# Vorwort

Ein Script ist ein kleines Stück Code zur Lösung eines konkreten Problems oder zur Automatisierung einer lästigen Aufgabe. Sie brauchen zur Entwicklung eines Scripts weder eine Entwicklungsumgebung noch einen Compiler – ein Editor genügt. Beim Scripting ist Minimalismus die Devise. Es geht darum, mit wenig Code maximale Wirkung zu erzielen. Salopp formuliert:

**Scripting ist die Kunst,**
**IT-Probleme in zehn Zeilen Code zu lösen.**

In meinem Arbeitsumfeld sind Scripts allgegenwärtig. Ich verwende Scripts, um den Energiesparmodus meines Notebooks einzustellen, um aus Markdown-Dateien die PDF-Datei für den Druck dieses Buchs zu erzeugen, um auf Servern das Backup zu automatisieren, um E-Books mit Wasserzeichen auszustatten, um in einer Webapplikation neue Kunden einzurichten, um Dutzende von Datenbanken nach logischen Fehlern zu durchsuchen, um virtuelle Maschinen für den Unterricht zu erstellen, um Fotos zu sortieren usw.

## Bash, PowerShell oder Python?

Die »klassische« Script-Sprache ist die Bash. Ihr Name *Bourne Again Shell* ist ein Wortspiel. Der Vorgänger war die unter Unix verbreitete Bourne-Shell. Für Linux war die Lizenz der Bourne-Shell ungeeignet, weswegen ein neues Projekt entstand. Heute ist die Bash die dominierende Linux-Shell, sowohl im interaktiven Betrieb zur Ausführung von Kommandos als auch zur Script-Programmierung. macOS setzt auf die weitgehend kompatible Zsh, die auch unter Linux immer mehr Anhänger findet. Gleichzeitig wird unter macOS und Linux die Fish (*Friendly Interactive Shell*) immer beliebter. Aber weil ihre Syntax inkompatibel mit der Bash ist, bleiben selbst Fish-Fans beim Scripting lieber bei der Bash oder Zsh.

Die Bash ist populär, ihre antiquierte Syntax gewinnt aber keine Schönheitspreise. Im Gegenteil, Bash-Scripts sehen mitunter grauenhaft aus. Insofern kann man verstehen, dass Microsoft gar nicht erst versucht hat, die Bash für Windows zu adaptieren. Stattdessen hat Microsoft mit der PowerShell Grundideen klassischer Unix-Shells mit den Konzepten objektorientierter Programmiersprachen zu einer vollkommen neuen Sprache kombiniert. Das ist überraschend gut gelungen! Nicht ohne Grund schwören Windows-Administratorinnen und -Administratoren auf die PowerShell.

Python ist eigentlich keine typische Script-Sprache. Je nach Einsatzzweck ist Python die Basis für KI-Entwicklungen, ein Tool für (Natur-)Wissenschaftler oder die erste Sprache für den Einstieg in die Programmierung. Es gibt wohl keine andere Sprache, die so universell eingesetzt wird! Die Eleganz der Python-Syntax und das schier unerschöpfliche

Angebot von Erweiterungsmodulen haben dazu geführt, dass Python auch zur Systemadministration, zur Umwandlung von Dateien zwischen verschiedenen Formaten, als Datenbank-Tool oder zur Hardware-Steuerung (Raspberry Pi) verwendet wird. Python spielt seine Stärken umso mehr aus, je komplexer die Aufgabenstellung ist.

Kurz und gut, jede der drei Sprachen hat ihre Vorzüge. Auch bei meinen eigenen Scripts entscheide ich je nach Aufgabenstellung und Betriebssystem, welche Sprache am besten geeignet ist.

## Über dieses Buch

Weil es nicht die eine perfekte Script-Sprache gibt, startet dieses Buch nach der Einleitung mit Crashkursen für die Sprachen Bash, PowerShell und Python. Wenn Sie möchten, können Sie sich für den Start auf eine dieser drei Sprachen konzentrieren und Ihren Sprachwortschatz nach und nach vergrößern.

Teil II des Buchs stellt Werkzeuge und Arbeitstechniken vor, die Sie in Scripts typischerweise verwenden: Dazu zählen Kommandos zur Verarbeitung von Textdateien, CmdLets zur Anwendung regulärer Ausdrücke und Funktionen zum Umgang mit JSON- und XML-Dateien. Ich zeige Ihnen, wie Sie Scripts regelmäßig und automatisiert ausführen und wie Sie E-Mails versenden. Mit SSH können Sie Code auch auf anderen Rechnern ausführen bzw. Dateien dorthin kopieren. Git bietet die Möglichkeit, die Versionsverwaltung Ihres Codes und Scripting zu kombinieren. Viele praktische Beispiele runden das Informationsangebot ab.

Ein weiterer Aspekt ist die Optimierung der Arbeitsumgebung: Nerdfonts und Prompt Frameworks machen das Terminal zu einer Umgebung mit hohem Arbeitskomfort. SSH-kompatible Editoren erleichtern die Arbeit auf externen Rechnern (also *remote work*, *remote shells*).

In Teil III geht es schließlich um konkrete Anwendungen: Zu den wichtigsten Themen zählen Backups, Bildverarbeitung, Diagrammerzeugung, Web Scraping, die Nutzung von REST-APIs, das Versenden bzw. Verarbeiten von E-Mails, die Auswertung und Manipulation von Datenbanken, die Nutzung der Cloud, die Anwendung von KI-APIs und die Administration virtueller Maschinen.

Ich habe mich bemüht, das Buch so gut wie möglich nach einem Baukastensystem zusammenzusetzen. Sie müssen also nicht alle Kapitel linear lesen, sondern können gezielt nach den Informationen suchen, die Ihnen gerade wichtig sind. Alle Kapitel von Teil II und III beginnen mit einem kurzen Informationskasten, der die Voraussetzungen für das Kapitel zusammenfasst.

## Scripting im KI-Zeitalter

Praktisch jedes Script, das ich Ihnen in diesem Buch präsentiere, können Sie mit einem Prompt in Claude, ChatGPT oder Gemini erzeugen. Moderne KI-Tools haben das Scripting in den vergangenen Jahren revolutioniert.

Wozu dann dieses Buch? KI-Tools funktionieren gut, solange sie von Leuten bedient werden, die sich auskennen. Sie müssen also wissen, dass es überhaupt Scripting-Sprachen gibt und deren Unterschiede verstehen. Sie müssen Python-Module sauber installieren und mit PowerShell-CmdLets umgehen können. Sie müssen die elementare Grundlagen des Scriptings beherrschen. Mit diesem Basiswissen formulieren Sie die richtigen Prompts, können den resultierenden Code bewerten und sicher einsetzen. Genau diese Grundlagen vermittle ich Ihnen in diesem Buch.

Der Wert des Buches liegt also weder in einer Aufzählung von Kommandos und Optionen (was ich gelegentlich in kompakter Form dennoch gemacht habe, um Ihnen einen ersten Überblick über die Möglichkeiten zu geben), noch in den Beispiel-Scripts, die Sie bei Bedarf mit KI-Tools selbst und genau Ihren Bedürfnissen entsprechend nachbilden können. Vielmehr ist es mein Ziel, Sie mit dem hier vermittelten Grundlagenwissen dazu in die Lage zu versetzen, eigene Scripts – mit oder ohne KI-Unterstützung – effizient zu schreiben und anzuwenden.

## Neu in dieser Auflage

Für die zweite Auflage habe ich dieses Buch vollständig überarbeitet und aktualisiert. Eine Menge neuer Inhalte sind dazugekommen, unter anderem:

- Terminal Tuning (»Pimp my Terminal«), Verwendung moderner CLI-Tools
- Python-Modulverwaltung mit `uv`
- Diagramme erstellen/zeichnen
- E-Mail-Versand per Script
- KI-APIs anwenden
- Viele neue Beispiele: Claude-Code-Sessions exportieren, Web Scraping mit Playwright, GPS-Tracks aus Fotos extrahieren, WordPress-Bilder nachträglich mit Alt-Beschreibungen ausstatten usw.

## Scripting als Kernkompetenz für effiziente IT-Arbeit

Ganz egal, in welcher Sparte der IT-Industrie Sie tätig sind, an welchen Projekten Sie gerade arbeiten: Ein wenig Scripting-Know-how macht Sie in Ihrer Arbeit noch effizienter.

Konzentrieren Sie sich auf das Wesentliche, überlassen Sie die lästigen Nebenaufgaben einem Script! Dabei wünsche ich Ihnen viel Erfolg!

Michael Kofler (*https://kofler.info*)

PS: Alle Beispieldateien zu diesem Buch finden Sie hier zum Download:

*https://rheinwerk-verlag.de/6369*