Berner Fachhochschule
Haute école spécialisée bernoise
Bern University of Applied Sciences

FS24 CAS PML - Python

Niklaus Johner

niklausbernhard.johner@bfh.ch

FS24 CAS PML - Python

13. standard libraries

Module importieren

- Ein Modul importiert man mit *import modulename.*
  - Importiert das ganze Modul als ein Objekt in der Variablen *modulename*

```
import math
math.sqrt(4) # 2
```

  - Man kann auch einen Namen für die Variable angeben:

```
import math as m
m.sqrt(4) # 2
```

Module importieren

- Von einem Modul eine spezifische Funktion importieren

```
from math import sqrt
sqrt(4) # 2
```

- Von einem Modul alle Funktionen importieren

```
from math import *
sqrt(4)
sin(pi)
```

Alles geht direkt in den aktuellen namespace!

Das *math* Modul

- Das *math* Modul enthält die erwarteten mathematischen Funktionen wie:

```
x = 2.3
math.nan ; math.isnan(x)
math.sqrt(x) ; math.pow(x, 2)
math.sin(x) ; math.asin(x)
math.cos(x) ; math.acos(x)
math.exp(x) ; math.log(x)
math.ceil() ; math.floor()
```

Das *os* Modul

- Das *os* Modul enthält Funktionen die vom OS abhängig sind
- Elemente (file oder directories) auflisten, die an einem gewissen Pfad liegen:

```
file_dir_list = os.listdir("path/to/dir")
```

- Directory kreieren:

```
os.mkdir("path/to/dir")
```

- Shell command ausführen:

```
os.system("ls ~/")
```

Das *os* Modul

- *os.path* um mit Directories und Dateien Pfade zu arbeiten
- Einen Pfad generieren:

```
my_dir = os.path.join("dir", "sdir", "ssdir")
my_file = os.path.join(my_dir, "fname")
```

- Prüfen ob ein Pfad, ein Directory oder ein File existiert

```
os.path.exists(my_dir)
os.path.isdir(my_dir)
os.path.isfile(my_dir)
```

Die *sys* und *subprocess* Modul

- *sys.argv* : enthält die Liste von command line arguments wenn python ein script ausführt

```
nj-mbp:~ njohner$ cat test.py
import sys
print("sys.argv= {}".format(sys.argv))
nj-mbp:~ njohner$ python test.py 2 3
sys.argv= ['test.py', '2', '3']
```

- *subprocess.call* um ein shell command auszuführen

```
import subprocess
subprocess.call(["ls","-lt","/some/directory"])
# ls -lt /some/directory
```

Das *pdb* Modul

- Das python debugger Modul.

```
import pdb; pdb.set_trace()
```

- Das Modul importieren und einen breakpoint setzen.
- Wenn diese Zeile ausgeführt wird, wird die Ausführung gestoppt und man landet in einer interaktiven debugger session.