Proseminar Python

Grundlagen in Interaktion mit Python

Anna Fuchs Jannek Squar

2022-04-19

Scientific Computing
Universität Hamburg

Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

Table of Contents

Entwicklungsstufen

(Mögliche) Schritte, um Python Skripte zu entwerfen

- Einzeiler testen (Interpreter)
- Rudimentäre Skript-Blöcke bauen
- Fortgeschrittene Programminfrastruktur aufbauen

Code schreiben

Möglichkeiten Python Code zu schreiben:

- Standard Python Interpreter
- Editor (z.B. Sublime[1], Atom[2], Visual Studio Code[3])
- Shellbasierte IDE: IPython
- GUI-IDE (z.B. PyCharm[4], Spyder[5])

[1] https://www.sublimetext.com/
[2] https://atom.io/
[3] https://code.visualstudio.com/
[4] https://www.jetbrains.com/pycharm/
[5] https://www.spyder-ide.org/

Python Interpreter[6]

- Simpler Aufruf aus der Kommandozeile
- Immer verfügbar
- Eingeschränkte Funktionsvielfalt
- Ausreichend für einfache Tests
- Return-Wert wird immer ausgegeben

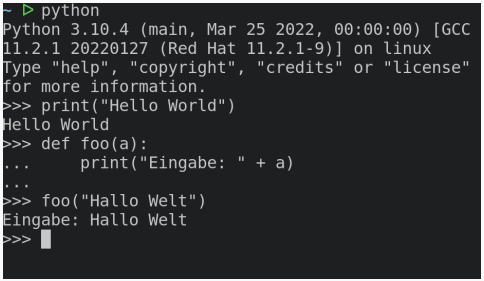

Figure 1: Beispiel Python Interpreter

[6] https://docs.python.org/3/tutorial/interpreter.html

IPython[7]

- Simpler Aufruf aus der Kommandozeile
- Aufgemotzter Interpreter
- Ermöglicht auch tlw. Visualisierungen
- Autovervollständigung, Inline-Hilfe, etc.

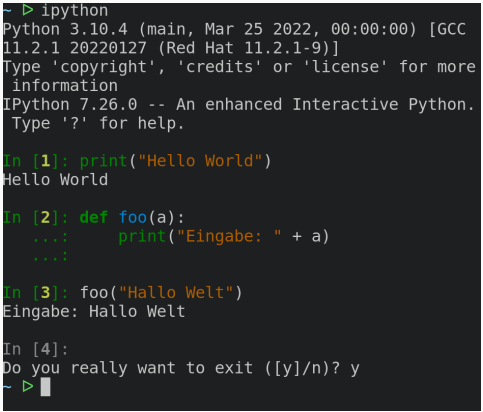

Figure 2: Beispiel IPython

[7] https://ipython.org/

Jupyter Lab[8]

- Server-Client-Modell
- Notebooks
- Codeaufteilung in Zellen
- Zellenausführung in beliebiger Reihenfolge
- Zellentypen:
  - Code
  - Markdown
  - Raw
- Erweiterungen

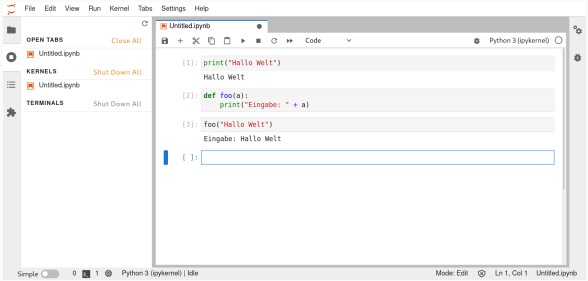

Figure 3: Beispiel Jupyter Lab Browser

[8] https://jupyter.org/

Tipps

1. IPython gut zum schnellen Testen
2. Jupyter Notebooks gut für das Testen größerer Komponenten
   - Auf Ausführungsreihenfolge achten!
   - Auswahl des Kernels bestimmt verfügbare Pakete
   - Export von `*.ipynb` nach `*.py`
3. ‘Stumpfe’ Skripte $\leftrightarrow$ ‘schlaue’ Skripte

Tool-Support
Editoren/IDEs können Programmierung unterstützen (tlw. auch Notebook Support)