Objektorientierte Programmierung

Operator Overloading

Was sind Operatoren?

Ein Operator ist eine Vorschrift, die aus einer Reihe von Operanden einen neuen Wert berechnet.

3 + 3

Der + Operator wurde verwendet, um die Summe aus 3 und 3 zu berechnen. Intern ist diese Vorgehensweise genau so in der Klasse int hinterlegt.

Built in Overloading

Genauso, wie der + Operator verwendet wurde, um eine Summe aus zwei ganzen Zahlen zu bilden, wird der selbe Operator verwendet, zwei Strings zu verbinden.

a = 'a' + 'b'

ab

Das heisst, für die Klasse string wurde der + Operator überladen.

Overloading in Python

In Python finden sich zahllose Beispiele für Operatoren-Overloading.

Zum Beispiel in der pathlib:

```
BASE_DIR = Path(__file__).resolve().parent
path_to_config = BASE_DIR / 'config.json'
```

Hier wurde der / Operator dahingehend überladen, dass er als Directory-Trenner verwendet werden kann.

Dictionaries verknüpfen

Ab Python 3.9 lassen sich Dictionaries mit dem Pipe-Symbol (binäres Oder) verknüpfen.

```
x = {'a': 3}
y = {'b': 3}

z = x | y
```

Overloading implementieren

Operator Overloading lässt sich einfach in eigene Klassen implementieren:

```
class A:
    [..]
    def __add__(self, other):
        if isinstance(other, A):
            return self.x + other.x
```

Hier ein Beispiel, wie der + Operator mit zwei Objekten der Klasse A funktioniert.

Übersicht der wichtigsten Operatormethoden

Methode	Operator
__add__(self, other)	+
__sub__(self, other)	-
__mul__(self, other)	*
__floordiv__(self, other)	//
__truediv__(self, other)	/
__mod__(self, other)	%
__pow__(self, other)	**
__lt__(self, other)	<
__le__(self, other)	<=
__eq__(self, other)	==
__ne__(self, other)	!=
__ge__(self, other)	>=