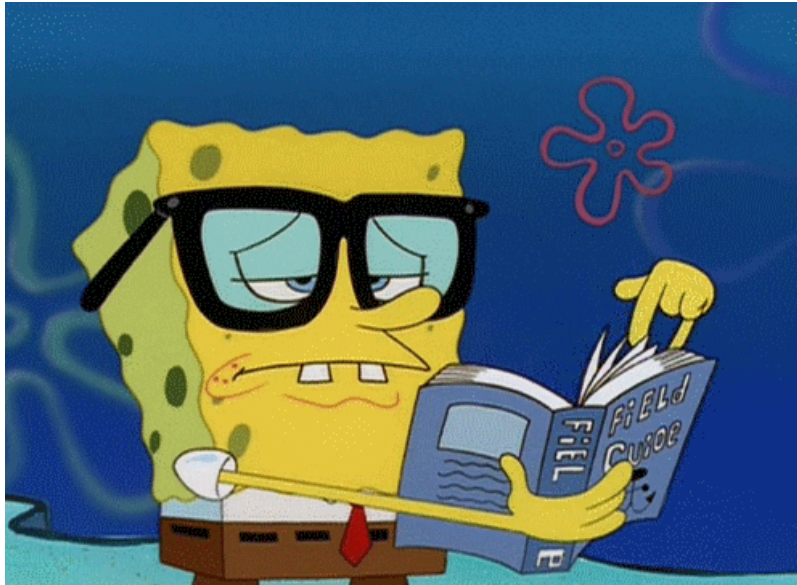


Einführung in die computerlinguistische Programmierung mit Python

07-01: Recherchieren mit der Python-Dokumentation 📖



Programmieren zu können, bedeutet nicht, dass man alle Funktionalitäten und Regeln auswendig können muss. Es ist gut, wenn man vieles im Kopf hat, aber am wichtigsten ist, dass man in der Lage ist, zu **recherchieren** und so genau die Information zu finden, die man braucht, um ein gegebenes Problem zu bearbeiten.

Für Python gibt es zwei wichtige Quellen, wenn man etwas nachschlagen möchte:

1. Die **offizielle Python-Dokumentation** ist eine Art umfangreiche Bedienungsanleitung und listet alle Funktionalitäten auf, die in Python zur Verfügung stehen.
2. Community-Plattformen wie **StackOverflow** sind dynamisch, sie enthalten nicht nur Informationen, sondern auch Diskussionen, Empfehlungen und Erfahrungsberichte.

Hier geht es zunächst um die offizielle Python-Dokumentation und wie wir sie benutzen.

Die Python-Dokumentation als ständiger Begleiter

Möglicherweise habt ihr schon Teile der Python-Doku gesehen und genutzt, wenn ihr die Übungsaufgaben im Kurs bearbeitet habt. In VSCode werden kleine Ausschnitte aus der Doku angezeigt, wenn man eingebaute Befehle wie z.B. `range()` benutzt:

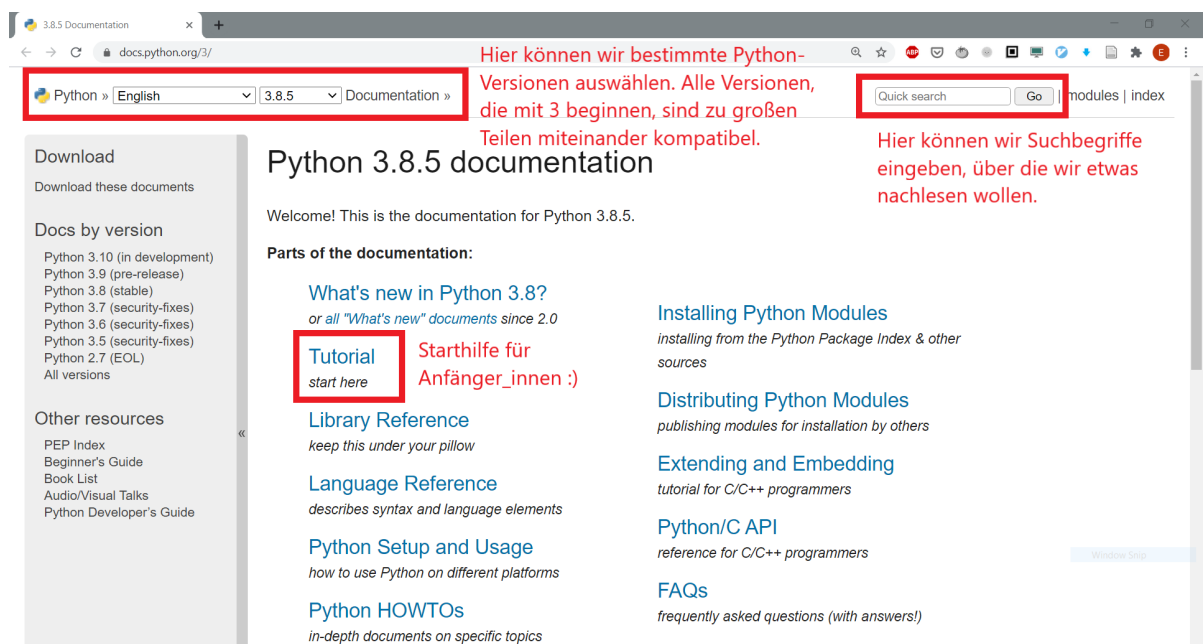
```
▶ M1  
for i in range  
range  
{}
```

Um den Hinweis aus der Dokumentation anzuzeigen, falls er nicht von alleine erscheint, können wir den Cursor ans Ende des Wortes "range" setzen und die Tastenkombination `Strg + [Leerzeichen]` nutzen, um den Tooltip **auszulösen**. Die erscheinende Nachricht beschreibt die Nutzung von `range()` und die Bedeutung der unterschiedlichen Argumente, die in den Klammern auftauchen können. So kann man zum Beispiel schnell mal nachschauen, ob die beiden Zahlen inklusive oder exklusive sind, ohne das ausprobieren zu müssen.

Die Python-Dokumentation als Online-Ressource

Die volle Python-Doku ist online unter <https://docs.python.org/3/> zu erreichen. Dort findet man Tutorials, FAQs, Informationen zur Installation, und am allerwichtigsten, eine umfassende Auflistung aller eingebauten Befehle mit **Nutzungsbeispielen** und **detaillierten Erklärungen**, sortiert nach Themen.

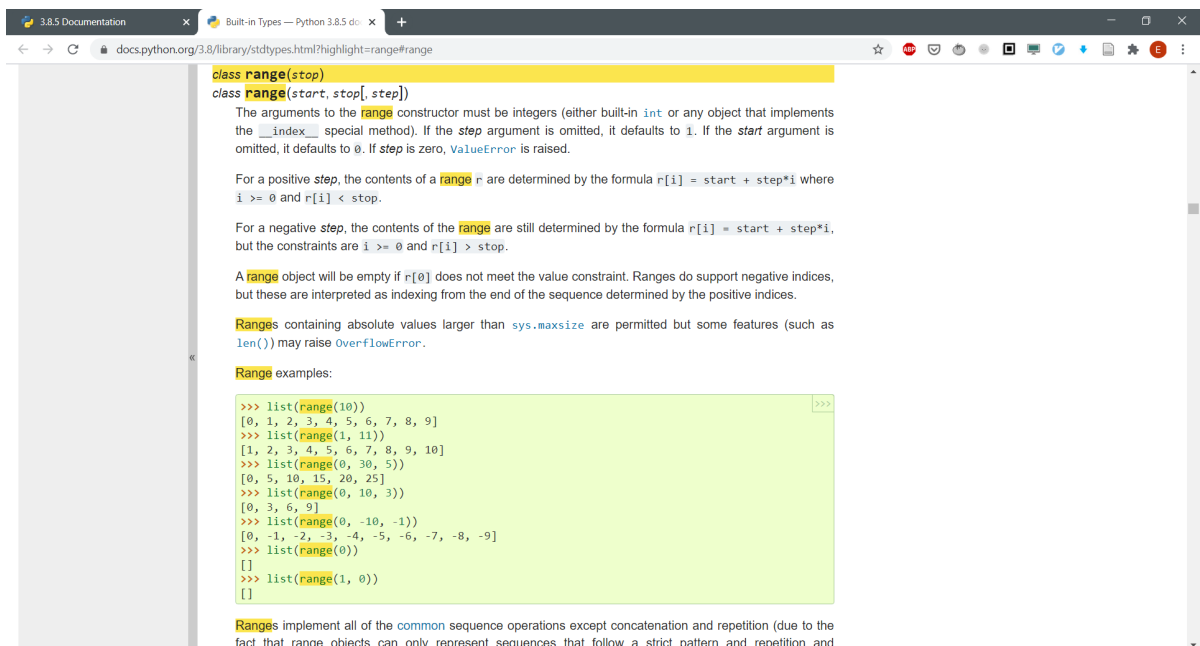
Da Python ständig weiterentwickelt wird, gibt es auch bei der Dokumentation **unterschiedliche Versionen**. Normalerweise reicht es, wenn wir darauf achten, dass wir die Doku für Python 3 verwenden und nicht für die veraltete Version, Python 2. Wenn wir es ganz genau wissen wollen, können wir im Drop-Down-Menü am oberen Rand **genau die Version** aktivieren, die bei uns installiert ist. Dabei ist die erste Zahl am wichtigsten (entweder 2 oder 3). Die zweite Zahl kennzeichnet Versionsschritte, die typischerweise untereinander kompatibel sind, bei denen aber eine spätere Version (z.B. Python 3.8) in manchen Bereichen mehr Funktionen bietet als eine frühere Version (z.B. Python 3.5). Die dritte Zahl (falls vorhanden) können wir ignorieren, weil sie nur kleinere "Bugfixes" und andere Optimierungen kennzeichnet, die sich hinter den Kulissen abspielen und die für uns als Python-Anfänger_innen nicht so bedeutsam sind.



`range()` in der Python-Doku nachschlagen

Um auf der Doku-Webseite nachzuschlagen, wie `range()` benutzt wird, können wir das **Suchfeld** am oberen Rand verwenden und dort den Befehl eintippen. In den Suchergebnissen sollte die Information, die wir suchen, am Anfang erscheinen. Ein Klick auf dieses Suchergebnis öffnet diese Seite: <https://docs.python.org/3.8/library/stdtypes.html?highlight=range#range>

Die Erklärung zum `range()`-Befehl ist hier etwas **ausführlicher** als in VSCode. Zum Beispiel gibt es einen speziellen Hinweis, was passiert, wenn man negative Werte als Argumente angibt. Darunter folgen **Verwendungsbeispiele**, die das Verhalten von `range()` veranschaulichen. Im Text sind einzelne Begriffe mit Links hinterlegt, die via Querverweis auf andere Stellen der Dokumentation zeigen, ähnlich wie in einem Wiki.



Neue Pythonbefehle in der Dokumentation lernen

Es kann vorkommen, dass wir wissen, was unser Code machen soll, aber wir noch **keinen Befehl kennen**, der uns dabei hilft. Wir können dann Suchbegriffe in der Doku eingeben und hoffen, dass wir etwas nützliches finden, aber diese Art von Recherche ist hier etwas schwieriger, als wenn man den genauen Namen eines Befehls eingibt.

Wir kennen schon das Modul `random` und haben die darin enthaltene Funktion `randint()` verwendet, um Zufallszahlen zu erzeugen. Wenn wir in einem anderen Kontext einen Befehl brauchen, der **aus einer Liste ein zufälliges Element auswählt**, können wir versuchen, den passenden Befehl dafür in der Dokumentation zu finden.

Der Suchbegriff "random" führt uns auf eine Seite, die alle Befehle erklärt, die im Modul `random` enthalten sind: <https://docs.python.org/3.8/library/random.html?highlight=random#module-random>

Wir wissen noch nicht genau, wonach wir suchen, also lesen wir ein bisschen quer und überlegen, welche Befehle wohl nützlich sein könnten. Am linken Rand ist ein **Inhaltsverzeichnis** zu sehen, in dem unter anderem ein Eintrag "Functions for sequences" auftaucht. Wir können an diese Stelle auf der Seite springen und anhand der Beschreibungen der verschiedenen Befehle überlegen, wie wir unser Programmierziel am besten erreichen können.

Die Beschreibung von `random.choice(seq)` scheint zu passen:

Return a random element from the non-empty sequence `seq`. If `seq` is empty, raises `IndexError`.

Wir verwenden also unsere Liste als Argument für `seq`. Wenn wir diesen Befehl zum ersten Mal verwendet haben, können wir einen Link zur Dokumentation als Kommentar in unseren Code einfügen, um später nachlesen zu können, wie `random.choice()` genau funktioniert.

Hier noch der Beweis, dass `random.choice()` der richtige Befehl für unser Ziel war:

```
In [9]: import random

lunch_options = ["Pasta", "Sandwich", "Cereal", "Salad"]

# random.choice in der Python-Doku:
# https://docs.python.org/3/library/random.html#random.choice
print(random.choice(lunch_options))
```

Salad

Zusammenfassung

- VSCode kann kleine Ausschnitte aus der Python-Dokumentation anzeigen, während man gerade programmiert. Diese Tooltips können mit `Strg + [Leertaste]` ausgelöst werden.
- Die Tooltips sind sehr knapp und dienen hauptsächlich als Erinnerungstütze, wenn man den Befehl schon kennt.
- Für ausführlichere Erklärungen kann man in der offiziellen Python-Dokumentation unter docs.python.org nachlesen.
- Wenn man noch keinen Befehl kennt, der das tut, was man gerade erreichen möchte, kann man in der Doku so lange herumstöbern, bis man einen passenden Vorschlag findet.

Weitere Themen dieser Woche

- 07-02: StackOverflow