

# LU08c - Python in der Linux-Shell

## Pfade in WSL

Linux und Windows unterscheiden sich erheblich in der Art und Weise, wie sie mit Ordnerpfaden umgehen. Hier sind die wesentlichen Unterschiede:

### Pfad-Trennzeichen

- **Linux:** Verwendet den Schrägstrich `/` als Trennzeichen zwischen Ordnern.
  1. Beispiel: `/home/user/documents`
- **Windows:** Verwendet den Backslash `\` als Trennzeichen.
  1. Beispiel: `C:\Users\User\Documents`

### Gross- und Kleinschreibung

- **Linux:** Unterscheidet zwischen Gros- und Kleinschreibung in Dateinamen und Ordnern. Das bedeutet, dass `File.txt` und `file.txt` zwei unterschiedliche Dateien sein können.
- **Windows:** Ignoriert standardmässig die Groß- und Kleinschreibung in Dateinamen. `File.txt` und `file.txt` werden als dieselbe Datei betrachtet.

### Pfadstruktur

- **Linux:**
  - Es gibt ein einheitliches Wurzelverzeichnis `/`.
  - Alle Laufwerke und Geräte werden unterhalb von `/` eingebunden (z. B. `/mnt/drive1`).
  - Das Dateisystem folgt einer hierarchischen Struktur mit standardisierten Verzeichnissen wie `/etc`, `/bin`, `/usr`, und `/home`.
- **Windows:**
  - Jeder Laufwerksbuchstabe (z. B. `C:`, `D:`) hat sein eigenes Wurzelverzeichnis.
  - Die Struktur ist nicht einheitlich und hängt von der Konfiguration und Installation ab.
  - Benutzerverzeichnisse befinden sich oft unterhalb von `C:\Users\<Benutzername>`.

### Absoluter vs. relativer Pfad

- **Linux:** Ein absoluter Pfad beginnt immer mit `/`, z. B. `/home/user/file`.
- **Windows:** Ein absoluter Pfad beginnt mit dem Laufwerksbuchstaben, z. B. `C:\Users\User\File`.



In WSL sind die Laufwerke `C:`, `D:`, ... unter `/mnt/` eingebunden. Um auf den Windowsordner `C:\BZZ\Python` zuzugreifen, verwenden Sie in WSL `/mnt/c/BZZ/Python`.

# Virtual Environment

In den meisten Fällen überlassen wir es unserer IDE, das Virtual Environment zu erstellen und zu aktivieren. Falls wir unser Programm jedoch auf einem Server ausführen wollen, haben wir nur die Shell zur Verfügung. In diesem Fall müssen wir uns selber um das venv kümmern.

## Ausgangslage

Ich gehe davon aus, dass ich einen Ordner mit einem Python-Projekt erstellt habe. Dieser kann manuell mit `mkdir coolproject` erstellt werden. Möchte ich ein Projekt von GitHub verwenden, kann ich den Ordner mit `git clone` erstellen.

```
git clone https://github.com/BZZ-Commons/python-template coolproject
```

Dieser Befehl klonet das Projekt von GitHub in den Ordner `coolproject`.

## Virtual Environment erstellen

Wir verwenden das Modul `venv` um unser [Virtual Environment](#) zu erstellen.

<code>cd coolproject</code>	Wechsle in das Verzeichnis mit deinem Pythonprojekt
-----------------------------	---

From:  
<https://wiki.bzz.ch/> - **BZZ - Modulwiki**

Permanent link:  
[https://wiki.bzz.ch/modul/m122/learningunits/lu08/linux\\_python?rev=1733818345](https://wiki.bzz.ch/modul/m122/learningunits/lu08/linux_python?rev=1733818345)

Last update: **2024/12/10 09:12**

