Python & LaTeX

Dante e.V. Frühjahrstagung 2017

Dr. Uwe Ziegenhagen

22. März 2017

Überblick

Was machen wir heute?

- Python Grundlagen
- Python in LaTeX Dokumenten
- Erzeugung von LaTeX Dokumenten

Voraussetzungen

Was wird benötigt

- Aktuelle TEX Installation
- Python3 Installation, vorzugsweise Anaconda/Winpython
- Pakete:
  - numpy
  - jinja2
  - pandas
- Diese Folien und Code-Beispiele unter:

`https://github.com/UweZiegenhagen/PythonAndLaTeX`

Testen der Installation

- Funktionieren die folgenden Befehle?

```
1 import jinja2
2 import pandas
3 import numpy
```

Python

- Erfunden von Guido van Rossum (Niederlande)
- Fokus auf lesbaren und verständlichen Code
- „batteries included“ $\Rightarrow$ umfangreiche Standardbibliothek
- Mein erster Kontakt mit Python: Downloadskript für SaveTV
- Python2 versus Python3 $\Rightarrow$ Python3
- Editor? Ich nutze Spyder3

Python „Hello World“

```
1  print('Hello Python')
2  # Kommentar
3  a = 123.4
4  a+=2
5  print(a+2)
6
7  def myFunction(a):
8      b = a + a
9      return b
10
11 print(myFunction(2)) # 4
12 print(myFunction('a')) # 'aa'
```

Listing 1: Hello World in Python 3.x

Strings, Listen und Tupel

Strings

```
1  a = 'Hallo'
2  b = 'Welt'
3
4  c = a + ' ' + b
5  'W' in c # True
6  print(c[0]) # 'H'
7  print(c[-1]) # 't'
8  print(c[1:3]) # 'all'
9
10 for i in c:
11   print(i)
```

Listing 2: Strings

Strings, Listen und Tupel

Stringfunktionen

```
1 meinString = 'Hallo Welt'
2
3 meinString.upper()
4 meinString.find('Welt')
5 meinString.split(' ')
6 meinString.replace('Welt', 'World')
```

Listing 3: Strings

Strings, Listen und Tupel

Listen

```
1
2 beatles = ['John', 'Paul', 'Ringo', 'George']
3 print(len(beatles))
4 beatles[0]
5 beatles.append('')
6 beatles.index('John')
```

Listing 4: Listen

Strings, Listen und Tupel

Tupel

```
1 monate=('Jan', 'Feb', 'Mar', 'Apr', 'Mai')
2 monate[1]
3 monate[1:3]
```

Listing 5: Tupel

Dictionaries

Key-Value Paare

```
1 lookup={'EUR':'Euro', 'GBP':'Pound', 'USD':'US-Dollar'}
2 lookup['EUR']
```

Listing 6: Tupel

Flusssteuerung

if/then

```
1 if condition:
2   DoThis()
3 else:
4   pass # pass = "Do nothing"
```

Listing 7: if-then

"condition"kann ein üblicher Boolean Ausdruck sein.

Flusssteuerung

for

```
1 myString = 'Python'
2 for c in myString:
3     print(c)
```

Listing 8: for-Schleifen

Flusssteuerung

while

```
1 n = 4
2 while n>0:
3   print(n)
4   n-=1
```

Listing 9: while-Schleifen

Flusssteuerung

break & continue

Listing 10: break & continue

Funktionen

```
1 def add(a,b):
2   return a+b
3
4 def multiply3(a,b,c=1):
5   return a*b*c
```

Listing 11: Definition von Funktionen

Ein- und Ausgabe

Kommandozeile

```
1 a = input('Erstes Wort')
2 b = input('Zweites Wort')
3
4 print(a, b)
5 print(a, b, sep='')
6 print(a, b, sep=':')
```

Listing 12: Ein- und Ausgabe: Kommandozeile

Ein- und Ausgabe

Dateien lesen

```
1 dateiname = 'exampleFile.txt'
2
3 filepointer = open(dateiname,'r')
4 # 'r' (read) oder 'a' (append)
5 for zeile in filepointer:
6         print(zeile)
7
8 filepointer.close()
```

Listing 13: Ein- und Ausgabe: Dateien

- Um Excel, CSV und ähnliches zu lesen $\Rightarrow$ pandas

Ein- und Ausgabe

UTF8-Dateien lesen und schreiben

```
1  import io
2  import datetime
3
4  jetzt = datetime.datetime.now()
5  dateiname = 'example.txt'
6
7  with io.open(dateiname,'w',encoding='utf8') as datei:
8      datei.write('äüöß ' + jetzt.isoformat())
9
10 with io.open(dateiname,'r',encoding='utf8') as datei:
11     text = datei.read()
12     print(text)
```

Listing 14: Ein- und Ausgabe: UTF8