__main__ - Hauptprogramm festlegen

Im letzten Kapitel haben wir die Möglichkeit kennengelernt, Code aus einer anderen Datei zu importieren und somit die dort vorhandenen Funktionen nutzen zu können. Die Schreibweise der Funktionen wurde mit einem Punkt und dem Import-Namen verkettet.

Was passiert aber, wenn in der importierten Datei neben den Funktionen auch direkt ausführbarer Code enthalten ist?

```
def bspfunktionfuerrueckgabe(eingabewert):
        rueckgabewert = eingabewert * 2
        return rueckgabewert

print("Ich stehe in der Datei:")
```

Dann wird immer genau an der Stelle, wo diese Datei importiert wird auch zusätzlich der im Beispiel eingebaute `print`-Befehl ausgeführt.

Das ist in den meisten Fällen nicht wünschenswert.

Daher gibt es den Main-Bereich. Diesen erzeugen wir, indem wir den Dateinamen mit dem Systemwert `__main__` vergleichen. Den Dateinamen erhalten wir über `__name__`

Wenn wir uns den Inhalt von `__name__` ausgeben lassen, werden wir feststellen, dass wir bei direktem Aufruf als Rückgabewert `__main__` erhalten. Bei indirektem Aufruf erhalten wir den Dateinamen!

```
print(__name__)
```

Ausgabe auf Bildschirm:

__main__

Packen wir nun unseren Programmcode in eine andere Datei mit dem Namen „fktsammlung.py“. Hier deutet sich auch die Funktion der „externen“ Datei an - wir haben dort eine Sammlung von Funktionen, die wir immer wieder verwenden können. Dann wird der Wert von `__name__` den Dateinamen der importierten Datei enthalten.

Um jetzt überprüfen zu können, ob eine Datei direkt ausgeführt wird, werden beide Systemkonstanten in eine `if`-Bedingung gepackt. So wissen wir, ob die Datei direkt aufgerufen wird und somit der Code ausgeführt werden soll oder diese importiert wurde uns somit nur die Funktionen zur Verfügung stehen sollen.

Hier nun der Inhalt der Datei „fktsammlung.py“:

```
def bspfunktionfuerrueckgabe(eingabewert):
        rueckgabewert = eingabewert * 2
        return rueckgabewert

if __name__ == "__main__":
        print("Datei wurde direkt aufgerufen und die Main wird ausgeführt")
else:
        print("Datei wurde als Modul aufgerufen")

print("Ich stehe in der Datei: " + __name__)
```

Nutzen wir nun dieses Programm als Modul und rufen dieses über eine andere Datei auf (siehe letzte Kapitel) erhalten wir die entsprechende Ausgabe.

Unser aufrufendes Programm nennen wir „hauptprogramm.py“:

```
import fktsammlung
fktsammlung.bspfunktionfuerrueckgabe(3)
```

Als Rückgabe erhalten wir:

Datei wurde als Modul aufgerufen
Ich stehe in der Datei: fktsammlung