

Programmieren

Dieter Probst

TEKO Bern

B-NIA-22-A-a

Übersicht

1 Lektion 1

Lernziele

Nach der erfolgreichen Absolvierung dieses Kurses kennt ihr

- die wichtigsten Konzepte der Programmiersprache Python (Unpacking, Comprehension, Iteratoren, Lambda Function, Exception Handling, Magic Methods, Decorators, Generators, ...)
- kennt die Datenstrukturen von Python und ihre wichtigsten Methoden diese zu Erzeugung und zu Manipulieren (Listen, Mengen, Diktionäre (assoziierte Listen), Arrays, ... und könnt bei Bedarf eigene Datenstrukturen implementieren (z.B. Bäume) und zur Lösung von Programmieraufgaben einsetzen,
- kennt das **Model-View-Controller Pattern**,

Lernziele

- könnt sauberen, verständlichen und wartbaren Code (clean Code) schreiben und dies in einem kleinen Projekt anwenden,
- könnt in einer Gruppe eine einfache GUI-Anwendungen entwickeln.

Wieso gerade Python?

- Eine der populärsten, am weitest verarbeiteten Programmiersprachen.
- Einfache Syntax, dynamisch getypt, wenig Ballast.

```
// Hello World in Java
class HelloWorld {
    public static void main(String[] args) {
        System.out.println("Hello, World!");
    }
}
```

```
# Hello World in Python
print("Hello, World!")
```

Wieso gerade Python?

- Grosses Angebot an Bibliotheken und Werkzeugen.
- Multi-Paradigma (OOP, prozedurales Programmieren, funktionales Programmieren, ...)

Benotung

- Keine Diplomprüfung,
- Fachnote: Mittelwert aus einer Einzelarbeit mit mündlicher Prüfung (2 Noten), eine Gruppenarbeit, ev. eine weitere schriftlich Prüfung (vor Ort).

Ratschläge

- Seien sie anwesend im Unterricht.
- Vertiefen und repetieren Sie die Inhalte zuhause.
- Wenn sie etwas nicht verstehen, fragen Sie!
Dumme Fragen sind nur die, die nicht gestellt werden.

Ziel für heute

- Kennenlernen, eure Programmiererfahrung, welches Betriebssystem nutzt ihr, Erwartungen an den Kurs.
- Online Ressourcen zu Python.
- Erste Schritte mit Python.
- Installation von Docker, ev. Installation von Python, pip, Jupyterlab, Visual Studio Code.

Python: einige Web-Ressourcen

- www.w3schools.com.
Einfach, mit vielen Beispielen, ev. nicht die ganze Wahrheit.
- www.programiz.com/python-programming. Immer noch leicht verständlich, mehr Information, detailliertere Erklärungen.
- <https://docs.python.org/3/>. Offizielle Dokumentation. Präzise, die ganze Wahrheit (oft zuviel Information für ein erstes Mal).
- <https://docs.python.org/3/tutorial/>.
Offizielles Tutorial. Deckt alles ab!
- stackoverflow.com, stackexchange.com, (wie Programmierer es tatsächlich machen).

Einige Python Tutorials

- <https://www.w3schools.com/python/default.asp>.
- www.programiz.com/python-programming.
- <https://docs.python.org/3/tutorial/>.
Offizielles Tutorial. Deckt alles ab! Sehr zu empfehlen.

Visualisieren von Python Code

- Lade `pythontutor_examples.py` aus Classnotebook/L1 herunter.
- Gehe auf <http://pythontutor.com> und visualisiere die Ausführung des Codes (Anleitung als Kommentar in `pythontutor_examples.py`).

Online Python Code ausführen - Turtle

- Lade `turtle_examples.py` aus Classnotebook/L1 herunter.
- Gehe auf trinket.io/python und führe die Beispiele aus. (Anleitung als Kommentar in `turtle_examples.py`).
- Das Module Turtle ist auf docs.python.org/3/library/turtle.html dokumentiert.

Jupyternotebook online testen

Jupyter Notebooks bieten eine web-basierte interaktive Umgebung, in welcher Dokumente mit ausführbarem Python Code dargestellt und editiert werden können.

- Lade die Files `.ipynb` aus `Classnotebook/L1` herunter.
- Gehe auf jupyter.org/try-jupyter/lab. Klicke auf den Upload Pfeil und lade das File `Erste_Schritte_1.ipynb` herauf.
- Code in einer Zelle ausführen: `shift –ENTER` drücken.

Installation von Docker

- Windows: Google "install docker", download Installer, folge den Instruktionen.

Computer muss zur Fertigstellung der Installation neu gestartet werden, ev. werdet ihr dann aufgefordert ein weiteres Tool zu installieren. Folge den Instruktionen, wähle immer die vorgeschlagene Option.

Starte, falls gefragt, Docker Desktop und klicke dich durch "get started". Du brauchst nicht genau zu verstehen was passiert, dies dient nur zu testen ob alles funktioniert.

- Ubuntu: folge z.B. [dieser Anleitung](#).

Docker Image für diesen Kurs von dockerhub pullen

- In Windows Powershell oder Command Prompt öffnen.

```
docker pull dieterprobst/jlab
```

Der Image wird nun heruntergeladen (das dauert eine Weile).

Docker Image starten, Ordner einbinden

- Erstelle einen Ordner "Programmieren".
- Windows: Klicke auf den Desktop Link der Docker Desktop started.
- Klicke rechts auf Images. Bewege die Mause über die Zeile dieterprobst/jlab und klicke "run". Expandiere "Optional Settings".

```
docker pull dieterprobst/jlab
```

Ports

Local Host	Container-Port
------------	----------------

10000	8888/tcp
-------	----------

Volumes

Host Path	Container Path
-----------	----------------

waehle den Ordner Programmieren	/home/jovyan/work
---------------------------------	-------------------

Dann klicke "run"

Docker Image starten, Ubuntu und Powershell

Öffne Powershell und wechsle ins Verzeichnis Programmieren. Gib nachstenden Befehl ein (auf einer Zeile!)

```
> docker run -d --rm -p 10000:8888  
-v ${PWD}:/home/jovyan/work dieterprobst/jlab
```

Ubuntu:

```
$ sudo docker run -d --rm -p 10000:8888  
-v "${PWD}"/:/home/jovyan/work dieterprobst/jlab
```

Via Brower auf Container zugreifen

- Windows: In Docker Desktop links auf Container/Apps klicken, dann mit Browser verbinden klicken.
- Ubunut: Im Browser die url localhost:10000 eingeben.

Erste Schritte im Container

```
$ cd work  
$ echo "Nachname" > .username.txt  
$ git clone https://github.com/didi64/NIA22Prog.git  
$ cp -r NIA22Prog/L1/*.ipynb L1
```

- Im Notebook sollten nun links im Ordner work die Ordner A1, NIA22Prog und L1 vorzufinden sein. Änderungen im Ordner L1 werden beim verlassen des Containers nicht gespeichert. Das Original .ipyn liegt in NIA22Prog/L1.
- Um ein bearbeitetes Notebook zu speichern, ist es in den Ordner A1 zu kopieren (Im linken Fenster File in Ordner A1 ziehen).

Python und pip und Visual Studio Code installieren

- Je nach OS. Am besten Python 3.9, da ist pip mit dabei.
- Auf <https://docs.python.org/3/using/index.html> nach deinem OS suchen und Anleitung folgen.
- Windows: ev. einfacher, <https://docs.microsoft.com/de-de/windows/python/beginners> folgen.
- <https://code.visualstudio.com> mit Python-Langauge-Extension installieren.

Viel Glück!

Packages mit pip installieren

- ```
>pip install turtle
```

```
>pip install pytube
```

```
>pip install ipython
```

```
>pip install jupyterlab
```
- Um .ipynb files mit double-click starten zu können sind ev. .ipynb files mit jupyter-lab zu assoziieren:

```
>pip install nbopen
```

```
>python -m nbopen.install_win
```

## Visual Studio Code

Folge den Instruktionen [hier](#):

- Falls noch nötig: “VS Code”, “Python Interpreter” und “Python extensions for VS Code” installieren.
- Ordner erstellen, wo dein Python Projekte ablegt wird (z.B. `Python_VS`), und Unterverzeichnisse `Test` und `Turtle`.
- Kopiere die Dateien `turtle_functions.py` und `tf.py` aus Class Materials auf Teams in den Ordner `Turtle`.
- Öffne Visual Studio Code z.B.
  - 1 via Start,
  - 2 aus Terminal. Navigiere z.B. in den Ordner `Python_VS/Test`, dann `>code` .

## Visual Studio Code

- Python Interpreter auswählen:
  - 1 Drücke `Shift-Ctrl-P`.
  - 2 Beginne "Python: Select Interpreter" zu Tippen, bis dies in der Auswahl erscheint. Es sollte bereits ein Interpreter ausgewählt sein, etwa `.../Python39/Python.exe`.
- Benutze VS Explorer (File Icon oben links) und erstelle ein neues File `test.py`. Schreibe ein kurzes Programm, z.B.

```
x=y=3
print(x,y)
```



## Visual Studio Code

- Möglichkeiten, das Programm zu starten:
  - 1 Klicke auf das grüne Dreieck oben rechts.
  - 2 Drücke Ctr-F5.
  - 3 Rechtsklicke im Fester mit Code, wähle "Run Python File im Terminal".