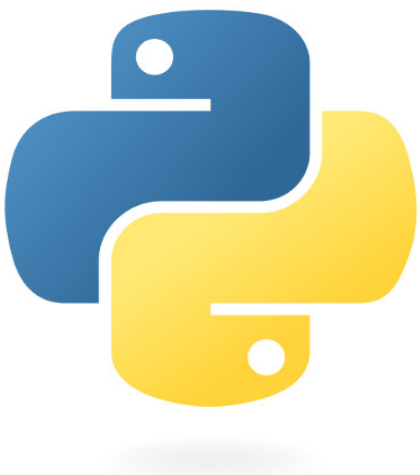
TRALIOS
Linux Server Management

Intern: Basic Python

...certainly uncontaminated by cheese



python™

Simon Fromme ▪ 16.10.2017

Eigenschaften

TRALIOS
Linux Server Management

- interpretiert (Referenzimplementierung: CPython (GPL-kompatibel))
- dynamisches Typsystem
- automatisches Speichermanagement
- Multi-Paradigma Sprache (objekt-orientiert, imperativ, functional, prozedural)
- große Standardlibrary mit hohem Funktionsumfang

Zitate

TRALIOS
Linux Server Management

Cuong Do, Software Architect, YouTube.com:

> *“Python is fast enough for our site and allows us to produce maintainable features in record times, with a minimum of developers.”*

Peter Norvig, director of search quality at Google, Inc.:

> *“Python has been an important part of Google since the beginning, and remains so as the system grows and evolves. Today dozens of Google engineers use Python, and we’re looking for more people with skills in this language.”*

Mark Shuttleworth, Thawte Consulting:

> *“Python makes us extremely productive, and makes maintaining a large and rapidly evolving codebase relatively simple.”*

TRALIOS
Linux Server Management

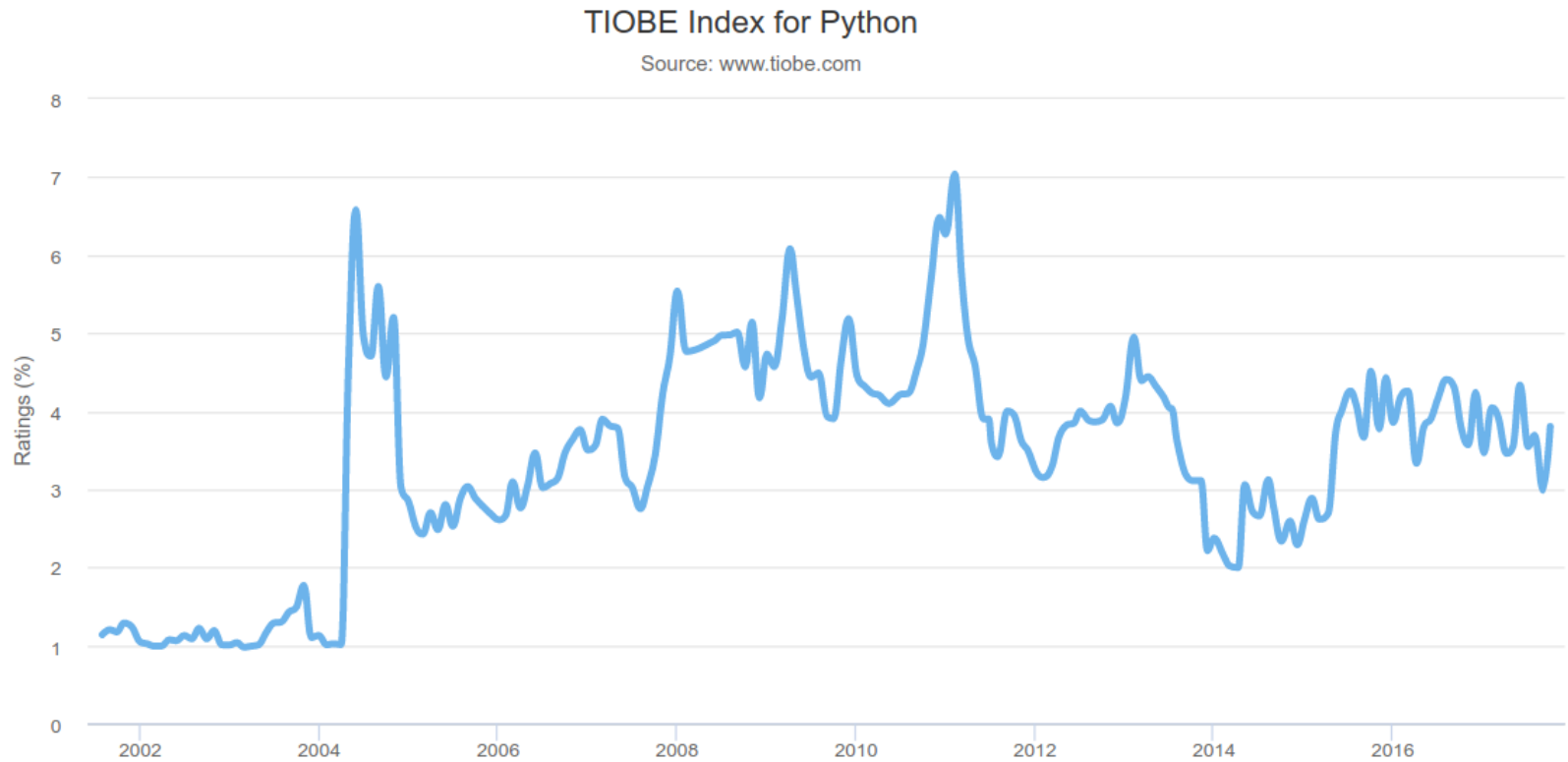


Abbildung: Tiobe Index Python (Quelle: https://www.tiobe.com/tiobe-index/python/)

Python vs. PHP

TRALIOS
Linux Server Management

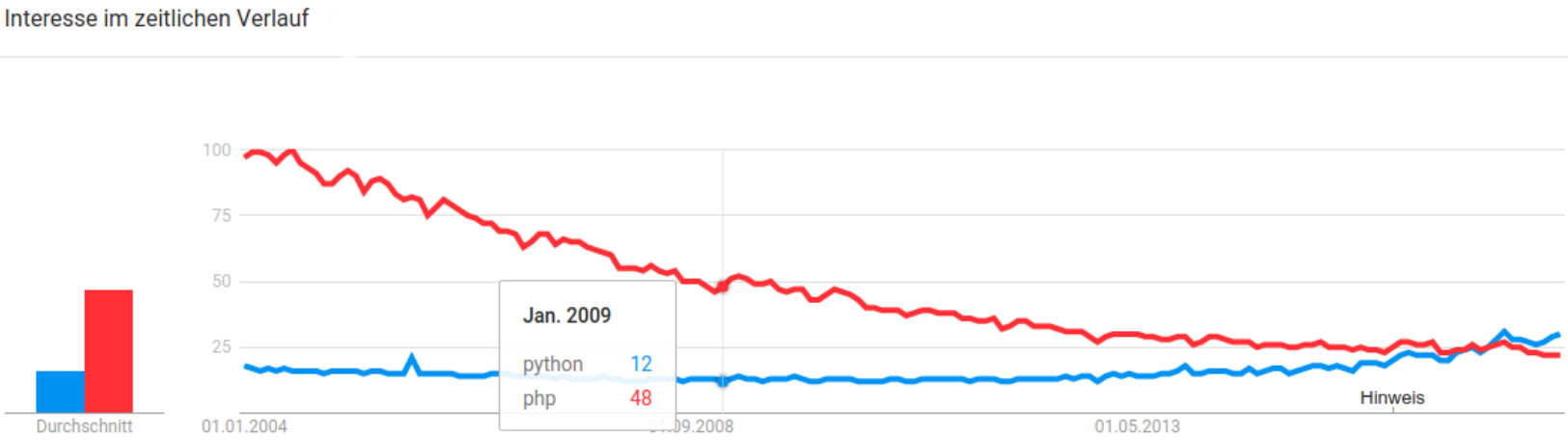


Abbildung: Python vs. PHP (Quelle: Google Trends)

Geschichte

TRALIOS
Linux Server Management

- Dezember 1989: Beginn Implementierung (Guido van Rossum)
- Januar 1994: Python 1.0
- Oktober 2000: Python 2.0 mit vielen neuen Features (z.B. List Comprehensions) , Garbage Collection mit Zyklenerkennung, Unicode, ...
- Dezember 2008: Python 3.0 (nicht rücktwärtskompatibel, Entfernung vieler redundanter Funktionsweisen), viele Features wurden auf Python 2.6 und 2.7 zurückportiert
- Dezember 2016: Python 3.6 (aktuelle Version)
- Alle Weiterentwicklungen in der 3.x Serie

Listen, Tupel, Sets, Dictionaries

TRALIOS
Linux Server Management

Listen (veränderliche Sequenz von Elementen beliebigen Typs)

```
some_list = [1, 2.0, "foo", "foo"]
```

Tupel (nicht veränderliche Sequenz von Elementen beliebigen Typs)

```
some_tuple = True, 3.14, "bar", 3.14
```

Sets (veränderliche Menge von Elementen beliebigen Typs, keine Duplikate)

```
some_set = {42, "tralios", False}
```

Dictionaries (veränderliche Zuordnung von Schlüsseln (hashbar) und dazugehörigen Werten)

```
some_dict = {"foo": "bar", 1:2, 5:[]}
```

TRALIOS
Linux Server Management

... Code Vorführung ...

Kontrollstrukturen

TRALIOS
Linux Server Management

```
while cond:
    # do stuff
else:
    # loop did exit prematurely

for var in collection:
      print(var):
else:
     # loop did exit prematurely
```

Kontrollstrukturen

TRALIOS
Linux Server Management

```
if x > 0:
    print('x is greater than 0')
elif x < 0:
    print('x is smaller than 0')
else:
    print('x is equal to 0')
```

Kontrollstrukturen

TRALIOS
Linux Server Management

```
import sys


try:
    f = open('myfile.txt')
    s = f.readline()
    i = int(s.strip())
except OSError as err:
    print("OS error: {0}".format(err))
except ValueError:
    print("Could not convert data to an integer.")
except:
    print("Unexpected error:", sys.exc_info()[0])
    raise
```

Kontrollstrukturen

TRALIOS
Linux Server Management

```
if x > 0:
    print('x is greater than 0')
elif x < 0:
    print('x is smaller than 0')
else:
    print('x is equal to 0')
```

Best Practices

TRALIOS
Linux Server Management

Iterieren über eine Collection

TRALIOS
Linux Server Management

```
colors = ['red', 'green', 'blue', 'yellow']

for i in range(len(colors) - 1, -1, -1):
    print(colors[i])

for color in reversed(colors):
    print(color)
```

Iterieren über mehrere Collections

TRALIOS
Linux Server Management

```
names = ['Bob', 'Alice', 'Eve']
colors = ['red', 'green', 'blue', 'yellow']

n = min(len(names), len(colors))
for i in range(n):
    print(names[i], '-->', colors[i])

for name, color in zip(names, colors):
    print(name, '-->', color)
```

Iterieren über sortierte Collections

TRALIOS
Linux Server Management

```
colors = ['red', 'green', 'blue', 'yellow']

for color in sorted(colors):
    print(color)

for color in sorted(colors, reverse=True):
    print(color)

for color in sorted(colors, reverse=True, key=len):
    print(color)
```

Verschiedene Exit Points in Schleifen

TRALIOS
Linux Server Management

```
def find(seq, target):
    found = False
    for i, value in enumerate(seq):
        if value == target:
            found = True
            break
    if not found:
        return -1
    return i
```

```
def find(seq, target):
    for i, value in enumerate(seq):
        if value == target:
            break
    else:
        return -1
    return i
```

Iterieren über mehrere Collections

TRALIOS
Linux Server Management

```
names = ['Bob', 'Alice', 'Eve']
colors = ['red', 'green', 'blue', 'yellow']

n = min(len(names), len(colors))
for i in range(n):
    print(names[i], '-->', colors[i])

for name, color in zip(names, colors):
    print(name, '-->', color)
```

Iterieren über mehrere Collections

TRALIOS
Linux Server Management

```
names = ['Bob', 'Alice', 'Eve']
colors = ['red', 'green', 'blue', 'yellow']

n = min(len(names), len(colors))
for i in range(n):
    print(names[i], '-->', colors[i])

for name, color in zip(names, colors):
    print(name, '-->', color)
```