PG
Consulting

Anwendungsfall 1: Benannte Argumente

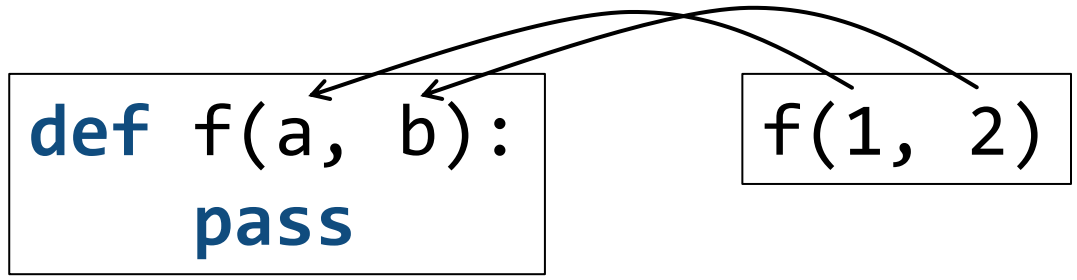
Bernhard Stadler

Klaudia Thellmann

PG Consulting

Was sind benannte Argumente?

- In Python können Argumente bei Funktionsaufrufen durch ihre Position auf Parameter abgebildet werden



```
def f(a, b):
    pass
```

```
f(1, 2)
```

- Alternativ kann man diese Zuordnung durch die Parameternamen aus der Funktionsdefinition vornehmen
- Dann ist die Reihenfolge der Argumente egal

```
def f(a, b):
    pass
```

```
f(a=1, b=2)
f(b=2, a=1)
```

```
f(a=1, 2)
f(b=2, 1)
```

SyntaxError: non-keyword arg after keyword arg

PG
Consulting

Beispiel in scikit-learn

```
from sklearn.tree import DecisionTreeClassifier
model = DecisionTreeClassifier(max_depth=3) # OK
model = DecisionTreeClassifier(maxdepth=3)  # Error
```

Beispiel in PyTorch

```
from torch.nn import Linear
model = Linear(in_features=1000, out_features=10)# OK
model = Linear(in_features=1000, out_featres=10) # Error
```

PG
Consulting

User Story

Als Python-Entwickler möchten wir schon während der Programmentwicklung eine Fehlermeldung bekommen, wenn wir in einem Aufruf einer Bibliotheksfunktion bei einem benannten Argument einen Parameternamen verwenden, der in der Definition der aufgerufenen Funktion nicht vorkommt.

Dadurch können wir Zeit sparen, weil wir das Programm nicht erst ausführen müssen, um diesen Fehler zu entdecken.

Hinweis: In künftigen Iterationen werden wir weitere Überprüfungen anfordern. Uns ist wichtig, dass diese Überprüfungen später mit geringem Aufwand in verschiedene IDEs und Code-Editoren eingebaut werden können und überall die gleichen Meldungen liefern.