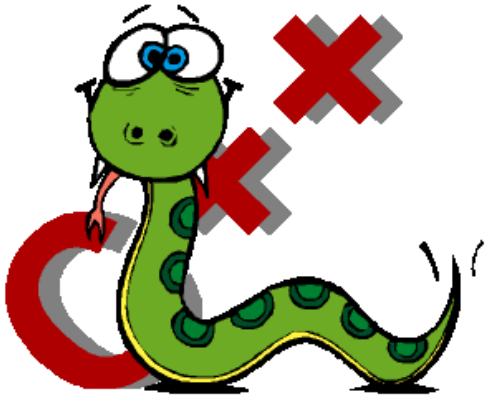
Python
Einführung in die Programmiersprache

Gerald Senarclens de Grancy
santa@sbox.tugraz.at
http://grancy.at.tf



Inhalt

- grundsätzliche Fähigkeiten und Merkmale von Python
- Python Maintainer
- Entwicklungsumgebungen und Tools
- Einführung in die Syntax
- fortgeschrittene Techniken
- allgemeine Bibliotheken, GUI Entwicklung
- bekannte Anwendungsbeispiele

Grundsätzliche Fähigkeiten und Merkmale

- Open Source
- objektorientierte Skript- und Programmiersprache
- rein dynamische Sprache (keine Typdeklaration notwendig)
- plattformunabhängig (Windows, Unix, Linux, Mac, Amiga, BeOS, Win CE, Dos, QNX, Psion Series 5, OpenVMS, VxWorks sowie weitere Umgebungen, auf denen eine Java Virtual Machine läuft)
- stellt optimierte "high level" Datentypen zur Verfügung
  - Strings, Tuples, Listen, Dictionaires und Dateien
- simple und elegante Syntax (schnell und einfach zu lernen)

- 1990 entwickelt von Guido van Rossum
  - Anfänge im "Centrum voor Wiskunde en Informatica"
  - derzeit gewartet und weiterentwickelt in der Corporation for National Research Initiatives (CNRI)
- Python Software Foundation (PSF)
  - verantwortlich für die Weiterentwicklung der Sprache
- Python Business Forum (PBF)
- die Community, SIGs (Special Interest Groupes)

- Standard Python Software
  - Python Debugger

```
>>> import pdb
>>> pdb.run('function(arguments)')
```

- GNU Emacs Python mode
- Python Shell (zum "Herumspielen", Ausprobieren von Statements und um Erfahrung zu gewinnen)
- IDLE
- Boa Constructor, wxGlade
- hunderte Bibliotheken, die mit Python mitgeliefert werden (sowie zahllose weitere, die kostenlos im Internet erhältlich sind)

- `$ python program.py`
- als Shellscript (unter MS Windows nicht möglich)
  - `#!/usr/bin/env python`
  - `$ chmod u+x program.py`
- `import program`
- in einer IDE (Integrated Development Environment)

- Wahrheitswerte
  - leere Sequenzen und Werte, die 0 ergiben werden als “false” evaluiert

    `[], {}, (), 0, 0.0, None`
- if

```
if test:
   suite
[elif test:
   suite]
[else:
   suite]
```

- while

```
while test:
   suite
[else:
   suite]
```

- for

```
for target in sequence:
   suite
[else:
   suite]
```

- **`pass`** (leeres Statement, wie ";" in Java)
- **`break`** (Ausführung wird nach der innersten Schleife fortgesetzt)
- **`continue`** (springt zur nächsten Iteration der innersten Schleife)
- **`return`** (beendet die aktuelle Funktion und gibt einen Wert oder **`Null`** zurück)
- no **`switch case`**
  - lediglich durch die Indizierung von dictionaries möglich

- Strings (immutable)
  “`Python's`”, `r'Python`”`s raw'`, `u`“““`multiline unicode`”””
  - Strings werden automatisch konkateniert
  - Escapesequenzen: `%` `\n`, `\t`, \\,...
  - wichtige Funktionen eines Strings s

```
s.strip(), s.lstrip()
s.find(sub [, start [, end]])
s.rfind(sub [, start [, end]])
s.replace(old, new [, maxtimes]
s.capitalize(), s.swapcase(), s.islower()
...
```

- Tuples (immutable)

```
(), (3), (1, 6, 2), ('a', 4, 'string', 34.876, (3, 1))
```

- Lists (mutable)

```
[], [3], [1, 6, 2], ['a', 4, 'string', [[1], (3, 1)],
(1, [2])]
```

  - wichtige Methoden einer Liste l

```
l.append(element), l.insert(i, element)
l.remove(element)
l.index(element), l.sort([function])
l.count()
l.pop([i])
...
```

- Sequenzoperatoren
  - Konkatenation `+`
  - Wiederholung `*`
  - Indizierung `[i]` (`l[0]`, `s[4]`)
  - Slicing `séq[start:end]` (`t[1:3]`, `l[:]`, `s[:-3]`)
  - sowie einige mehr

```
elem in seq, elem not in S
for elem in seq
len(seq)
min(seq), max(seq)
```

- Dictionaries (mutable)
  `{}, {'spam': 1, 'egg': 5}, {'d': 'Ed':1, 'l':[1, 2] }`
  - wichtige Funktionen eines Dictionaries d

```
d.keys(), d.values(), d.has_key(key)
d.copy(), d.clear()
d.update(dictionary)
d.get(key, [default]), d.setdefault(key, [default])
d.popitem()
...
```

- Dateien
- Lesezugriff

```
file = open('filename', 'r')
file.read(), file.read(n)
file.readline(), file.readlines(), file.xreadlines()
```

- Schreibzugriff

```
file = open('filename', 'w')
file.write(string), file.writelines(stringlist)
```

- weitere Methoden und Konstanten

```
file.closed, file.mode, file.name
file.close(), file.tell()...
```

- Funktionen

```
def myFunction(arg1, arg2):
    return arg1, arg2
def myFunction2(arg='default value'):
    print arg
def myFunction3(*args):
    for arg in args:
        print arg
def myfunction4(**args):
        keys = args.keys()
        for key in keys:
            print args[key]
```

- objektorientierte Programmierung
  - Klassen sind "Objektfabriken"

```
class MyClass[(superclass)]:
   static = 'salut'
   def __init__(self, arg):
      self.property = arg
   def __repr__(self):
      reprStr = 'property = ' + str(self.property) + '\n'
      reprStr += 'static = ' + str(self.static)
      return reprString
   def setProperty(self, newValue):
      self.property = newValue
   def getProperty(self):
      return self.property
myObject = MyClass(arg)
```

- Exceptions

```
myError = 'a very simple exception'
def problemFunction():
   raise myError, 'additional description of the exception'
try:
   problemFunction()
except myError, data:
   print 'Exception caught:', data
else:
   print 'no exception occured'

try:
   raise 'new Exception'
finally:
   print 'doing cleanup'
```

- das `os` Modul zur Interaktion mit dem Betriebssystem
  - wichtig für plattformunabhängiges Programmieren
    `os.sep`, `os.linesep`,...
  - mit der folgenden Funktion können Befehle an die Shell des OS weitergegeben werden `os.system('shell command')`
- `sys`
  - dient zur Interaktion mit dem Python Interpreter (eg. `sys.argv`)
- `pickle` (ermöglicht einfaches speichern von Objekten in Dateien)
  `pickle.dump(myObject, file)`, `object = pickle.load(file)`
- einige weitere: `string`, `re` (reguläre Ausdrücke),...

- eine kleine Auswahl an bekannten GUI-Bibliotheken
  - Tkinter
  - wxPython
  - Java Swing (durch Verwendung von Jython kann man auf das komplette Java API und alle Javabibliotheken zugreifen)
- bekannte Anwendungsbeispiele
  - BitTorrent
  - Boa-Constructor
  - Nicotine

- http://www.python.org
  - offizielle Python Dokumentation
- das Python Tutorial
- Python Newsgroups
  news://your.news.server/comp.lang.python
  news://your.news.server/comp.lang.python.announce
- Buchempfehlungen
  - Learning Python, 2nd Edition
  - Python Pocket Reference 2nd Edition

You can obtain the document's LaTeX sources at http://www.sbox.tugraz.at/home/s/santa/ (section Lectures), where you can also find a French version of this document. All examples I used during my presentations are freely available at the former mentioned webside.