

# LU08a - Einführung Bash-Befehle in Python

Python ist nicht nur eine vielseitige Programmiersprache für Anwendungen, Datenanalyse und maschinelles Lernen, sondern eignet sich auch hervorragend, um mit der Kommandozeile zu interagieren und Bash-Befehle auszuführen. Diese Funktionalität ist besonders nützlich, wenn Sie Skripte schreiben, die Prozesse automatisieren, Daten sammeln oder Systemaufgaben ausführen müssen.

## Warum Bash-Befehle in Python nutzen?

Die Integration von Bash-Befehlen in Python bietet die Möglichkeit:

- vorhandene Bash-Skripte und Befehle wiederzuverwenden,
- komplexe Workflows zu automatisieren,
- sowohl die Vorteile von Python (Lesbarkeit, Datenverarbeitung) als auch von Bash (Systemzugriff) zu kombinieren.

## Sicherheitshinweise

Eingabesanitierung: Vermeiden Sie die direkte Übergabe von Benutzereingaben an Shell-Befehle, um Sicherheitsrisiken wie Code-Injection zu verhindern. Beispiel:

```
# Unsicher: Benutzer kann Schadcode einschleusen  
os.system(f"rm -rf {user_input}")
```

Stattdessen:

```
subprocess.run(['rm', '-rf', user_input], check=True)
```

## Fehlerbehandlung

Überprüfen Sie Rückgabecodes und Ausgaben auf Fehler.

## Fazit

Python bietet flexible Werkzeuge zur Integration von Bash-Befehlen und ist eine ideale Wahl für die Entwicklung leistungsstarker Automatisierungs- und Verwaltungsskripte. Das subprocess-Modul ist die bevorzugte Methode, da es präzise Kontrolle und Sicherheit bietet. Durch das Verständnis dieser Techniken können Sie die Stärken von Python und Bash optimal kombinieren, um Ihre Workflows effizienter zu gestalten.

Last update:

2024/12/02 09:16 modul:m122:learningunits:lu08:einfuehrung <https://wiki.bzz.ch/modul/m122/learningunits/lu08/einfuehrung?rev=1733127360>

---

From:

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

Permanent link:

<https://wiki.bzz.ch/modul/m122/learningunits/lu08/einfuehrung?rev=1733127360>

Last update: **2024/12/02 09:16**

