Schnappe dir ein **Blatt Papier** und notiere dir deine Erkenntnisse zu den nachfolgenden Aufgaben!

Öffne ein **Terminal** und bearbeite die folgenden Aufgaben, um die wichtigsten Befehle kennenzulernen. Übrigens: Innerhalb des Terminals (des geöffneten Fensters) läuft wiederum die sogenannte `shell`, mit der du dann interagierst.

(A1) whoami

Aufgabe: Finde heraus, unter welchem Benutzer du aktuell angemeldet bist.
Erklärung: Der Befehl `whoami` zeigt den Benutzernamen des aktuell angemeldeten Benutzers an.

(A2) pwd

Aufgabe: Zeige das aktuelle Verzeichnis an, in dem du dich befindest
Erklärung:

- Der Befehl pwd (print working directory) zeigt den vollständigen Pfad des aktuellen *Arbeitsverzeichnisses* an (also des Verzeichnisses, in dem die `shell` / das Terminal gerade geöffnet ist).
- In Linux beginnt der Dateipfad immer beim Root-Verzeichnis, das durch einen Schrägstrich (/), auch Slash genannt, dargestellt wird.
- Im Gegensatz zu Windows, wo Pfade mit einem Laufwerksbuchstaben beginnen (z.B. C:\), gibt es in Linux kein Konzept von Laufwerksbuchstaben. Alle Dateien und Verzeichnisse befinden sich in einer einzigen hierarchischen Struktur unterhalb des Root-Verzeichnisses.

(A3) ls

Aufgabe: Liste alle Dateien und Verzeichnisse im aktuellen Verzeichnis auf.
Erklärung:

- Nutze den Befehl `ls`.
- Die bisherigen Befehle haben keine **Parameter** genutzt. Du hast also einfach nur den Namen des Befehls eingetippt und das hat genügt. Häufig benötigst du aber **Parameter**, um bestimmte Dinge zu erreichen. Tippe zuerst den Befehl, dann ein Leerzeichen und dann die gewünschten Parameter ein:
  - `-l`: Zeigt die Liste im Langformat, inklusive Details wie Berechtigungen, Anzahl der Links, Besitzer, Größe und Änderungsdatum.
  - `-a`: Zeigt alle Dateien an, inklusive versteckter Dateien, die mit einem Punkt (.) beginnen.

    - `-h`: Zeigt die Dateigrößen in einem menschenlesbaren Format (nur in Kombination mit `-l` sinnvoll).

Weitere Aufgaben (mit Parametern):

- Liste alle Dateien und Verzeichnisse im Langformat auf.

Lösung

```
ls -l
```

- Liste alle Dateien und Verzeichnisse im Langformat und in menschenlesbarem Format auf. Erkennst du den Unterschied?

Lösung

`ls -l -h` oder auch in Kurzschreibweise: `ls -lh`

- Hänge jetzt zuletzt noch den Parameter `-a` an den Befehl an und schaue, was du dann alles siehst.

Lösung

```
ls -lha
```

(A4) cd

Aufgabe: Wechsle in das Verzeichnis /home.
Erklärung:

- Der Befehl `cd` (change directory) wird verwendet, um in ein anderes Verzeichnis zu wechseln.
- Nach dem Befehl musst du bei `cd` noch mit einem **Argument** angeben, wo du hinmöchtest. Dieses Argument benötigt, anders als die Parameter, **keinen** Bindestrich zu Beginn.

Lösung

```
cd /home
```

Absolute und relative Pfade:

- **Absoluter Pfad:** Ein absoluter Pfad beginnt immer mit dem Root-Verzeichnis (/) und gibt den vollständigen Pfad zu einer Datei oder einem Verzeichnis an. Beispiel:

/home/benutzername/Dokumente
- **Relativer Pfad:** Ein relativer Pfad gibt den Pfad relativ zum aktuellen Verzeichnis an. Er beginnt **nicht** mit einem Schrägstrich. Beispiel: Wenn du dich im Verzeichnis /home/benutzername befindest, ist Dokumente ein relativer Pfad, der zum Verzeichnis /home/benutzername/Dokumente führt.

Weitere Aufgaben zu `cd`:

- Bist du noch immer in `/home`? Dann bewege dich wieder relativ zu `/home/<user>`, wobei mit `<user>` dein Benutzername gemeint ist.

Lösung

```
cd <user>
```

- Wechsel zum absoluten Pfad `/var/www`

Lösung

```
cd /var/www
```

- Wechsel zum absoluten Pfad `/home/<user>/Dokumente`

Lösung

```
cd /home/<user>/Dokumente
```

- Wechsel eine Ebene **nach oben**, also zu `/home/<user>`. Das kannst du mit einem absoluten Pfad machen, es gibt aber auch den relativen Pfad-Befehl `..`, mit dem du dich relativ in das darüberliegende Verzeichnis begibst.

Lösung

```
cd ..
```

- Wechsel wieder relativ ins Verzeichnis `Dokumente`. Fachlich korrekt müsste man vom Verzeichnis `./Dokumente` reden. Wenn man vor einen Pfad einen Punkt `.` schreibt, dann ist damit immer der Pfad ausgehend vom aktuellen Verzeichnis gemeint. Man stellt mit dem Punkt also klar, dass der Pfad wirklich als relativer Pfad zum aktuellen Verzeichnis zu interpretieren ist.

Lösung

```
cd Dokumente
```

- Wechsel abschließend zum **benachbarten** Verzeichnis `/home/<user>/Desktop`. Du musst

dazu keinen absoluten Pfad eingeben, sondern kommst durch die Kombination der letzten Befehle mit einem einzigen Befehl zu diesem Pfad!

Tipp

Du musst erst eins "nach oben".

Lösung

```
cd ../Desktop
```

(A5) mkdir

Aufgabe: Erstelle ein neues Verzeichnis namens `Projekttage` im aktuellen Verzeichnis.

Erklärung: Der Befehl `mkdir` (make directory) wird verwendet, um ein neues Verzeichnis zu erstellen. Du musst (genau wie bei `cd`) noch ein Argument nach dem mkdir-Befehl eingeben.

Lösung

```
mkdir Projekttage
```

Weitere Aufgabe:

- **Prüfe** in der shell, ob der Ordner auch tatsächlich erstellt wurde! (Nutze einen der zuletzt kennengelernten Befehle.)

(A6) rm

Aufgabe: Lösche den soeben erstellten Ordner wieder.
Erklärung: Der Befehl `rm` löscht Dateien. Wenn du zusätzlich den Parameter `-r` verwendest, dann können auch Verzeichnisse gelöscht werden.

ACHTUNG: Bei der Benutzung von `rm` gibt es weder eine Nachfrage, ob du die Datei/Verzeichnis wirklich löschen willst, noch gibt es einen Papierkorb.

Wenn du etwas gelöscht hast, dann ist es weg - für immer!
Gehe also mit dem Befehl `rm` **sehr vorsichtig** um!

Lösung

```
rm -r Projekttage
```

From:
https://www.info-bw.de/ -

Permanent link:
https://www.info-bw.de/schulen:kmg:linux:shell_aufgaben:start?rev=1720775015

Last update: **12.07.2024 09:03**