## 1.3 Anwendung/Befehle und Dateisystem

### 1.3.1 Programme aufrufen

Das Starten von Programmen mit der graphischen Oberfläche ist allgemein bekannt. Bei der Verwendung des Kommandozeileninterpreters (so genannte Eingabeaufforderung) bieten sich einige weitere Möglichkeiten.

- Eine Kommandozeile besteht im einfachsten Fall aus dem Namen des zu startenden Programms, z.B. `word.exe`.
- Bei Windows kann der Anteil ab dem letzten Punkt weggelassen werden.
- Bei Windows werden Großbuchstaben und Kleinbuchstaben oft nicht unterschieden, bei Linux dagegen immer.
- Weitere Worte auf der Kommandozeile nennt man Argumente oder Parameter.
  Beispiel: `dir c:\` – hier gibt es genau einen Parameter.
- Die meisten Parameter dienen zur Auswahl oder Einschränkung dessen, worauf der Befehl angewandt wird.
  Beispiel: **`del *.xls`**
- Bestimmte Parameter dienen dazu, das Ergebnis zu beeinflussen. Sie heißen Optionen. Auf der Kommandozeile beginnen sie mit einem besonderen Zeichen (Windows: Schrägstrich, Linux: Minuszeichen).
- Beispiel unter Windows: `dir /P`
- Beispiel unter Linux: `ls -l`

### 1.3.2 Wichtige Kommandozeilen-Befehle unter Linux und Windows

| Aktion | Linux | Windows |
|---|---|---|
| Verzeichnisinh. anzeigen | **`ls`** | **`dir /w`** |
| Dto. mit Einzelheiten | **`ls -l`** | **`dir`** |
| Dto. mit versteckten Dateien | **`ls -a`** | **`?`** |
| Dto. seitenweise | **`ls \| more`** | **`dir \| more`** |
| Verzeichnis wechseln | **`cd verzname`** | **`cd verzname`** |
| aktuelles Verz. anzeigen | **`pwd`** | **`cd`** |
| ein Verz. nach oben | **`cd ..`** (mit Leerzeichen!) | **`cd ..`** |
| Verzeichnis erstellen | **`mkdir verzname`** | **`md verzname`** |
| Verzeichnis löschen | **`rmdir verzname`** | **`rd verzname`** |
| Datei kopieren | **`cp datei1 datei2`** | **`copy datei1 datei2`** |
| Datei löschen | **`rm datei`** | **`del datei`** |
| Datei anzeigen | **`cat datei`** | **`type datei`** |
| Dto. seitenweise | **`cat datei\|more`** (Ende mit Q ) | **`type datei\|more`** |
| Datei drucken | **`lpr datei`** | **`print datei`** |
| Datei umbenennen | **`mv datei1 datei2`** | **`rename datei1 datei2`** |
| Text in Datei suchen | **`grep suchwort datei`** | **`find suchwort datei`** |
| Datei im Verzeichnis suchen | **`find verz -name datei`** | **`dir /s verz/datei`** |
| Datenträger formatieren | **`mkfs /dev/sda1`** | **`format a:`** |
| Datum und Uhrzeit anzeigen | **`date`** | **`time`** |
| Hilfe zum Dienstprogramm xyz | **`man xyz`** | **`help xyz`** |

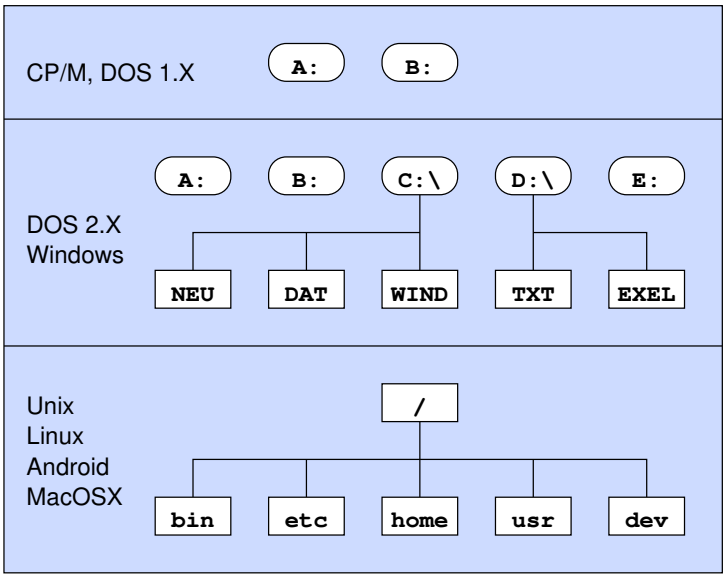


Abbildung 1: Verzeichnisbäume bei Linux, Windows und MS-DOS

### 1.3.3 Wo sind meine Dateien? – Dateisysteme

Im Hauptspeicher findet man seine Daten (z.B. Variablen) anhand ihrer Adresse. So hat jede Variable in einem C-Programm eine eigene Speicheradresse. Innerhalb eines Programmes wird diese Methode aber wesentlich vereinfacht durch Variablennamen. So fällt es dem Programmierer leichter, sich den Namen `radius` zu merken als die Adresse `0xb3b810`.

Genauso hat es sich eingebürgert, Dateien, also Speicherbereiche auf Massenspeichern (Festplatten, DVDs usw.) mit Namen zu kennzeichnen.

Mit größer werdenden Massenspeichern reichen diese Namen nicht mehr aus; die meisten Betriebssysteme ordnen daher den Namensraum weiter, indem sie Verzeichnisse (=Ordner) schaffen. Für den Benutzer sieht es nun so aus, dass jedes Verzeichnis Dateien oder aber weitere Verzeichnisse enthalten kann. Es entsteht ein baumförmiges System, der sogenannte Verzeichnisbaum. Auf diese Art erweitert sich der ursprüngliche Dateiname um die Verzeichnisse; er wird damit zum Pfadnamen.

Bei Linux sieht die Massenspeicher-Organisation (=Dateisystem) nun so aus, dass der Verzeichnisbaum genau *eine* Wurzel haben kann, von dem aus alle weiteren Verzeichnisse und alle Dateien ausgehen.

Im Gegensatz dazu hat bei Windows jeder Massenspeicher einen eigenen unabhängigen Verzeichnisbaum, beginnend mit einem sogenannten Laufwerksbuchstaben.

Der Vorteil des Windows-Verfahrens liegt in der Abwärts-Kompatibilität zu MS-DOS, der Vorteil des Linux-Verfahrens liegt darin, dass durch Auf- oder Abwärtsgehen im Baum jede Stelle des Baums erreicht werden kann. Abbildung 1 stellt die Unterschiede noch einmal dar.

Wird unter Windows ein neuer Massenspeicher installiert, erhält er vom System nach einem internen Verfahren eben einen neuen Laufwerksbuchstaben. Wird unter Linux ein neuer Massenspeicher installiert, so wird er vom System an eine Stelle des Verzeichnisbaums, nämlich an ein (meist leeres) Verzeichnis montiert, er wird gemountet (mittlerweile wird diese Technik auch bei Windows-Systemen mehr und mehr verwendet). Bei Benutzung von Wechseldatenträgern wird dieser Prozess heute meist automatisiert.

Ein weiterer Unterschied zwischen den Systemen liegt in der Darstellung des Pfadnamens einer Datei (bzw. eines Verzeichnisses): Linux benutzt den Schrägstrich (=Slash) als Trenner zwischen den Bestandteilen des Pfadnamens, Windows den Backslash (leider war bei PC-DOS der Slash

schon als Optionen-Trenner vergeben worden):
Windows: `c:\bin\turboc.hlp`
Linux: `/bin/turboc.hlp`
Allerdings kann der Pfadtrenner in beiden Systemen auf ein anderes Zeichen umgestellt werden.

### 1.3.4 Aktuelles Verzeichnis und relativer Pfadname

Standardmäßig beziehen sich viele Befehle, wenn sie ohne Argument aufgerufen werden, auf das aktuelle Verzeichnis:

- `dir` und `ls` zeigen den Inhalt des aktuellen Verzeichnisses an,
- `ls /etc` zeigt den Inhalt von `/etc` an.
- Ebenso: `find, du, ...`

Unmittelbar nach dem Einloggen ist das Heimatverzeichnis (unter Windows: `Eigene Dateien`, unter Linux: `/home/nutzername`) das aktuelle Verzeichnis. Ermitteln kann man das mit dem Befehl:

- `cd` bei Windows und
- `pwd` (=*print working directory*) bei Linux.

Danach kann der Benutzer es jederzeit selbst ändern (soweit die Verzeichnisrechte es erlauben), und zwar mit dem Befehl `cd`:

- `cd /etc` – Nun ist `/etc` das aktuelle Verzeichnis.
- `cd ..` – Hiermit kann man im Verzeichnisbaum eine Ebene höher erreichen.

Wenn man das aktuelle Verzeichnis kennt, kann man sich das Leben vereinfachen, indem man relative Pfadnamen für Dateien und Verzeichnisse verwendet. Ist man z.B. in `/home/willi`, kann man mit `rm brauchichnicht.txt` die Datei `/home/willi/brauchichnicht.txt` löschen, ohne diesen kompletten Pfadnamen zu verwenden:

- `brauchichnicht.txt` – ist ein relativer Pfadname (kurz),
- `/home/willi/brauchichnicht.txt` – ist ein absoluter Pfadname (lang).

Man muss jedoch selbst aufpassen, dass man sich im richtigen Verzeichnis befindet und die richtige Datei löscht!

Absolute Pfadnamen erkennt man nur daran, dass sie

Linux: mit einem Slash (`/`, Schrägstrich) beginnen

Windows: mit einem Backslash (Gegenschrägstrich) beginnen (Laufwerksbuchstaben zählen dabei nicht)

Bei relativen Pfadnamen fehlt dieses Zeichen am Anfang.

Relative Pfadnamen dürfen auch Verzeichnisnamen enthalten:

- `rm alt/wegdamit/brauchichnicht.txt` – löscht die gleichnamige Datei im Verzeichnis `alt/wegdamit/` unterhalb des aktuellen Verzeichnisses.
- `mkdir neu` – legt im aktuellen Verzeichnis ein Unterverzeichnis mit dem Namen `neu` an.

### 1.3.5 Abkürzungen zu Verzeichnisnamen

Damit man bei relativen Pfadnamen auch das eigene Verzeichnis insgesamt sowie das Oberverzeichnis ansprechen kann, gibt es zwei Abkürzungen:

- `..` – Zwei Punkte bezeichnen das Oberverzeichnis des aktuellen Verzeichnisses. Somit kann man mit `cd ..` eine Ebene höher wechseln.
- `.` – Ein Punkt bezeichnet das aktuelle Verzeichnis (so kann man bei DOS/Windows mit `DEL.` den kompletten Inhalt des aktuellen Verzeichnisses löschen).

Die Linux-Shell versteht dazu eine weitere Abkürzung, die nichts mit dem aktuellen Verzeichnis zu tun hat:

- ~ – Tilde bezeichnet das Heimatverzeichnis des Benutzers.

### 1.3.6 Editoren

Soll eine Textdatei erstellt werden, kann dies am Anfang am besten mit einem geeigneten Editor geschehen. Hier ist eine Auswahl.

- Linux-Textmodus:
  - `joe`: ähnlich dem Turbo-Pascal-Editor - **empfehlenswert**, Hilfe mit Strg K – H
  - `pico`: einfacher Editor des Email-Clients pine, auch aufrufbar unter dem Namen `nano`
  - `vi`: klein und überall vorhanden, aber schwierig zu bedienen
  - `emacs`: riesig, unbegrenzt erweiter- und anpassbar
  - `ue`: der Micro-Emacs, kleiner und einfacher als Emacs
- Linux-Grafikmodus:
  - `kate`: einfach zu bedienen
  - `gedit`: genauso
- Windows-Textmodus:
  - `edit.exe`: einfach zu bedienen
- Windows-Grafikmodus:
  - `notepad.exe`: Standard-Werkzeug
  - `wordpad.exe`: Bessere Erkennung von Dateiformaten