __init__(self, …) muss self zwingend an der ersten Stelle stehen.
-

```
def __init__(self, xPos, yPos)
```

- . . .

self als Parameter

self im Konstruktor

```
schrank = Schrank(100,50)
```

- Beim Aufruf dieses Konstruktors ist dafür kein Parameter zu übergeben.
- . . .

self als Parameter

self im Konstruktor

- . . .
-

```
Moebel.__init__(self, xPos, yPos)
```

- Beim Aufruf des Konstruktors der Oberklasse dagegen muss self als erster Parameter übergeben werden.