Zusammenfassung Tag 18

Mehrere Befehle kombinieren

- **Befehle können mit einem Simikolon aneinander gehangen werden**
  - *ls; ls*
  - Falls das erste Programm einen Fehler wirft, wird das zweite trozdem ausgeführt
  - unabhänige Befehle
- **Befehle können auch mit einem && verknüpft werden**
  - *ls && ls*
  - Falls der erste Befehl ein Fehler wirft wird der zweite Befehl nicht ausgeführt
  - verkettete Befehle
- **Befehle können mit einem || verknüpft werden**
  - *ls || ls*
  - Nur wenn der erste Befehl einen Fehler wirft, wird der zweite Befehl ausgeführt
  - Oder Befehle

Programme im Hintergrund laufen lassen

- ***wget https://Linkzumdownload.de/test.zip &***
  - Durch das & am Ende wird das Programm im Hintergrund ausgeführt
    und man kann die Shell weiterhin nutzen.
  - Ausgabe des Programms wird in ein log umgeleitet welches automatisch erstellt wird
- ***jobs***
  - zeigt die momentan aktiven Befehle
- ***fg***
  - bringt das zuletzt verwendete Programm wieder in den Vordergrund
- ***STRG + Z***
  - Bringt das Programm wieder in den Hintergrund, wird jedoch angehalten
- ***bg***
  - das Programm wird wieder ausgeführt
- ***ps***
  - zeigt aktive Prozesse im System
- ***kill prozessid***
  - Beendet das Programm mit der ProzessID
- ***kill -9 prozessid***
  - Beendet(killt) das Programm direkt
- ***nohup wget https://Linkzumdownload.de/test.zip &***
  - Programm läuft im Hintergrund weiter selbst wenn die Shell geschlossen wird

Bonuswissen – Programme im Hintergrund ausführen

- ***pstree***
  - gibt eine Baumstruktur mit allen aktiven Prozessen und deren unterprozesse aus
- ***pstree prozessid***
  - gibt die Baumstruktur von einem Bestimmten Prozess aus
- Wenn der Hauptprozess beendet wird werden auch alle Prozesse beendet die der Hauptprozess gestartet hat
  - nohup verhindert dies

Terminal mit anderen Leuten teilen, der Screen-Befehl

- ***sudo apt-get install screen***
  - Installiert das Programm screen
- ***screen***
  - startet das Programm
- ***screen -x***
  - verbindet sich mit der aktiven screen instanz
  - auch von einem anderer Standort aus
  - Mehrere Leute können gleichzeitig an einer Shell arbeiten (sehen die selbe Shell)
- ***STRG + A + F***
  - Passt die Shell an die Größe des Bildschirms an
- ***exit***
  - beendet den screen für alle Leute
- ***STRG + A + D***
  - screen wird verlassen läuft aber im Hintergrund weiter
- ***screen -x -r ID***
  - Falls mehrere screen aktiv sind muss beim aufrufen die entsprechende ID übergeben werden

Befehlssubstitution

- ***currentDate=$(date +"%Y-%m-%d")***
  - schreibt den Befehl date in die Variable currentDate
- ***cp hallo.txt hallo.txt.$currentDate***
  - kopiert hallo.txt und erstellt die Datei hallo.txt.2019-07-30
    - Befehl in der Variable wird ausgeführt und die ausgabe in den Namen der neuen Datei geschrieben
- ***contents=$(cat hallo.txt)***
  - echo "$contents"
    - gibt den Inhalt der Datei hallo.txt aus

Variablensubstitution

- ***echo "${d}test"***
  - gibd die Variable d aus und dahinter test
- ***echo "${d:-heute}test"***
  - überprüft ob die Variable d existiert
    - Falls ja wird Variable d + test ausgegeben
    - Falls nein wird heute + test ausgegeben
- ***echo "${e:=heute}test"***
  - überprüft ob Variable e existiert
    - Falls ja wird Variable e + test ausgegeben
    - Falls nein wird in die Variable e heute geschrieben und dann + test ausgegeben
- ***echo "${g:?Variable g existiert nicht}..."***
  - überprüft ob Variable g existiert
    - Falls Ja wird alles ausgegeben
    - Falls nein wird der String hinter dem Fragezeichen ausgegeben und der restliche Befehl abgebrochen

Befehle in einer Funktion bündeln

- ***hallo() { echo "Hallo Welt";};***
  - erstellt die funktion hallo mit dem Code: echo "Hallo Welt"
  - wenn man jetzt ein hallo in der Konsole eingibt wird die Funktion ausgeführt und in der Konsole Hallo Welt ausgegeben
- ***hallo() { echo $1}***
  - erstellt die Funktion mit einem Übergabe Parameter
  - hallo(test) gibt dann test aus

Nützliche Befehle:

clear	Bereinigt die Konsole
strg+c	Beendet ein Programm / unterbricht einen Befehl
cat	Erzeugt eine Ausgabe z.B. von einer Datei
nano	Einfacher Editor zum bearbeiten von Dateien
commandname –help	Öffnet meistens die Hilfe eines Programm
man commandname	Öffnet das Manual eines Programm falls vorhanden
type commandname	gibt aus worum es sich handelt (Befehl/Funktion/Programm)