EBERHARD KARLS UNIVERSITÄT TÜBINGEN

Dr. Eberle Zentrum für digitale Kompetenzen

DataBASHing - Konsolenbasics

Martin Raden

Einführung von `cd, cp, history, ln, ls, man, mkdir, mv, pwd, rm`

Die folgenden Videos von Bashino liefern einen ersten Eindruck, wie man sich auf der Kommandozeile in Ordnerstrukturen bewegt und wie Dateien und Ordner verwaltet werden können.

Video: Bashinho - #01 Befehle für Verzeichnisse (ls, cd, mkdir, tree etc.) [9 min] (ggf. Wiedergabegeschwindigkeit via “Einstellungen” ==> “Wiedergabegeschwindigkeit” ==> “1,25” erhöhen)

Das Folgevideo #2 Befehle für Dateien (touch, cp, mv, rm etc) [7 min] in der Playlist von Bashinho ist in der zweiten Hälfte etwas zu tief für diesen Kurs. Wenn man allerdings unseren “Grundlagen der Informatik” Kurs besucht hat, sollte man einige Sachen wiedererkennen und folgen können. :-D

Handhabung

Die folgenden Dinge betreffen vorrangig das Windows Terminal.

- **Markieren von Text**: LINKE Maustaste oder SHIFT+Cursortasten
- **Kopieren von Textmarkierung**: STRG+C oder RECHTE Maustaste
- **Einfügen** (vom zuletzt Kopierten): STRG+V oder RECHTE Maustaste
- **Blockmarkierungen**: ALT + LINKE Maustaste
- **Letzte(r) Befehl(e)**: Cursortasten up/down
- Innerhalb der aktuellen Eingabe zum **nächsten Wort** springen: STRG + Cursortasten links/rechts
- **Autovervollständigung**(svorschläge) der Eingabe: TAB
- (Windows) Zwischen Tabs wechseln: STRG+TAB
- (Windows) Zwischen Fenstern wechseln: ALT+TAB

Pfade und Trennzeichen

Im allgemeinen gilt in Linux Shells:

- Es wird zwischen **absoluten** und **relativen Pfaden** unterschieden:
  - absolut = vollständiger Pfad der immer mit einem “`/`” (der Wurzel, also obersten Ebene des Verzeichnisbaumes) beginnt
  - relativ = Pfad in Relation zum derzeitigen Verzeichnis, beginnt mit
    - “`.`” (repräsentiert aktuelles Verzeichnis) oder
    - “`..`” (repräsentiert übergeordneten Ordner)

  - Ordnernamen werden mit “ `/` ” getrennt (Unterschied zu PowerShell/DOS, wo mit “ `\` ” getrennt wird!)
  - ein “ `*` ” in einem Datei oder Pfadnamen dient als **Platzhalter** für kein oder *mehrere Zeichen*
  - ein “ `?` ” ist ein Platzhalter für *genau ein Zeichen*
- “ `;` ” **beendet** das vorangehende Kommando, sodass mehrere Kommandos hintereinander in einer Zeile geschrieben werden können
- Ordner-/Dateinamen mit **Leer-/Sonderzeichen** müssen “gequotet” (`cs "Mein Ordner"`) oder “escaped” (`cd Mein\ Ordner`) werden , daher besser derartige Namen vermeiden
- alles nach einer “ `#` ” wird ignoriert, da dieses Zeichen einen **Kommentar** einleitet
- **case sensitivity** bzw. Befehlen und Datei-/Ordnernamen, d.h. Gross-/Kleinschreibung ist wichtig

---

Basisbefehle. . .

. . . die man wissen muss:

- `ls` = listet aktuellen **Ordnerinhalt** auf (list)
  - “ `-l` ” = Tabellenansicht des Inhalts (ggf. Kurzform/Alias “ `ll` ” verfügbar)
- `pwd` = gibt den (absoluten) **Pfad des aktuellen Ordners** aus (print working directory)
- `cd` = **wechselt das Verzeichnis** (change directory) :
  - ohne Argument ins home dir
  - mit Argument zum entsprechenden Ort
  - mit “ `..` ” in den übergeordneten Ordner
  - Beispiele
- `man` = liefert eine **Hilfeseite** zum nachfolgenden Befehl (manual)
  - browsen mit Cursortasten
  - Taste `h` liefert Befehlsmöglichkeiten
  - verlassen mit Taste `q`
- `rm` = **löschen** (remove)
- `mkdir` = **Ordner erstellen** (make directory)
- `cp` = **kopieren** einer Datei (copy)
  - “ `-r` ” = auch Unterverzeichnisse und deren Inhalt (recursive)
- `ln` = **Verknüpfung/Shortcut** zu Datei (oder Verzeichnis) anlegen (link)
  - “ `-s` ” = **WAS MAN MEISST WILL** = symbolische Verknüpfung, d.h. link ist quasi Pfad zum Ziel
    - wenn Ziel gelöscht ist der Link ungültig (dangling)
    - wenn Ziel (gleichnamig) ersetzt, zeigt der Link auf neue Variante
    - geht auch mit Verzeichnissen
  - Standardfall = hard link, d.h. Link ist quasi zweite Pseudodatei die direkt auf der gleichen Datenbasis arbeitet
    - geht nur für Dateien (da Verzeichnisse nicht wirklich auf Festplatte liegen)
    - wenn Zieldatei gelöscht, bleibt der Link intakt, da er immer noch auf die Datenbasis zeigt und dort operiert (nur alternativer Zugang zu den Daten gelöscht)
- `mv` = **verschieben oder umbenennen** (move)
  - Beispiele
- `cat` = **Inhalt** einer oder mehrerer Dateien **ausgeben** (concatenate)
- `less` = Inhalt einer **Datei betrachten** (Dateibrowser)
  - Steuerung siehe “ `man` ” (weil letzteres less benutzt)
  - kann **auch als letztes Kommando** in einer pipe verwendet werden, um die aktuelle Ausgabe zu testen
- `history` = Liste der **zuletzt eingegebenen Befehle**
  - “ `!XYZ` ” = sucht in der history den letzten Befehl, der mit “ `XYZ` ” beginnt *und führt ihn erneut aus*
  - “ `!XYZ:p` ” = Ausgabe des letzten Befehls, der mit “ `XYZ` ” beginnt

  - “ `!$` ” = liefert den Argumententeil des letzten Befehls (“ `!$:p` ” gibt ihn nur aus), um ihn z.B. im nächsten Befehl zu verwenden
- `touch`
  - erzeugt eine *leere* Datei (wenn noch nicht existent)
  - aktualisiert den “Bearbeitet” Zeitstempel auf jetzt (wenn Datei schon da)

Befehl abbrechen

Wenn man

- einen Befehl unvollständig eingegeben hat und die Konsole “hängt” (d.h. auf Eingabe wartet), oder
- man den gerade aktiven Befehl stoppen will (weil er zu lange dauert oder gaaaanz viel Output auf die Konsole schreibt),

kann man den Befehl mit “**STRG+C**” abbrechen (cancel).

> Tutorials <

Online Tutorials

- Shell Intro von swcarpentry
- Navigating Files and Directories von swcarpentry
- Working with Files and Directories von swcarpentry
- man Usage and Interpretation von carpentries-incubator

Hard vs. Soft Links

Und wer noch herausfinden will, was der Unterschied zwischen “hard” und “soft” links ist, der sollte sich folgendes Video anschauen.

Video: (en) Pedagogy - soft link vs hard link in linux | inode in linux | ln command in linux | link count field in ls -l [17 min]

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License
by Dr. Eberle Zentrum für digitale Kompetenzen, Universität Tübingen